IBM WebSphere Application Server for z/OS, Version 8.0

Setting up the application serving environment

IBM

Contents

How to send your comments

Your feedback is important in helping to provide the most accurate and highest quality information.

- To send comments on articles in the WebSphere Application Server Information Center
    1. Display the article in your Web browser and scroll to the end of the article.
    2. Click on the **Feedback** link at the bottom of the article, and a separate window containing an e-mail form appears.
    3. Fill out the e-mail form as instructed, and click on **Submit feedback** .
- To send comments on PDF books, you can e-mail your comments to: **wasdoc@us.ibm.com** or fax them to 919-254-5250.

    Be sure to include the document name and number, the WebSphere Application Server version you are using, and, if applicable, the specific page, table, or figure number on which you are commenting.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

ix

Changes to serve you more quickly

Print sections directly from the information center navigation

PDF books are provided as a convenience format for easy printing, reading, and offline use. The information center is the official delivery format for IBM WebSphere Application Server documentation. If you use the PDF books primarily for convenient printing, it is now easier to print various parts of the information center as needed, quickly and directly from the information center navigation tree.

To print a section of the information center navigation:

1. Hover your cursor over an entry in the information center navigation until the **Open Quick Menu** icon is displayed beside the entry.
2. Right-click the icon to display a menu for printing or searching your selected section of the navigation tree.
3. If you select **Print this topic and subtopics** from the menu, the selected section is launched in a separate browser window as one HTML file. The HTML file includes each of the topics in the section, with a table of contents at the top.
4. Print the HTML file.

For performance reasons, the number of topics you can print at one time is limited. You are notified if your selection contains too many topics. If the current limit is too restrictive, use the feedback link to suggest a preferable limit. The feedback link is available at the end of most information center pages.

Under construction!

The Information Development Team for IBM WebSphere Application Server is changing its PDF book delivery strategy to respond better to user needs. The intention is to deliver the content to you in PDF format more frequently. During a temporary transition phase, you might experience broken links. During the transition phase, expect the following link behavior:

* Links to Web addresses beginning with http:// work
* Links that refer to specific page numbers within the same PDF book work
* The remaining links will *not* work. You receive an error message when you click them

Thanks for your patience, in the short term, to facilitate the transition to more frequent PDF book updates.

Chapter 1. Configuring port settings

When you configure WebSphere® Application Server resources or assign port numbers to other applications, you must avoid conflicts with other assigned ports. In addition, you must explicitly enable access to particular port numbers when you configure a firewall.

Before you begin

You can also use the port validator tool to find port conflicts between different WebSphere Application Server profiles, products, and servers. Read the "port validator tool" article in the information center for more information.

Tip: Port conflicts might occur if you install WebSphere Application Server on multiple systems with deployment managers managing servers or clusters on different systems. The configuration-service port-resolution mechanism does not support cross profiles on different host machines.

- **Example 1:**
    1. On system A, create a cell profile that includes Dmgr and AppSrv01 (Node1).
    2. On system B, create AppSrv01 and federate AppSrv01 (Node2) to Dmgr on system A.
    3. Create server1 on Node1 and server2 on Node2.
    4. The server1 server and server2 server might contain duplicate server endpoint ports in the `serverindex.xml` file because Node1 and Node2 are located on different host systems.
- **Example 2:**
    1. On system A, create a cell profile that includes Dmgr and AppSrv01 (Node1).
    2. On system B, create AppSrv01 and federate AppSrv01 (Node2) to Dmgr on system A.
    3. On system B, create JobManager.
    4. Create a cluster and add two servers, server1 on Node1 and server2 on Node2.
    5. The server2 server and the JobManager server might contain duplicate server endpoint ports in the `serverindex.xml` file because server2 and JobManager are in cross profiles. The server2 server is under Dmgr, JobManager is under the JobManager profile. and the Dmgr and JobManager profiles are located on different machines.

About this task

Procedure
1. Review the port number settings, especially when you are planning to coexist.
2. Optional: Change the port number settings.

   You can set port numbers when configuring (customizing) the product after installation. Start thinking about port numbers during the planning phase described in the "Planning for product configuration" article in the information center.

Updating ports in existing profiles

Use the `updatePorts.ant` script to change ports in an installed profile.

Before you begin

Each profile template has its own `updatePorts.ant` script.

The `updatePorts.ant` script for application server profiles is in the *app_server_root*/profileTemplates/*template_name*/actions directory. To use the script, you have to identify which profile to update.

Note: You should only run this script if the profile is unfederated and if the configuration is the same structure as it was when the profile was created. For example, this script is ideal for changing ports for an unfederated application server profile after you created the profile but before you altered its configuration. For all other situations, use the techniques described in "Setting port numbers kept in the serverindex.xml file using scripting".

About this task

Use the following procedure to become familiar with using the `updatePorts.ant` script. Each step is an exercise that results in reassigning ports using a particular method that the `updatePorts.ant` script supports.

Look at steps for all of the operating systems mentioned. The differences are mainly in the extension of the script file and the direction of the directory delimiters. For example, Linux shell scripts (`*.sh`) and other commands require a `./` before the command to tell the operating system that the command is in the current working directory.

Chapter 2. Administering nodes and resources

You can monitor and control incorporated nodes and the resources on those nodes by using these tasks with the administrative console or other administrative tools.

About this task

After you set up the WebSphere Application Server, Network Deployment product, you mainly need to monitor and control incorporated nodes and the resources on those nodes by using the administrative console or other administrative tools. Use the following tasks to perform these activities.

Procedure
- Manage nodes.
- Configure cells.
- Configure deployment managers.
- Manage node agents.
- Manage node groups.
- Administer stand-alone application servers on the same computer using an administrative agent.
- Administer stand-alone application servers and deployment managers remotely using a job manager.
- Configure remote file services.
- Use the settings page for an administrative service to configure administrative services.
- Configure location service daemons on the z/OS® system.
- Administer job managers.
- Change the host name.

What to do next

Administer nodes and node resources as needed using the administrative console or other administrative tools.

Working with nodes - groups of managed servers

A node is a grouping of managed or unmanaged servers. You can add both managed and unmanaged nodes to your product topology. If you add a new node for an existing WebSphere application server to the network deployment cell, you add a managed node. If you create a node in the topology for managing web servers or servers other than WebSphere application servers, you add an unmanaged node. You can add, configure, remove, and otherwise work with nodes, node agents, and node groups.

Managed and unmanaged nodes

A *node* is a logical grouping of managed servers.

A node usually corresponds to a logical or physical computer system with a distinct IP host address. Nodes cannot span multiple computers.

By default, node names are based on the host name of the computer, for example MyHostNode01.

Nodes can be managed or unmanaged. An unmanaged node does not have a node agent or administrative agent to manage its servers, whereas a managed node does. Both application servers and supported web servers can be on unmanaged or managed nodes.

A stand-alone application server is an unmanaged node. The application server node becomes a managed node when it is either federated into a cell or registered with an administrative agent.

When you create a managed node by federating the application server node into a deployment manager cell, a node agent is automatically created. The node agent process manages the application server configurations and servers on the node.

When you create a managed node by registering an application server node with an administrative agent, the application server must be an unfederated application server node. The administrative agent is a single interface that monitors and controls one or more application server nodes so that you can use the application servers only to run your applications. Using a single interface reduces the overhead of running administrative services in every application server.

A managed node in a cell can have WebSphere Application Server, Java Message Service (JMS) servers (on Version 5 nodes only), web servers, or generic servers. A managed node that is not in a cell, but is instead registered to an administrative agent, can have application servers, web servers, and generic servers on the node.

An unmanaged node can exist in a cell as long as the unmanaged node only has a supported web server defined on it. Unsupported Web servers can be on unmanaged nodes only and cannot be in a cell.

You can use the command line only to create a managed node that is registered to an administrative agent.

You can create a managed node in a cell in one of the following ways:
- Administrative console
- Command line
- Administrative script
- Java program

Each of these methods for adding a node to a WebSphere Application Server, Network Deployment cell includes the option of specifying a target node group for the managed node to join. If you do not specify a node group, or you do not have the option of specifying a node group, the default node group of DefaultNodeGroup is the target node group.

On the z/OS system, the default DefaultNodeGroup node group is the sysplex node group for the deployment manager node and any other node in the cell from the same sysplex. A z/OS system node from a different sysplex cannot be a member of this node group and must be a member of a sysplex node group for its sysplex.

Whether you specify an explicit node group for a cell or accept the default, the node group membership rules must be satisfied. If the node that you are adding does not satisfy the node group membership rules for the target node group, the add node operation fails with an error message.

Each managed node that is joined to a cell must be a member of a node group. However, a managed node that is registered to an administrative agent cannot be a member of a node group.

The concepts of managed and unmanaged nodes are not applied to the registration of nodes to the job manager.

Administrative functions for web server nodes

WebSphere Application Server supports basic administrative functions for all supported web servers. For example, the generation of a plug-in configuration can be performed for all web servers. However, propagation of a plug-in configuration to remote web servers is supported only for IBM® HTTP Servers that

are defined on an unmanaged node. If the web server is defined on a managed node, propagation of the plug-in configuration is done for all the web servers by using node synchronization. The web server plug-in configuration file is created according to the web server definition and is based on the list of applications that are deployed on the web server. You can also map all supported Web servers as potential targets for the modules during application deployment.

WebSphere Application Serversupports some additional administrative console tasks for IBM HTTP Servers on managed and unmanaged nodes. For instance, you can start IBM HTTP Servers, stop them, terminate them, display their log files, and edit their configuration files.

Node groups

A *node group* is a collection of managed nodes. Managed nodes are WebSphere Application Server nodes. A node group defines a boundary for server cluster formation.

- "Node groups"
- "Sysplex node groups"
- "Example: Using node groups" on page 6

Node groups

Nodes that you organize into a node group need to be similar in terms of installed software, available resources, and configuration to enable servers on those nodes to host the same applications as part of a server cluster. The deployment manager does no validation to guarantee that nodes in a given node group have anything in common.

Node groups are optional and are established at the discretion of the WebSphere Application Server administrator. However, a node must be a member of a node group. Initially, all Application Server nodes are members of the default DefaultNodeGroup node group.

A node can be a member of more than one node group.

Nodes on distributed platforms and the IBM i platform cannot be members of a node group that contains a node on a z/OS platform. However, nodes on distributed platforms and nodes on the IBM i platform can be members of the same node group.

To delete a node group, the node group must be empty. The default node group cannot be deleted.

An Application Server node must be a member of a sysplex node group. Nodes in the same sysplex must be in the same sysplex node group. A node can be in one sysplex node group only.

When the deployment manager is configured on a z/OS node, the default node group, DefaultNodeGroup, is the sysplex node group for the deployment manager node and any other node in the cell from the same sysplex. Sysplex node groups are special node groups that the system manages.

Sysplex node groups

A sysplex node group is a node group unique to the z/OS operating system. The sysplex node group includes a sysplex name and a z/OS operating system location service configuration. A sysplex is a collection of z/OS systems that cooperate by using certain hardware and software products to process workloads.

You cannot explicitly create a sysplex node group. The z/OS operating system creates sysplex node groups in the following ways:

- When you configure a deployment manager server on the z/OS operating system, the default node group is a sysplex node group. The deployment manager is automatically a member of the sysplex node group. Application Server for z/OS nodes that you add to the network deployment cell are automatically members of this node group.
- You can add an Application Server for z/OS node to a network deployment cell whose deployment manager is on a distributed platform node. In this case, you must add the first Application Server for z/OS node for the network deployment cell to an empty node group. The system automatically configures the node group into a sysplex node group by using the sysplex name and the z/OS location service configuration that belongs to the Application Server for z/OS node.

You cannot remove a node from a sysplex node group. However, if a node is the only member of a sysplex node group, you can add that node to an empty node group. The empty node group is converted into a sysplex node group and the former sysplex node group of the node is converted into a regular node group.

You cannot delete a node group that is a sysplex node group.

Example: Using node groups

By organizing nodes that satisfy your application requirements into a node group, you establish an administrative policy that governs which nodes can be used together to form a cluster. The people who define the cell configuration and the people who create server clusters can operate with greater independence from one another, if they are different people.

Example 1

Assume the following information:
- A cell is comprised of nodes one to eight.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes six, seven, and eight are additionally configured as WebSphere Business Integration Server Foundation nodes.
- All nodes are either z/OS system nodes from the same sysplex, or some combination of distributed platform nodes and IBM i platform nodes.
- By default, all the nodes are in the default DefaultNodeGroup node group.

Applications that exploit WebSphere Business Integration Server Foundation functions can run successfully only on nodes six, seven, and eight. Therefore, clusters that host these applications can be formed only on nodes six, seven, and eight. To define a clustering policy that guides users of your WebSphere cell into building clusters that can span only predetermined nodes, create an additional node group called WBINodeGroup, for example. Add to the node group nodes six, seven, and eight. If you create a cluster on a node from the WBINodeGroup node group, the system allows only nodes from the WBINodeGroup node group to be members of the cluster.

Example 2

Assume the following information:
- A cell is comprised of nodes one to six.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes one to four are some combination of distributed platform nodes and IBM i platform nodes.
- Nodes five and six are nodes on the z/OS operating system and are in the PLEX1 sysplex.
- The deployment manager is on a distributed platform node.
- Nodes one to four are members of the DefaultNodeGroup node group by default.

- You created empty PLEX1NodeGroup node group to group the z/OS operating system nodes on the PLEX1 sysplex.
- You joined the nodes on the z/OS operating system to the PLEX1NodeGroup node group when you added them to the cell. Nodes on the z/OS operating system cannot be in the same node group with the distributed platform nodes.

Applications that exploit z/OS functions in the PLEX1 sysplex can run successfully on nodes five and six only. Therefore, clusters that host these applications can be formed only on nodes five and six. The required separation of distributed platform nodes and IBM i platform nodes from z/OS system nodes establishes a natural clustering policy that guides users of your Application Server cell into building clusters that can span only predetermined nodes. If you create a cluster on a node from the PLEX1NodeGroup node group, the system allows only nodes from the PLEX1NodeGroup node group to be members of the cluster.

Example: Using node groups with clusters

Use node groups to define groups of nodes that are capable of hosting members of the same cluster. An application that is deployed to a cluster must be capable of running on any of the cluster members. The node that hosts each of the cluster members must be configured with software and settings that are necessary to support the application.

By organizing nodes that satisfy your application requirements into a node group, you establish an administrative policy that governs which nodes can be used together to form a cluster. The people who define the cell configuration and the people who create server clusters can operate with greater independence from one another, if they are different people.

Example 1

Assume the following information:
- A cell is comprised of nodes one to eight.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes six, seven, and eight are additionally configured as WebSphere Business Integration Server Foundation nodes.
- All nodes are either z/OS system nodes from the same sysplex, or some combination of distributed platform nodes and IBM i platform nodes.
- By default, all the nodes are in the default DefaultNodeGroup node group.

Applications that exploit WebSphere Business Integration Server Foundation functions can run successfully only on nodes six, seven, and eight. Therefore, clusters that host these applications can be formed only on nodes six, seven, and eight. To define a clustering policy that guides users of your WebSphere cell into building clusters that can span only predetermined nodes, create an additional node group called WBINodeGroup, for example. Add to the node group nodes six, seven, and eight. If you create a cluster on a node from the WBINodeGroup node group, the system allows only nodes from the WBINodeGroup node group to be members of the cluster.

Example 2

Assume the following information:
- A cell is comprised of nodes one to six.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes one to four are some combination of distributed platform nodes and IBM i platform nodes.
- Nodes five and six are nodes on the z/OS operating system and are in the PLEX1 sysplex.
- The deployment manager is on a distributed platform node.

- Nodes one to four are members of the DefaultNodeGroup node group by default.
- You created empty PLEX1NodeGroup node group to group the z/OS operating system nodes on the PLEX1 sysplex.
- You joined the nodes on the z/OS operating system to the PLEX1NodeGroup node group when you added them to the cell. Nodes on the z/OS operating system cannot be in the same node group with the distributed platform nodes.

Applications that exploit z/OS functions in the PLEX1 sysplex can run successfully on nodes five and six only. Therefore, clusters that host these applications can be formed only on nodes five and six. The required separation of distributed platform nodes and IBM i platform nodes from z/OS system nodes establishes a natural clustering policy that guides users of your Application Server cell into building clusters that can span only predetermined nodes. If you create a cluster on a node from the PLEX1NodeGroup node group, the system allows only nodes from the PLEX1NodeGroup node group to be members of the cluster.

Adding, managing, and removing nodes

You can add a node, select the discovery protocol for a node, define a custom property for a node, stop servers on a node, and remove a node.

Before you begin

A node is a grouping of managed or unmanaged servers. You can add both managed and unmanaged nodes to the WebSphere Application Server topology. If you add a new node for an existing WebSphere Application Server to the network deployment cell, you add a managed node. If you create a new node in the topology for managing web servers or servers other than WebSphere Application Servers, you add an unmanaged node.

You can recover an existing managed node of a deployment manager cell. One of the options to add a managed node enables you to quickly recover a damaged node. The option is similar to the `-asExistingNode` parameter of the addNode command.

To view information about nodes and managed nodes, use the Nodes page. To access the Nodes page, click **System administration** > **Nodes** in the administrative console navigation tree.

About this task

You can manage nodes on an application server through the wsadmin scripting tool, through the Java application programming interfaces (APIs), or through the administrative console. Perform the following tasks to manage nodes on an application server through the administrative console.
- Add a node.
- Select the discovery protocol.
- Define a custom property for a node.
- Synchronize the node configuration.
- Stop servers on a node.
- Recover an existing managed node of a deployment manager cell.
- Remove a node.
- View node capabilities.

Procedure

- Add a node.
    1. Go to the Nodes page and click **Add Node**.
    2. On the Add Node page, choose whether you want to add a managed or unmanaged node, and click **Next**.
    3. For a managed node, complete the following actions.

a. Verify that an application server is running on the remote host for the node that you are adding.

b. Specify a host name, connector type, and port for the application server at the node you are adding. Perform one of the following sets of actions listed in the table:

Table 1. Managed node actions. Perform the set of actions appropriate for your product environment.

If the deployment manager is on	And the node that you add to the cell is on	Complete the appropriate set of actions:
A z/OS system	A z/OS system and is in the same sysplex as the deployment manager	Optionally specify a node group and a core group. Click **OK**.
A z/OS system	A z/OS system, but is on a different sysplex than the deployment manager	Specify a node group that contains nodes from the same sysplex as the node you are adding. If no such node group exists, create a node group and then specify that node group. Optionally specify a core group. Click **OK**.
The distributed platform or the IBM i platform	A z/OS system	Specify a node group that contains nodes from the same sysplex as the node you are now adding. If no such node group exists, create a node group and then specify that node group. Optionally specify a core group. Click **OK**.
A z/OS system	The distributed platform or the IBM i platform	Specify a node group that contains distributed nodes. If no such node group exists, create a node group and then specify that node group. Optionally specify a core group. Click **OK**.

For the node group option to display, a group other than the default node group must first be created. Likewise, for the core group option to display, a group other than the default core group must first be created.

c. For managed nodes, another administrative console page is displayed on a Windows operating system. Specify on the page whether you want to register the node agent to run as a Windows service.

If security is enabled, you can optionally enter the local operating system user name and password under which you will run the service. If you do not specify a user name and password, the service runs under the local system identity. When you run remove the node, the node agent is de-registered as a Window service.

4. For an unmanaged node, on the **Nodes** > **New** page, specify a node name, a host name, and a platform for the new node. Click **OK**.

The node is added to the WebSphere Application Server environment and the name of the node is displayed in the collection on the Nodes page.

Join subsequent WebSphere Application Server for z/OS nodes from the same sysplex to the same sysplex node group. If you add WebSphere Application Server for z/OS nodes from different sysplexes to the same cell, establish a separate sysplex node group for the nodes of each sysplex. On completing this step, you will have added one or more nodes.

Note: When nodes are added while LDAP security is enabled, the following exception is generated in the deployment manager `System.out` log under certain circumstances. If this happens, restart the deployment manager to resolve the problem.

```
0000004d ORBRas E com.ibm.ws.security.orbssl.WSSSLClientSocketFactoryImpl
createSSLSocket ProcessDiscovery : 0 JSSL0080E: javax.net.ssl.SSLHandshakeException -
The client and server could not negotiate the desired level of security.
Reason?com.ibm.jsse2.util.h: No trusted certificate found
```

- Select the discovery protocol.

  If the discovery protocol that a node uses is not appropriate for the node, select the appropriate protocol.

  1. On the Nodes page, click the node to access the node setting page.
  2. Select a value for **Discovery protocol**.
  3. Click **OK**.

  User Datagram Protocol (UDP) is faster than Transmission Control Protocol (TCP). However, TCP is more reliable than UDP because UDP does not guarantee the delivery of datagrams to the destination. The default of TCP is the recommended value.

  For a node agent or deployment manager, use **TCP** or **UDP**.

  A managed process uses multicast as its discovery protocol. The discovery protocol is fixed for a managed process. The main benefit of using multicast on managed processes is efficiency for the node agent. Suppose you have forty servers in a node. A node agent that uses multicast sends one broadcast to all forty servers. If a node agent did not use multicast, it would send discovery queries to all managed processes one at a time, totaling forty sends. Additional benefits of using multicast are that you do not have to configure the discovery port for each server or prevent port conflicts because all servers in one node listen to one port instead of to one port for each server.

- Define a custom property for a node.

  1. On the Nodes page, click the node for which you want to define a custom property.
  2. On the node settings page, click **Custom Properties**.
  3. On the Property collection page, click **New**.
  4. On the Custom property settings page, specify a name-value pair and a description for the property, and click **OK**.

- Synchronize the node configuration.

  After you add a managed node or change a managed node configuration, synchronize the node configuration. On the Node agents page, ensure that the node agent for the node is running. Then, on the Nodes page, select the check box beside the node whose configuration files you want to synchronize and click **Synchronize** or **Full Resynchronize**.

  Clicking either option sends a request to the node agent for that node to perform a configuration synchronization immediately, instead of waiting for the periodic synchronization to occur. This action is important if automatic configuration synchronization is disabled, or if the synchronization interval is set to a long time, and a configuration change is made to the cell repository that needs to replicate to that node. Settings for automatic synchronization are on the File synchronization service page.

  Synchronize requests that a node synchronization operation be performed using the normal synchronization optimization algorithm. This operation is fast, but might not fix problems from manual file edits that occur on the node. It is still possible for the node and cell configuration to be out of synchronization after this operation is performed.

  Full Resynchronize clears all synchronization optimization settings and performs configuration synchronization anew, so there is no mismatch between node and cell configuration after this operation is performed. This operation can take longer than the **Synchronize** operation.

  Unmanaged nodes cannot be synchronized.

- Stop servers on a node.

  On the Nodes page, select the check box beside the managed node whose servers that you want to stop running, and click **Stop**.

- Recover an existing managed node of a deployment manager cell.

  You can recover an existing damaged node using one of the options to add a managed node. The node must be at the deployment manager level.

1. Ensure that the existing damaged node is not running. Stop the node agent and any application servers residing on the node.
2. Create a profile to replace the damaged node and give it the same profile and node names.

   For example, suppose the `myNode01` node that has the profile name `AppSrv01` stops functioning. To replace it with a new node, create an application server profile named `AppSrv01` for node `myNode01`.
3. Start the new node, or application server, that you want to use to replace the damaged node.
4. Use the Recover managed node page to replace the damaged node in the cell with the new node.

   a. In the deployment manager administrative console, click **System administration** > **Nodes** > **Add Node** > **Recover an existing node** > **Next**.

   b. For **Host**, specify the host name or IP address of the node to add to the cell. The host value can be an IP address, a domain name server (DNS) name that resolves to an IP address, or the word `localhost` if the application server is running on the same machine as the deployment manager.

   c. For **JMX connector type**, select the type of Java Management Extensions (JMX) connectors that communicate with the product when you run a script.

   d. For **JMX connector port**, specify the port number of the JMX connector of the new node.

   You can find the port number in the console of the new application server node. Click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Ports**. For example, for a SOAP connector port type, specify the SOAP_CONNECTOR_ADDRESS value for the JMX connector port number.

   e. Specify values for the remaining fields as needed and click **OK**.

Instead of using the Recover managed node console page to recover a node, you can run the addNode command with the `-asExistingNode` option from a command line at the `bin` directory of the damaged application server profile. The name of the new node must match the name of the node where you run addNode with the `-asExistingNode` option.

You can also use the `-asExistingNode` option of the addNode command to move a node to a product installation on a different computer but at the same path, to move a node to a product installation on a different operating system or with a different path, or to create new cells from a template cell. See the topic on recovering or moving nodes with the addNode -asExistingNode command.

- Remove a node.

  On the Nodes page, select the check box beside the node that you want to delete and click **Remove Node**. If you cannot remove the node by clicking **Remove Node**, remove the node from the configuration by clicking **Force Delete**.

- View node capabilities.

  Review the node capabilities, such as the product version through the administrative console. You can also query them through the Application Server application programming interface (API) or the wsadmin tool. For information on the wsadmin tool, see the *Using the administrative clients* PDF.

  The product versions for WebSphere Application Server are as follows: The base edition of WebSphere Application Server is listed in the version column as `Base`. The express edition of WebSphere Application Server is listed in the version column as `Express`. The WebSphere Application Server, Network Deployment product is listed in the version column as `ND`.

What to do next

If you changed a node configuration, examine the configuration changes.

Recovering or moving nodes with the addNode -asExistingNode command

You can use the `-asExistingNode` option of the addNode command to recover and move nodes of a deployment manager. Using the `-asExistingNode` option, federate a new custom node to a deployment manager as an existing node. During federation, the product uses information in the deployment manager master configuration to transform the custom node into the existing node.

Before you begin

This topic assumes that a WebSphere Application Server, Network Deployment product has a deployment manager with one or more managed nodes.

About this task

Note: Use the -asExistingNode option of the addNode command to quickly recover a damaged node, to move a node to a product installation on a different computer but at the same path, to move a node to a product installation on a different operating system or with a different path, or to create cells from a template cell.

The following procedures describe how to use the -asExistingNode option:
- Recover an existing managed node of a deployment manager.
- Move a node to a product installation on a different computer but at the same path.
- Move a node to a product installation on a different operating system or with a different path.
- Create a cell from a template cell.

Note: Other addNode options for node configuration are incompatible with this -asExistingNode option. Do not use -asExistingNode with the following incompatible options:
- -includeapps
- -includebuses
- -startingport
- -portprops
- -nodeagentshortname
- -nodegroupname
- -registerservice
- -serviceusername
- -servicepassword
- -coregroupname
- -excludesecuritydomains

When the addNode command is run with the -asExistingNode option, the product does not check for or resolve conflicts among ports. You must verify that the ports associated with a node do not conflict with ports that are already in use on the target host.

Procedure
- Recover an existing managed node of a deployment manager.

  You can recover an existing damaged node using the -asExistingNode option of the addNode command. For example, if a computer failure results in an unavailable node but node information remains on the deployment manager, you can use the -asExistingNode option to recreate the unavailable node.

  1. Ensure that the existing damaged node is not running. Stop the node agent and any application servers that reside on the node.
  2. Remove the original profile, and create a profile to replace the damaged node and give it the same profile path, profile name, and node name as the unavailable node. Or, you can create the profile on a different computer from the original node, if your original computer is unavailable and you have configured a new computer with the same host name.

     For example, suppose the myNode01 node that has the profile name AppSrv01 stops functioning. To replace it with a new node, create an application server profile named AppSrv01 for node myNode01.
  3. Run the addNode command with the -asExistingNode option from a command line at the bin directory of the damaged application server profile.

     The name of the new node must match the name of the node where you run addNode with the -asExistingNode option.

a. Open a command prompt and change to the application server profile `bin` directory. For example, for the application server profile `AppSrv01`, go to the *profile_root*/`AppSrv01`/`bin` directory.

b. Run the addNode command with the -asExistingNode option to replace the application server node with the new node. The following example command assumes that security is enabled and that the product requires you to enter a user name and password. For *dmgr_host* and *dmgr_port*, specify the host name and port number of the deployment manager.

```
addNode dmgr_host dmgr_port -asExistingNode -username user_name -password password
```

4. Synchronize all the other active nodes in the cell.

 – The easiest and most efficient way to synchronize active nodes is to allow automatic synchronization to run. By default, automatic synchronization is enabled and nodes synchronize themselves at their configured interval.

 – If automatic synchronization is not enabled, you can synchronize the nodes explicitly.
   a. Click **System administration** > **Nodes**.
   b. On the Nodes page, select the unsynchronized nodes and click **Synchronize**.

   If you have more than five unsynchronized nodes, only synchronize five nodes at a time.

To recover a managed node using a deployment manager administrative console, see the topic on adding, managing, and removing nodes.

• Move a node to a product installation on a different computer but at the same path.

You can use the -asExistingNode option to move a node to a different computer, provided the following settings are the same on the different computer:
 – WebSphere Application Server installation directory
 – Profile name
 – Profile directory
 – Node name

This procedure involves three different profiles:
 – The *deployment manager profile* is the profile for the deployment manager. Run the changeHostName command from the deployment manager profile.
 – The *source profile* is the original profile from which you want to move.
 – The *destination profile* is the profile that you want to move to on the different computer.

1. Ensure that the node you want to move, the source profile, is not running. Stop the node agent and any application servers that reside on the node.

2. Change the host name of the node within the master configuration present at the deployment manager.

Perform the following steps, which involve the deployment manager profile:

a. Open a command prompt and change to the deployment manager profile `bin` directory. For example, if the deployment manager profile is named `Dmgr01`, go to the *profile_root*/`Dmgr01`/`bin` directory.

b. Run wsadmin Jython commands that change the host name of the node. The following example commands assume that security is enabled and that the product requires you to enter a user name and password. For *new_host_name*, specify the host name of the target computer.

```
wsadmin -lang jython -userName user_name -password password

AdminTask.changeHostName('[-hostName new_host_name -nodeName node_name]')

AdminConfig.save()

quit
```

3. Move the node from the product installation on the source computer to the product installation on the target computer.

Perform the following steps, which involve the destination profile, on the target computer:

a. Install WebSphere Application Server in a directory that has the same name as the product installation directory on the source computer.

b. Create a custom profile that has the same profile name, profile directory, and node name as the profile for the node that you want to move. When creating the custom profile, select to federate the node later. Do not select to federate the node during profile creation.

c. Open a command prompt and change to the application server profile `bin` directory. For example, if the application server profile is named `AppSrv01`, go to the *profile_root*/`AppSrv01`/`bin` directory.

d. Run the addNode command with the -asExistingNode option to replace the application server node with the node that you want to move. The following example command assumes that security is enabled and that the product requires you to enter a user name and password. For *dmgr_host* and *dmgr_port*, specify the host name and port number of the target deployment manager.

   `addNode dmgr_host dmgr_port -asExistingNode -username user_name -password password`

4. Use the administrative console of the target deployment manager or wsadmin to enable servers on the node to run properly.

   a. Start the node. This step involves the destination profile.

   b. Update the virtual hosts (host aliases) to include the target host name of the application server node.

   c. Start the application servers of the node.

5. If the node uses a Secure Sockets Layer (SSL) certificate, change the default certificate to contain the host name of the node.

   See the topic on creating SSL certificates to replace existing certificates in a node.

6. Synchronize all the other active nodes in the cell.

You might need to update the configurations of other infrastructure components, such as web servers, that are statically configured to use application servers residing on specific hosts.

- Move a node to a product installation on a different operating system or with a different path.

   You can use the `-asExistingNode` option to move a node to a product installation on a different computer with the same operating system, but different host name and path. You can also use the option to move a node to a product installation on a different computer that has a different operating system but compatible configuration files; for example, from an AIX operating system to a Windows operating system.

   Restriction:

   – Applications that use Scheduler only work with the same host name. Because the host name is embedded in each scheduled task, tasks that exist before you move a node will not work properly, but tasks created after the move will work properly. After you move a node, reschedule any scheduled tasks that existed when you moved the node.

   – You cannot move nodes between product installations on z/OS and non-z/OS operating systems.

   This task assumes that the WebSphere Application Server installation directory and profile directory on the computer that has the node you want to move (source computer) is different from the directories on the target computer. However, the node profile name and node name must be the same on both the source and target computers.

   To complete this task, perform the steps in the Move a node to a product installation on a different computer but at the same path task, except change the product installation and profile paths of each node in the variable maps on the deployment manager configuration before moving the node to the target computer. For example:

   1. In a deployment manager administrative console, click **Environment** > **WebSphere variables**.

   2. On the WebSphere Variables page, select the node scope and then click the **WAS_INSTALL_ROOT** variable.

   3. On the settings page for the WAS_INSTALL_ROOT variable, change the **Value** setting to specify the new product installation path and save the change.

4. On the WebSphere Variables page, with the node scope selected, click the **USER_INSTALL_ROOT** variable.

5. On the settings page for the USER_INSTALL_ROOT variable, change the **Value** setting to specify the new profile installation path and save the change.

6. Repeat these steps as needed to change the product installation and profile paths of each node so that the paths are correct for the target computer.

For this task, the product installation and profile directories do not need to be the same on the target computer as on the source computer.

- Create a cell from a template cell.

  You can quickly create a cell from an existing cell using the -asExistingNode option of the addNode command. The new cell must have the same name as the template cell.

  Restriction:
  - Scheduler application does not work with multiple environments. Because the host name is embedded in each scheduled task, tasks that exist before you move a node will not work properly, but tasks created after the move will work properly. After you move a node, reschedule any scheduled tasks that existed when you moved the node.
  - You must assess whether different resources, such as data sources, are required for each environment.

  If security is enabled, you likely must regenerate new keys and tokens for a new cell.

  1. Create and configure a cell to be the template cell that you want to use for new product installations.

  2. Make a copy of the deployment manager profile configuration using the backupConfig command. You will use this copy of the configuration to restore the deployment manager configuration in the new installation.

  3. Copy the template cell to a new product installation.

     For each new environment to be provisioned, complete the following steps:

     a. Install WebSphere Application Server.

     b. Create the deployment manager and application server node profiles.

     c. Restore the deployment manager profile configuration using the restoreConfig command. Update the deployment manager host name using wsadmin in local mode. If the profile path or the product installation path has changed, modify the `variables.xml` file of the deployment manager node to reflect the new paths. Update additional properties files as needed. Properties files that you might need to update include, for example, `wsadmin.properties` and `soap.client.props`.

     d. Customize each node configuration on the deployment manager profile. For example, change the following settings:
        - Host name
        - Ports
        - Product installation directory
        - Profile directories
        - Security configuration

     e. Run addNode –asExistingNode for each node. You can run the command concurrently from each node.

        1) Open a command prompt and change to the application server profile `bin` directory. For example, if the application server profile is named `AppSrv01`, go to the *profile_root*/`AppSrv01`/`bin` directory.

        2) Run the addNode command with the -asExistingNode option to replace the application server node with the node on the target cell. The following example command assumes that security is enabled and that the product requires you to enter a user name and password. For *dmgr_host* and *dmgr_port*, specify the host name and port number of the target deployment manager.

           `addNode dmgr_host dmgr_port -asExistingNode -username user_name -password password`

4. Use the administrative console of the new deployment manager or wsadmin to enable servers for each node to run properly.
   a. Start the node. Run the startNode command from the node profile.
   b. Update the virtual hosts (host aliases) to include the host name of the application server node.
   c. Start the application servers of the node.
5. If the cell uses a Secure Sockets Layer (SSL) certificate, replace the self-signed root certificate in the root keystore, DmgrDefaultRootStore.

   See the topic on creating SSL certificates to replace existing certificates in a cell.
6. Synchronize all the other active nodes in the cell.

What to do next

Examine the nodes in the target installation to ensure that the node configuration operates properly. If necessary, delete profiles of the source installation.

Node collection

Use this page to manage nodes in the WebSphere Application Server environment. Nodes group managed servers. The table lists the managed and unmanaged nodes in this cell. The first node is the deployment manager. Add new nodes to the cell and to the list by clicking **Add Node**.

To view this administrative console page, click **System administration** > **Nodes**.

Name:

Specifies a name for a node that is unique within the cell.

A node corresponds to a physical computer system with a distinct IP host address. The node name is usually the same as the host name for the computer.

Version:

Specifies the product name and version number of the node.

The product version is the version of a WebSphere Application Server for managed nodes.

For unmanaged nodes on which you can define web servers, the version displays as not applicable

The base edition of WebSphere Application Server (base) is listed in the version column as `Base`. The express edition of WebSphere Application Server is listed in the version column as `Express`. The WebSphere Application Server, Network Deployment product is listed in the version column as `ND`.

The product in the version column indicates the product that you used to create the profile, not the type of profile that you installed. For example, if you use the WebSphere Application Server, Network Deployment product to install a profile type of application server, the version column indicates `ND`.

Discovery protocol:

Specifies the protocol that servers use to discover the presence of other servers on this node.

The possible protocol options follow:
UDP   User Datagram Protocol (UDP)
TCP   Transmission Control Protocol (TCP)

Status:

Indicates that the node is either synchronized, not synchronized, unknown, or not applicable.

Table 2. Node status.   Shows whether node changes are synchronized.

	Synchronized	The configuration files on this node are synchronized with the deployment manager.
	Not synchronized	The configuration files on this node are not synchronized with the deployment manager and are out-of-date. Perform a synchronize operation to get the latest configuration changes on the node.
	Unknown	The state of the configuration file cannot be determined because the node agent cannot be reached for this node.
	Not applicable	The status column is not applicable for this node because the node is an unmanaged node.

Node settings:

Use this page to view or change the configuration or topology settings for either a managed node instance or an unmanaged node instance.

A managed node is a node with an Application Server and a node agent that belongs to a cell. An unmanaged node is a node defined in the cell topology that does not have a node agent running to manage the process. Unmanaged nodes are typically used to manage web servers.

To view this administrative console page, click **System administration** > **Nodes** > *node_name*.

Name:

Specifies a logical name for the node. The name must be unique within the cell.

A node name usually is identical to the host name for the computer. However, you choose the node name. You can make the node name some name other than the host name.

Data type   String

Short Name:

Specifies the name of a node. The name is 1-8 characters, alphanumeric or national language. It cannot start with a numeric.

The short name property is defined during installation and customization. However, you can change the short name using the **renameNode.sh** command.

Host Name:

Specifies the host name of the unmanaged node that is added to the configuration.

Date type   String
Default   None

Discovery Protocol:

Specifies the protocol that the node follows to retrieve information from a network. The Discovery protocol setting is only valid for managed nodes.

Select from one of these protocol options:
UDP   User Datagram Protocol (UDP)

TCP   Transmission Control Protocol (TCP)

Data type	String
Default	TCP
Range	Valid values are UDP or TCP.

UDP is faster than TCP, but TCP is more reliable than UDP because UDP does not guarantee delivery of datagrams to the destination. Between these two protocols, the default of TCP is recommended.

File permissions: Specifies the most lenient file permissions for the application files that the product extracts into the application destination location. A deployer can override the permissions by configuring the permissions at the application level. However, if the file permissions specified at the application level are more lenient than the ones specified at the node, the ones specified at the node are used. The File permissions setting is only valid for managed nodes.

Data type	String
Default	`755` , or `rwx-rx-rx`, for files that end in `.dll`, `.so`, `.a` and `.sl` if no value is set

Platform type:

Specifies the operating system on which the unmanaged node runs.

Valid options are:

Windows
AIX®
HP-UX
Solaris
Linux
OS/400®
z/OS

Add managed node settings

A managed node is a node with an application server and a node agent that belongs to a deployment manager cell. Use this page to add an application server node to a deployment manager cell.

To view this deployment manager administrative console page, click **System administration** > **Nodes** > **Add Node** > **Managed node** > **Next** .

Node connection: Specifies connection information for WebSphere Application Server.

- **Host**

  Specifies the host name or IP address of the node to add to the cell. A WebSphere Application Server instance must be running on this machine.

Data type	String
Default	None

- **JMX connector type**

  Specifies the Java Management Extensions (JMX) connectors that communicate with the WebSphere Application Server when you invoke a scripting process.

  Select from one of these JMX connector types:

Simple Object Access Protocol (SOAP)

Use when the Application Server connects to a SOAP server.
Remote Method Invocation (RMI)

Use when the Application Server connects to an RMI server.

- **JMX connector port**

  Specifies the port number of the JMX connector on the instance to add to the cell. The default SOAP connector port is 8880.

Date type	Integer
Default	8880

- **Application server user name**

  Specifies the administration user name that connects to the remote Application Server whose node is being added to the cell. The Application Server user name and password are used to connect to the Application Server and start the add node process at the Application Server. The Application Server user name and password settings always display. You must specify values for them if security is enabled at the Application Server. Otherwise, leave them blank.

- **Application server password**

  Specifies the password for the Application Server user name that you supply.

- **Deployment manager user name**

  Specifies the deployment manager administration user name that the Application Server uses when connecting to the deployment manager to add its node to the cell. The deployment manager user name and password settings display only if security is enabled at the deployment manager. The deployment manager user name and password are required if their settings display.

- **Deployment manager password**

  Specifies the password for the deployment manager user name that you supply.

Options: Select from the following settings to further specify characteristics when adding a managed node to a cell.

- **Include applications**

  Copies the applications installed on the remote instance into a cell. If the applications to copy have the same name as the applications that currently exist in the cell, the Application Server does not copy the applications.

- **Include buses**

  Specifies whether to move the bus configuration at the node to the deployment manager.

- **Starting port**

  Specifies the port numbers for the node agent process.

Use default	Specifies whether to use the default node agent port numbers. The topic *Port number settings in WebSphere Application Server versions* provides a list of all of the default port numbers.
Specify	Allows you to specify the starting port number in the Port number field. WebSphere Application Server administration assigns the port numbers in order from the starting port number. For example, if you specify 9950, the administration program configures the node agent ports as 9950, 9951, 9952, and so on.

- **Core Group**

Specifies the group to which you can add a cluster or node agent. By default, clusters or node agents are added to the DefaultCoreGroup group.

Select from one of the core groups if a list is displayed. The list displays if a core group in addition to the default core group exists.

- **Node group**

  Specifies the group to which you can add the node. By default, nodes are added to the DefaultNodeGroup group.

  Select from one of the node groups if a list is displayed. The list displays if a node group in addition to the default node group exists.

Recover managed node settings

Use this page to recover an existing managed node of a deployment manager cell. The node must be at the deployment manager level.

To view this deployment manager administrative console page, click **System administration** > **Nodes** > **Add Node** > **Recover an existing node** > **Next**.

Before you use this page to recover an existing node that is no longer functioning, create a new profile to replace the damaged node and give it the same profile and node names. Then, use this page to replace the damaged node in the cell with the new node.

The new node, or application server, that you want to use to replace the damaged one must be running.

Instead of using this page to recover a node, you can run the addNode command with the `-asExistingNode` option from a command line at the `bin` directory of the stopped application server profile. The name of the node must match the name of the node where you run addNode with the option. Other addNode options for node configuration are incompatible with the `-asExistingNode` option.

Host:

Specifies the host name or IP address of the node to add to the cell. The application server process must be running at the IP address identified by the host field.

The value can be an IP address, a domain name server (DNS) name that resolves to an IP address, or the word `localhost` if the application server is running on the same machine as the deployment manager.

Data type	String
Default	None

JMX connector type:

Specifies the Java Management Extensions (JMX) connectors that communicate with the product when you invoke a scripting process.

Select from one of these JMX connector types:

- **Simple Object Access Protocol (SOAP)**

  Use when the Application Server connects to a SOAP server.

- **Remote Method Invocation (RMI)**

  Use when the Application Server connects to an RMI server.

- **JSR160RMI**

  Use when the Application Server connects to an JSR 160 RMI server.

JMX connector port:

Specifies the port number of the JMX connector on the instance to add to the cell. The default SOAP connector port is 8880.

You can find the port number under **Ports** on the Configuration tab of the server setting page. Click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Ports**.

Date type	Integer
Default	8880

Application server user name:

Specifies the administration user name that connects to the remote Application Server whose node is being added to the cell.

The Application Server user name and password are used to connect to the Application Server and start the add node process at the Application Server. The Application Server user name and password settings always display. You must specify values for them if security is enabled at the Application Server. Otherwise, leave them blank.

Application server password:

Specifies the password for the Application Server user name that you supply.

Deployment manager user name:

Specifies the deployment manager administration user name that the Application Server uses when connecting to the deployment manager to add its node to the cell.

The deployment manager user name and password settings display only if security is enabled at the deployment manager. The deployment manager user name and password are required if their settings display.

Deployment manager password:

Specifies the password for the deployment manager user name that you supply.

Config URL:

Specifies the security settings that enable a remote application server to communicate with the deployment manager. The default is a properties file with encoded passwords.

To view this setting, security must be enabled.

Date type	String
Default	file:/${USER_INSTALL_ROOT}/properties/sas.client.props

Node installation properties

Use this page to view read-only installation properties for this node. These properties provide information about the capabilities of the node that are collected during product installation time, such as the operating system name, architecture and version, or WebSphere Application Server product levels that are installed on the node.

To view this administrative console page, click **System administration** > **Nodes** > *node_name* > **Node installation properties**.

Information about a node, such as operating system platform and product features, is maintained in the configuration repository in the form of properties. As product features are installed on a node, new property settings are added.

WebSphere Application Server system management uses the managed object metadata properties as follows:

- To display the node version in the administrative console
- To ensure that new configuration types or attributes are not created or set on older release nodes
- To ensure that new resource types are not created on old release nodes
- To ensure that new applications are not installed on old release nodes because the old run time cannot support the new applications

For detailed information about the following properties, see the Application Server application programming interface (API).

com.ibm.websphere.baseProductShortName:

The product short name for the WebSphere Application Server that is installed.

com.ibm.websphere.baseProductVersion:

The version of WebSphere Application Server that is installed.

com.ibm.websphere.nodeOperatingSystem:

The operating system platform on which the node runs.

com.ibm.websphere.nodeSysplexName:

The sysplex name on a z/OS operating system.

Installed SDK collection

Use this page to specify the default software development kit (SDK) for a node. This page lists the software development kits that are installed on the node. A node can have one default SDK. Servers on the node use the default SDK unless a server overrides the SDK selection and specifies a different SDK.

To view this administrative console page, click **System administration** > **Nodes** > *node_name* > **Installed SDKs**.

To specify a default SDK for the node, select an SDK from the list and click **Make Default**.

Name:

Specifies the name of an SDK that is installed on the node.

Version:

Specifies the version number of the SDK.

Location:

Specifies the path of the SDK installation.

Default:

Specifies whether the SDK is the current default SDK for the node. A `true` value indicates that the SDK is the default. A `false` value indicates that the SDK is not the default.

A node can have no more than one SDK with a `true` value. A node can have many software development kits with a `false` value.

Changing the node host names

After creating a profile or adding a node, the host name of the server or its ports might be incorrect. You can follow the examples to change the server host name using command line tools and the wsadmin scripting tool, and the host name of the server ports using the administrative console and command line tools.

Before you begin

Create a profile or add a node to a cell. Verify that the host name of the server and the server ports are correct.

About this task

If the host name of a server or its ports is incorrect, then you might experience problems such as errors when you attempt to stop a server. One example task shows how to correct the server host name through command line tools and the wsadmin scripting tool. The other example task shows how to correct the host name of the server ports using the administrative console and command line tools.

Procedure

- Correct the host name for an application server node, a node agent, or a deployment manager node using the wsadmin scripting tool and command line tools.

  1. Launch the wsadmin tool.

     Enter the following command:

     `wsadmin -lang jython`

  2. List the contents of the server configuration file.

     Enter the following line of code:

     `AdminConfig.list('ServerIndex')`

  3. In the output, find the ServerIndex object for the application server node, the node agent, or the deployment manager, similar to the following examples:

     Application server and node agent:

     `cells/isthmusCell16/nodes/isthmusNode06|serverindex.xml#ServerIndex_1`

     Deployment manager:

     `cells/isthmusCell16/nodes/isthmusCellManager06|serverindex.xml#ServerIndex_1`

  4. Modify the host name for the application server node, the node agent, or the deployment manager, similar to the following examples:

     Application server and node agent:

     Enter the following line of code:

     ```
     AdminConfig.modify('(cells/isthmusCell16/nodes/isthmusNode06|serverindex.xml
     #ServerIndex_1)', "[[hostName new_host_name]]")
     ```

     Deployment manager:

     Enter the following line of code:

     ```
     AdminConfig.modify('(cells/isthmusCell16/nodes/isthmusCellManager06|
     serverindex.xml#ServerIndex_1)', "[[hostName new_host_name]]")
     ```

     The commands are split on multiple lines for printing purposes.

  5. Modify the host name for the Daemon instance, similar to the following examples:

     Application server and node agent:

Enter the following line of code:

```
AdminTask.modifyNodeGroupProperty('DefaultNodeGroup', '[ -name was.WAS_DAEMON_protocol_iiop_daemon_listenIPAddress -value newHostname]')
```

Deployment manager:

Enter the following line of code:

```
AdminTask.modifyNodeGroupProperty('DefaultNodeGroup', '[ -name was.WAS_DAEMON_protocol_iiop_daemon_listenIPAddress -value newHostname]')
```

6. Verify that the host names are correct, similar to the following examples:

Application server and node agent:

Enter the following line of code:

```
AdminConfig.show('(cells/isthmusCell07/nodes/isthmusCellManager07|
serverindex.xml#ServerIndex_1)', 'hostName')
```

The response is:

```
'[hostName isthmus]'
```

Deployment manager:

Enter the following line of code:

```
AdminConfig.show('(cells/isthmusCell07/nodes/isthmusNode04|
serverindex.xml#ServerIndex_1)', 'hostName')
```

The response is:

```
'[hostName isthmus]'
```

The commands are split on multiple lines for printing purposes.

7. Save the configuration.

Enter the following line of code:

```
AdminConfig.save()
```

8. Type `exit` to end the wsadmin session.

9. If you changed the host names for the application server and node agent, update the node with the changes.

   a. Stop the node agent.

   Enter the following command:

```
stopNode -profileName AppSrv01
```

   b. Stop the application server.

   Enter the following command:

```
stopServer server1 -profileName AppSrv01
```

   c. Use the syncNode script found in each federated node's /bin directory to synchronize the changes from the master configuration in the node

   Deployment manager:

   Enter the following command:

```
syncNode <DMGR_HOST> <SOAP_PORT>
```

   d. Restart the node agent.

   Enter the following command:

```
startNode -profileName AppSrv01
```

   e. Restart the application server.

   Enter the following command:

```
startServer server1 -profileName AppSrv01
```

10. If you changed the host name for the deployment manager, restart the deployment manager to apply the changes.

    a. Stop the deployment manager(from the deployment manager's /bin directory)..

    Enter the following command:

```
stopManager -profileName DMgr01
```

    b. Start the deployment manager.

Enter the following command:

```
startManager -profileName DMgr01
```

- Correct the host names for the ports that an application server, node agent, or deployment manager opens.

  If you have to correct the host names of the server ports, then you can make the correction using command line tools and either the wsadmin scripting tool or the administrative console. You might have to correct the host names of multiple ports for a particular server. This example shows you how to correct the host names using the administrative console and command line tools.

  1. For the application server, select **Servers** > **Server Types** > **WebSphere application servers** > *application_server* > **Ports**. For the node agent, select **System administration** > **Node agents** > *node_agent* > **Ports**. For the deployment manager, select **System administration** > **Deployment manager** > **Ports**.

  2. Select a port whose host name needs changing.

  3. Change the host name in the **Host** field; Click **OK**.

  4. Continue selecting ports and changing host names until you correct each of the host names for the server ports.

  5. Save the changes to the master configuration.

  6. If you changed the host names for the application server and node agent, update the node with the changes.

     a. Stop the node agent.
        - Select **System administration** > **Node agents**.
        - Select the node agent that you want to stop.
        - Click **Stop**.

     b. Stop the application server.
        - Select **Servers** > **Server Types** > **WebSphere application servers**.
        - Select the server that you want to stop.
        - Click **Stop**.

     c. Synchronize the nodes.

        Enter the following command:

```
syncNode deployment_manager_host deployment_manager_port
```

     d. Restart the node agent.

        Enter the following command:

```
startNode -profileName AppSrv01
```

     e. Restart the application server.
        - Select **Servers** > **Server Types** > **WebSphere application servers**.
        - Select the server that you want to restart.
        - Click **Start**.

  7. If you changed the host name for the deployment manager, restart the deployment manager to apply the changes.

     a. Stop the deployment manager.
        - Select **System administration** > **Deployment manager**.
        - Click **Stop**.

     b. Start the deployment manager.

        Enter the following command:

```
startManager -profileName DMgr01
```

Results

You have changed the host name of the server, the host names of the server ports, or both.

What to do next

You can continue to administer the product by doing such tasks as managing nodes, node agents, and node groups.

Starting and stopping a node

You can start a node by using the startNode command. You can stop a node by using the stopNode command. Starting and stopping a node is applicable only if your profile, also known as a node, is added to a WebSphere Application Server WebSphere Application Server, Network Deployment domain or cell.

Before you begin

Before you can start and stop a node, you must federate the node into a cell.

Start the deployment manager and add the node as a managed node of the deployment manager. Adding the managed node federates the node.

About this task

Start or stop a node as need in administering your WebSphere Application Server, Network Deployment environment. Before your environment can service requests, you must have the deployment manager and node started, and typically an HTTP server.

Procedure

- Start a node.

  Use one of these methods to start a node:

  - Use the `startNode` command.
- Stop a node.

  Use one of these methods to stop a node:

  - Use the stopNode command:

    `stopNode`

  - Use the deployment manager administrative console.

    To use the deployment manager administrative console to stop a node:
    1. Start the deployment manager profile that manages your node.
    2. Start the administrative console for the deployment manager.
    3. Expand **System Administration** and click **Node Agents**.
    4. Select the check box for the node that you want to stop.
    5. Click **Stop**.

Results

You have started and stopped a node.

What to do next

You can deploy applications, create a cluster, and generally administer your WebSphere Application Server, Network Deployment environment.

Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root
>Refers to the top directory for a WebSphere Application Server node.
>
>The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.
>
>The default varies based on node type. Common defaults are `configuration_root`/`AppServer` and `configuration_root`/`DeploymentManager`.

configuration_root
>Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.
>
>The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.
>
>The default is `/wasv8config/`*`cell_name`*`/`*`node_name`*.

plug-ins_root
>Refers to the installation root directory for Web Server Plug-ins.

profile_root
>Refers to the home directory for a particular instantiated WebSphere Application Server profile.
>
>Corresponding product variables are *server.root* and *user.install.root.*
>
>In general, this is the same as `app_server_root`/`profiles`/*`profile_name`*. On z/OS, this will always be `app_server_root`/`profiles`/`default` because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root
>Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.
>
>The corresponding product variable is *smpe.install.root*.
>
>The default is `/usr/lpp/zWebSphere/V8R0`.

Viewing, configuring, creating, and deleting node groups

This task discusses how to create and manage node groups.

Before you begin

Read about Nodes groups if you are unfamiliar with them.

About this task

Your WebSphere Application Server environment has a default node group. However, if you need additional node groups to manage your Application Server environment, you can create and configure additional node groups. You can delete a node group as long as it is not a default node group.

Procedure

- View and configure node groups.
  1. Click **System Administration > Node groups** in the console navigation tree.
  2. To view additional information about a particular node group or to further configure a node group, click on the node group name under **Name**.
- Create a node group.
  1. Click **System Administration > Node groups** in the console navigation tree.

2. Click **New**.

3. Specify the node group name and description.

The node group is added to the WebSphere Application Server environment . The name of the node group appears in the name column of the Node group page.

You can now add nodes to the node group. See "Adding, managing, and removing nodes" on page 8 and "Viewing, adding, and deleting node group members" on page 29 for information on how to add the nodes.

- Delete a node group if the node group is not the default node group.
    1. If the node group contains members, delete the members:
        a. Click **System Administration > Node groups** in the console navigation tree.
        b. Under **Name**, click the node group whose members you want to delete.
        c. Click **Node group members**.
        d. Select all the node group members.
        e. Click **Remove**.
    2. Click **System Administration > Node groups**.
    3. Select an empty node group.
    4. Click delete.

Node group collection

Use this page to manage node groups. A node group is a collection of WebSphere Application Server nodes. A node group defines a boundary for server cluster formation.

Nodes that are organized into a node group should be enough alike in terms of installed software, available resources, and configuration to enable servers on those nodes to host the same applications as part of a server cluster. The deployment manager does no validation to guarantee that nodes in a given node group have anything in common.

Node groups are optional and are established at the discretion of the WebSphere administrator. However, a node must be a member of a node group. Initially, all Application Server nodes are members of the default node group. The default node group is DefaultNodeGroup.

A node can be a member of more than one node group.

An Application Server node must be a member of a sysplex node group. Nodes in the same sysplex must be in the same sysplex node group. A node can only be in one sysplex node group. Sysplex node groups are special node groups that the system manages.

Nodes on distributed platforms and the IBM i platform cannot be members of a node group that contains a node on a z/OS platform. However, nodes on distributed platforms and nodes on the IBM i platform can be members of the same node group.

To delete a node group, the node group must be empty. The default node group cannot be deleted.

To view this administrative console page, click **System administration** > **Node groups**.

Name:

Specifies a name for a node group that is unique within the cell.

Members:

Specifies the number of members or nodes in the node group.

Description:

Specifies a description that you define for the node group.

Node group settings:

Use this page to view or change the configuration or topology settings for a node group instance.

To view this administrative console page, click **System administration** > **Node groups** > *node_group_name*.

Name:

Specifies a logical name for the node group. The name must be unique within the cell. The name can start with a number.

Data type	String
Maximum length	64 characters

Short name:

Specifies the name of a node. The name must contain 1-8 characters, which are either alphanumeric or national language. It cannot start with a number.

On the z/OS system the short name property is:
- Read-only
- Used only by sysplex node groups
- Defined during installation and customization

Sysplex:

Specifies the name of a node. The name is eight characters, alphanumeric or national language. It cannot start with a numeric. It is used only by sysplex node groups on the z/OS platform. It is defined during installation and customization on z/OS platforms only.

The Sysplex property is read only.

Members:

Specifies the number of nodes within the node group.

Data type	Integer

Description:

Specifies the description that you define for the node group. The description has no specific maximum length.

Viewing, adding, and deleting node group members

Use this topic to manage the nodes in your node groups by viewing, adding or deleting the nodes in a node group.

Before you begin

Read about Nodes groups if you are unfamiliar with them.

About this task

Make the nodes that you organize into a node group enough alike in terms of installed software, available resources, and configuration to enable servers on those nodes to host the same applications as part of a server cluster.

Node group membership must adhere to the following rules:
- A node in a node group must be a managed node.
- A managed node must be a member of at least one node group. Initially, all WebSphere Application Server nodes are members of the default node group named DefaultNodeGroup.
- If the node is on the z/OS platform, the node must be a member of a sysplex node group. The node can also be a member of other node groups that are not sysplex node groups.
- Nodes on distributed platforms and IBM i platforms cannot be members of a node group that contains a node on a z/OS platform.
- Nodes that are in different sysplexes must be members of different node groups. Nodes that are in the same sysplex must belong to the same node group.

Procedure

- View node groups members.
  1. Click **System Administration > Node groups >** *node group name* **> Nodes > Node group members** in the console navigation tree.
  2. To view additional information about a particular node group member for this node group, click on the node group member name under **Name**.
- Add a node to a node group.
  1. Click **System Administration > Node groups >** *node group name* **> Nodes > Node group members** in the console navigation tree.
  2. Click **Add**.
  3. Select the node from a list. The node group member name is the node name.

  The node group member is added to the node group specified on the breadcrumb trail. The name of the node group member appears in the name column of the Node group member page. You can add additional nodes of similar characteristics to the node group by repeating the steps for adding a node to a node group.

  If the node you add does not satisfy the node group membership rules for the target node group, the add node operation fails with an error message.
- Remove a node from a node group.
  1. Click **System Administration > Node groups >** *node group name* **> Nodes > Node group members** in the console navigation tree.
  2. Select the box next to each node group member that you want to remove from the node group.
  3. Click **Remove**.

  Each node group member that you selected is removed from the node group specified on the breadcrumb trail.

Node group member collection

Use this page to manage node groups members. A node group member is a WebSphere Application Server node.

To view this administrative console page, click **System administration** > **Node groups** > *node_group_name* > **Node group members**.

Click **Add** to add node members to the node group. Click **Remove** to remove node members from the node group.

Name:

Specifies the name of a node group member.

Node group member settings:

Use this page to view or change the configuration or topology settings for a node group member.

To view this administrative console page, click **System administration** > **Node groups** > *node_group_name* > **Node group members** > *node_group_member_name*.

Name:

Specifies a logical name for the node group member. A node group member is a node. The name must be unique within the cell.

A node group member name usually is identical to the host name for the computer.

Data type	String
Maximum length	64 characters

The name must contain alphanumeric or national language characters and can start with a number.

Managing node agents

Node agents are administrative agents that represent a node to your system and manage the servers on that node. Node agents monitor application servers on a host system and route administrative requests to servers.

Before you begin

Before you can manage a node agent, you must install the WebSphere Application Server, Network Deployment product.

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

About this task

A node agent is a server that is created automatically when a node is added to a cell. A node agent runs on every host computer system that participates in the WebSphere Application Server, Network Deployment product. You can view information about a node agent, stop and start the processing of a node agent, stop and restart application servers on the node that is managed by the node agent, and so on.

A node agent is purely an administrative agent and is not involved in application serving functions. A node agent also hosts other important administrative functions, such as file transfer services, configuration synchronization, and performance monitoring.

You can manage nodes through the wsadmin scripting tool, through the Java application programming interfaces (APIs), or through the administrative console. Perform the following tasks to manage nodes on an application server through the administrative console.

Procedure

- View information about a node agent. Click **System Administration > Node agents** in the console navigation tree. To view additional information about a particular node agent or to further configure a node agent, click the node agent name under **Name**.
- Stop and then restart all of the application servers on the node that is managed by the node agent. On the Node agents page, select the check box beside the node agent that manages the node with servers that you want to restart, and click **Restart all servers on node**.

  Clicking **Restart all servers on node** also stops and then restarts the node agent. Servers that were stopped when you clicked **Restart all Servers on Node** remain stopped.

  Tip: The node agent for the node must be processing to restart application servers on the node.
- Stop the processing of a node agent. On the Node agents page, select the check box beside the node agent that you want to stop processing; click **Stop**.

Results

Depending on the steps that you completed, you have viewed information about a node agent, stopped and started the processing of a node agent, and stopped and restarted application servers on the node that is managed by the node agent.

What to do next

You can administer other aspects of the WebSphere Application Server, Network Deployment environment, such as the deployment manager, nodes, and cells.

Node agent collection

Use this page to view information about node agents. Node agents are administrative agents that monitor application servers on a host system and route administrative requests to servers. A node agent is the running server that represents a node in a Network Deployment environment.

To view this administrative console page, click **System Administration** > **Node agents**.

You must initially start a node agent outside the administrative console. For information on how to initially start a node agent, see the addNode command and the startNode command.

Name:

Specifies a logical name for the node agent server.

Node:

Specifies a name for the node. The node name is unique within the cell.

A node name usually is identical to the host name for the computer. That is, a node usually corresponds to a physical computer system with a distinct IP host address.

However, the node name is a purely logical name for a group of servers. You can name the node anything you please. The node name does not have to be the host name.

Version:

Specifies the product version of the node.

The product version is the version of a WebSphere Application Server node agent and Application Servers that run on the node.

Host Name:

Specifies the IP address, the full domain name system (DNS) host name with a domain name suffix, or the short DNS host name for the node agent.

Status:

Indicates whether the node agent server is started or stopped.

Table 3. Node agent server status. Shows whether the node agent is running.

	Started	The node agent is running.
	Stopped	The node agent is not running.

Node agent server settings:

Use this page to view information about and to configure a node agent. A node agent coordinates administrative requests and event notifications among servers on a machine. A node agent is the running server that represents a node in a WebSphere Application Server, Network Deployment environment.

To view this administrative console page, click **System administration** > **Node agents** > *node_agent*.

A node agent must be started on each node for the deployment manager node to collect and control servers that are configured on that node. If you use configuration synchronization support, a node agent coordinates with the deployment manager server to synchronize the configuration data of the node with the master copy that the deployment manager manages.

You must initially start a node agent outside the administrative console. For information on how to initially start a node agent, see the addNode command and the startNode command.

The Runtime tab displays only when a node agent runs.

Name:

Specifies a logical name for the node agent server.

Data type	String

Node:

Specifies the name of the node for the node agent server.

Data type	String

Short name:

Specifies the short name of the node agent server.

The server short name must be unique within a cell. The short name identifies the server to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, System Authorization Facility (SAF) (for example, Resource Access Control Facility (RACF®)), started task control, and others.

The name can be 1-8 alphanumeric or national language characters and cannot start with a numeric.

The system assigns a cell-unique, default short name.

Unique ID:

Specifies the unique ID of this node agent server.

The unique ID property is read only. The system automatically generates the value.

Run in 64 bit JVM mode:

Specifies whether to run the node agent in 64-bit addressing mode to obtain more than the virtual memory that is available to the node agent when the node agent runs in 31-bit addressing mode, which is 2 gigabytes (GB).

Note: You should eventually convert all of your servers to run in 64-bit addressing mode because support for running servers in 31-bit addressing mode is deprecated.

Default	true (checked)

Start components as needed: Select this property if you want the server components started as they are needed for applications that run on this server.

When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

gotcha: If you are running other WebSphere products on top of the this product, make sure that those other products support this functionality before you select this property.

Process ID:

Specifies a string identifying the process.

Data type	String

Cell name:

Specifies the name of the cell for the node agent server.

Data type	String
Default	*host_name*Network

Node name:

Specifies the name of the node for the node agent server.

Data type	String

State:

Indicates whether the node agent server is started or stopped.

Data type	String
Default	Started

Current heap size:

Indicates the current heap size that the server Java virtual machine (JVM) uses, in megabytes.

Maximum heap size:

Indicates the maximum heap size that the server JVM can use, in megabytes.

Configuring remote file services

Configuration data for the WebSphere Application Server product resides in files. Two services help you reconfigure and otherwise manage these files: the file transfer service and file synchronization service.

About this task

By default, the file transfer service is always configured and enabled at a node agent, so you do not need to take additional steps to configure this service. However, you might need to configure the file synchronization service.

Procedure

1. Go to the File Synchronization Service page. Click **System Administration** > **Node agents** in the console navigation tree. Then, click the node agent for which you want to configure a synchronization server and click **File synchronization service**.

2. On the File Synchronization Service page, customize the service that helps make configuration data consistent across a cell by moving updated configuration files from the deployment manager to the node. Change the values for properties on the File Synchronization Service page. The file synchronization service is always started, but you can control how it runs by changing the values.

3. By default if node synchronization fails five times consecutively, automatic synchronization is disabled. To keep automatic synchronization enabled, specify a custom property on the Java virtual machine (JVM) of node agent using the administrative console.

   a. Click **System Administration** > **Node Agents** > *nodeagent*. Click on the node agent for which you want to configure the optional custom property.

   b. Under Server Infrastructure, expand **Java and Process Management** then click **Process Definition** > **Java Virtual Machine** > **Custom Properties**.

   c. Click **New**.

   d. Specify a name and value for the custom property.

      The com.ibm.websphere.management.sync.allowfailure custom property:

      To keep automatic synchronization enabled, specify this custom property with a value of `true`.

Property	com.ibm.websphere.management.sync.allowfailure
Data type	Boolean
Default	False

4. Optionally add a custom property for file synchronization on the file synchronization service page.

   a. Click **System Administration** > **Node Agents** > *nodeagent*. Click on the node agent for which you want to configure the optional custom property.

b. Under Additional Properties, click **File synchronization service**

c. Under Additional Properties, click **Custom Properties**

d. Click **New**.

e. Specify a name and value for the custom property.

The com.ibm.websphere.management.application.expand.wto custom property:

Specify this custom property with a value of `true` if you want to display Enterprise Archive (EAR) file expansion errors on thez/OS operating system console. When the errors display, the operator can take appropriate corrective action for the failure.

Property	com.ibm.websphere.management.application.expand.wto
Data type	Boolean
Default	False

File transfer service settings

Use this page to configure the service that transfers files from the deployment manager to individual remote nodes.

To view this administrative console page, click **System Administration > Node agents >** *node_agent_name* **> File transfer service**.

Enable service at server startup:

Specifies whether the server attempts to start the specified service. Some services are always enabled and disregard this property if set. This setting is enabled by default.

Data type	Boolean
Default	true

Retries count:

Specifies the number of times you want the file transfer service to retry sending or receiving a file after a communication failure occurs.

Data type	Integer
Default	3
	If the retries count setting is blank, the file transfer service sets the default to 3. If the retries count setting is 0, the file transfer service does not retry. The default is the recommended value.

Retry wait time:

Specifies the number of seconds that the file transfer service waits before it retries a failed file transfer.

Data type	Integer
Default	10
	If the retry wait time setting is blank, the code sets the default to 10. If the retry wait time setting is 0, the file transfer service does not wait between retries. The default is the recommended value.

File synchronization service settings

Use this page to specify that a file set on one node matches that on the central deployment manager node and to ensure consistent configuration data across a cell.

You can synchronize files on individual nodes or throughout your system.

Note: If your installation includes mixed release cells, a large numbers of nodes, and runs a large number of applications, you might want to use the **Generic JVM Arguments** field, on the Java Virtual Machine Settings page of the administrative console, to enable the hot restart sync feature of the synchronization service. This feature indicates to the synchronization service that the installation is running in an environment where configuration updates are not made when the deployment manager is not active. Therefore, the service does not have to perform a complete repository comparison when the deployment manager or node agent servers restart. For more information, see the Generic JVM arguments in the Java virtual machine settings documentation.

Important: Do not routinely disable the synchronization process as various portions of the application server run time depend on synchronization. For example, the security run time depends on node synchronization to propagate updated certificates during automated replacement processes. Also, the security runtime also depends on it for Lightweight Third Party Authentication (LTPA) key changes. Other portions of the run time are dependent on synchronization. However, a complete list is not available. If you *permanently* disable synchronization, nodes might not be synchronized and an outage will result. The only exception is the configuration save operation. To avoid synchronizing the configuration when the configuration repository is being updated by a save operation, it might be beneficial to *temporarily* disable the synchronization process, save the configuration, and then re-enable the synchronization process. This process ensures that changes are fully committed to the configuration repository before the node synchronization.

To view this administrative console page, click **System administration** > **Node agents** > *node_agent_name* > **File synchronization service**.

Enable service at server startup:

Specifies whether the server attempts to start the file synchronization service. This setting does not cause a file synchronization operation to start. This setting is enabled by default.

Important: Disabling synchronization is not recommended. Various portions of the run time have dependencies on synchronization being enabled. For example, in a multiple-server product, the security environment depends on node synchronization to propagate updated certificates during automated replacement and for LTPA key changes. If synchronization is disabled, the nodes might get out of synchronization, resulting in an outage.

Data type	Boolean
Default	true

Synchronization interval:

Specifies the number of minutes that elapse between synchronizations. Increase the time interval to synchronize files less often. Decrease the time interval to synchronize files more often.

Data type	Integer
Units	Minutes
Default	10

The minimum value that the application server uses is 1. If you specify a value of 0, the application server ignores the value and uses the default of 1.

Important: You can change the node synchronization interval to a greater number of minutes. However, before you make the change, carefully consider the impact because some functions critically depend on node synchronization. Before changing the value to any value other than the default value, it is advisable that you complete careful, large-scale testing in your environment.

Automatic synchronization:

Specifies whether to synchronize files automatically after a designated interval. When this setting is enabled, the node agent automatically contacts the deployment manager every synchronization interval to attempt to synchronize the node's configuration repository with the master repository owned by the deployment manager.

If the Automatic synchronization setting is enabled, the node agent attempts file synchronization when it establishes contact with the deployment manager. The node agent waits the synchronization interval before it attempts the next synchronization.

Remove the check mark from the check box if you want to control when files are sent to the node.

Data type	Boolean
Default	true

Startup synchronization:

Specifies whether the node agent attempts to synchronize the node configuration with the latest configurations in the master repository prior to starting an application server.

The default is to not synchronize files prior to starting an application server. Enabling the setting ensures that the node agent has the latest configuration but increases the amount of time it takes to start the application server.

Note that this setting has no effect on the startServer command. The startServer command launches a server directly and does not use the node agent.

Data type	Boolean
Default	false

Exclusions:

Specifies files or patterns that should not be part of the synchronization of configuration data. Files in this list are not copied from the master configuration repository to the node, and are not deleted from the repository at the node.

The default is to have no files specified.

To specify a file, use a complete name or a name with a leading or trailing asterisk (*) for a wildcard. For example:

`cells/`*`cell name`*`/nodes/`*`node name`*`/file name`	Excludes this specific file

`*/file name`	Excludes files named *file name* in any context
`dirname/*`	Excludes the subtree under `dirname`

Press **Enter** at the end of each entry. Each file name appears on a separate line.

Since these strings represent logical document locations and not actual file paths, only forward slashes are needed no matter the platform.

Changes to the exclusion list are picked up when the node agent is restarted.

Data type	String
Units	File names or patterns

Working with cells - groups of nodes

When you create a deployment manager profile, a cell is created. A cell provides a way to group one or more nodes of the product. You can configure the cell. After configuring a cell, you probably do not need to configure the cell again.

Configuring cells

This topic describes how to change the cell protocol information, define custom properties for the cell, and add additional nodes.

Before you begin

Before you can configure cells, you must install the WebSphere Application Server, Network Deployment product.

About this task

When you create a deployment manager profile, a cell is created. A cell provides a way to group one or more nodes of your WebSphere Application Server, Network Deployment product. You define the nodes that make up a cell, according to the specific criteria that make sense in your organizational environment. You probably do not need to configure the cell again.

Administrative configuration data is stored in XML files. A cell retains master configuration files for each server in every node in the cell. Each node and server also have their own local configuration files. Changes to a local node or to a server configuration file are temporary, if the server belongs to the cell. While in effect, local changes override cell configurations. Changes to the master server and master node configuration files made at the cell level replace any temporary changes made at the node when the cell configuration documents are synchronized to the nodes. Synchronization occurs at designated events, such as when a server starts.

To view information about and to manage a cell, use the settings page for a cell.

Procedure

1. Access the settings page for a cell. Click **System Administration > Cell** from the navigation tree of the administrative console.
2. If the protocol that the cell uses to retrieve information from a network is not appropriate for your system, select the appropriate protocol. By default, a cell uses Transmission Control Protocol (TCP). If you want the cell to use User Datagram Protocol, select **UDP** from the list for **Cell discovery protocol** on the settings page for the cell. It is unlikely that you need to change the cell protocol configuration from TCP.

3. Click **Custom Properties** and define any name-value pairs that your deployment manager needs.
   a. Click **New**.
   b. Specify a name and value for the custom property.
4. When you install the WebSphere Application Server, Network Deployment product, a node may have been added to the cell. You can add additional nodes on the Node page. Click **Nodes** to access the Node page, which you use to manage nodes.

   Both Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) are now supported by WebSphere Application Server, but there are restrictions that apply to using both IPv4 and IPv6 in the same cell. Note that when you add a node to a cell, the format in which you specify the host name is based on the version of IP the node will be using. For further information, read the topic on IP version considerations for cells.

Results

Depending on which steps you performed, you changed the cell protocol information, defined custom properties for the cell, and added additional nodes.

What to do next

You can continue to administer your WebSphere Application Server, Network Deployment product by doing such tasks as managing nodes, node agents, and node groups.

Cell custom properties

You can configure name-value pairs of data, where the name is a property key and the value is a string value that you can use to set configuration properties for a cell. Defining a new custom property for a cell enables you to configure a setting beyond that which is available in the administrative console. This topic lists custom properties that are available to configure a cell.

To specify a custom property for a cell:

1. In the administrative console, click **System administration** > **Cell** > **Custom properties**.
2. On the Custom properties page, click **New**.
3. On the settings page, enter the name of the custom property that you want to configure in the **Name** field and the value that you want to set it to in the **Value** field.
4. Click **Apply** or **OK**, and then click **Save** to save your configuration changes.
5. Restart the server on which the cell resides.

The following custom properties are provided with the product:
- "com.ibm.websphere.management.launcher.options" on page 41
- "com.ibm.websphere.process.terminator.deletepid" on page 41
- "enableAdminAuthorizationCache" on page 41
- "IBM_CLUSTER_CALLBACK_TIMEOUT" on page 41
- "IBM_CLUSTER_CUSTOM_ADVISOR_THREAD_POOL_SIZE" on page 41
- "IBM_CLUSTER_ENABLE_ACS_DELAY_POSTING" on page 42
- "IBM_CLUSTER_ENABLE_CAR_DELAY_POSTING" on page 42
- "IBM_CLUSTER_ENABLE_PRELOAD" on page 42
- "IBM_CLUSTER_ENABLE_NON_DEFAULT_COOKIE_NAMES" on page 42
- "IBM_CLUSTER_PURGE_NOTIFICATIONS" on page 43
- "IBM_CLUSTER_RIPPLESTART_NOTIFICATION_TIMEOUT" on page 43
- "IBM_CLUSTER_USE_LEGACY_COMPRESSOR" on page 43
- "IBM_CLUSTER_WBI_SUPPORT" on page 44

com.ibm.websphere.management.launcher.options: Specify a value of `displayServerInFront` to display the name of the cell, node, and server in front of the output for the ps -ef command. Use of this property is intended to help you identify the process ID of a server. The property has no impact on the server process.

Property	com.ibm.websphere.management.launcher.options
Data type	String
Default	none

com.ibm.websphere.process.terminator.deletepid: By default, the autoRestart attribute for the server monitoring policy is set to `false` because the Automatic Restart Management (ARM) is typically used to manage automatic restarts of the application servers. The autoRestart attribute for the server monitoring policy is defined in the `server.xml` file.

If set the autoRestart attribute to `true` because you do not want to use ARM to manage your server automatic restarts, you must also add the com.ibm.websphere.process.terminator.deletepid cell custom property to your cell configuration and set this custom property to `true`. If you change only the setting on the autoRestart attribute, the servers in the cell keep restarting in response to a stopImmediate command.

Property	com.ibm.websphere.process.terminator.deletepid
Data type	Boolean
Default	false

enableAdminAuthorizationCache: By default, caching for authorization is disabled. Caching can be enabled by setting this property value to `true`.

```
{
wsadmin> set c [$AdminConfig list Cell]
wsadmin> $AdminConfig create Property $c {{name enableAdminAuthorizationCache}{value true}}
}
```

When this property is set, there should be a lower number of RACF authorizations.

IBM_CLUSTER_CALLBACK_TIMEOUT: Specifies, in milliseconds, the length of time the node agent waits for cluster data to be gathered after a client submits the first request for that cluster. You do not need to specify a value for this property if the IBM_CLUSTER_ENABLE_PRELOAD custom property is set to `true`, because in that situation, the data is preloaded during the server startup process.

If the amount of time specified for this property is not sufficient for the amount of cluster data that needs to be gathered, NO_IMPLEMENT: No Cluster Data Available exceptions might still occur the first few times a client sends requests to a cluster. Specifying an appropriate length of time for this property or specifying a value of `0`, which eliminates the timeouts completely prevents the NO_IMPLEMENT: No Cluster Data Available exceptions from occurring because the cluster data is gathered within the specified length of time.

Property	IBM_CLUSTER_CALLBACK_TIMEOUT
Data type	Integer
Default	180000, which is equivalent to 3 minutes

IBM_CLUSTER_CUSTOM_ADVISOR_THREAD_POOL_SIZE: Specifies the number of threads in the thread pool that are used to run the custom advisors.

There is one thread pool that is used to run all the custom advisors configured on a proxy server. If there are more custom advisors configured than there are threads in the pool, and the polling interval and other

circumstances are such that more custom advisors should be running at the same time than there are threads in the pool, some custom advisors get queued up and run as soon as a thread becomes available.

Property	IBM_CLUSTER_CUSTOM_ADVISOR_THREAD_POOL_SIZE
Data type	Integer
Range	1 - 50
Default	5

IBM_CLUSTER_ENABLE_ACS_DELAY_POSTING: Specifies whether publishing of updates to the ActiveClusterSet is delayed. Enabling this custom property provides a performance improvement in large SIBus topologies and configured destinations, which results in a reduction of Messaging Engine Start and Stop times.

When this property is set to `true`, publishing of updates is delayed.

When this property is set to `false`, updates are published immediately.

Property	IBM_CLUSTER_ENABLE_ACS_DELAY_POSTING
Data type	Boolean
Default	false

IBM_CLUSTER_ENABLE_CAR_DELAY_POSTING: Specifies whether publishing of updates to the ClusterDescription is delayed. Enabling this custom property provides a performance improvement in large service integration bus (SIBus) topologies and configured destinations, which results in a reduction of messaging engine start and stop times.

When this property is set to `true`, publishing of updates is delayed.

When this property is set to `false`, updates are published immediately.

Property	IBM_CLUSTER_ENABLE_CAR_DELAY_POSTING
Data type	Boolean
Default	false

IBM_CLUSTER_ENABLE_PRELOAD: Specifies whether the preload logic runs at server startup on the node agent. Without preload, a node agent only loads the data for a cluster after the node agent receives the first request for that cluster.

When this property is set to `true`, cluster data is loaded on the node agent at startup, and does not have to be created and propagated at run time.

When this property is set to `false`, the cluster data is created and propagated the first time there is a request to a cluster, which sometimes causes `NO_IMPLEMENT: No Cluster Data Available` exceptions the first few times a client sends requests to a cluster.

Property	IBM_CLUSTER_ENABLE_PRELOAD
Data type	Boolean
Default	false

IBM_CLUSTER_ENABLE_NON_DEFAULT_COOKIE_NAMES: Specify this custom property with a value of `true` to override the cookie name at the server, application, or module level so that the proxy server can

maintain session affinity with multiple applications on different clusters. The proxy server can maintain the session affinity since it can recognize session cookies other than JSESSIONID.

If you change any session management configuration, wait until all members of the cluster have been updated with the new configuration before doing a ripple start of the cluster. Otherwise, session failover might not work.

If you change the application or module session management configuration, wait until all members of the cluster have been updated with the new configuration before stopping and then restarting the application. Otherwise, session failover might not work.

Property	IBM_CLUSTER_ENABLE_NON_DEFAULT_COOKIE_NAMES
Data type	Boolean
Default	false

IBM_CLUSTER_PURGE_NOTIFICATIONS: Specifies whether references to identities are deleted when there are no ClusterObservers registered for notifications on those identities. When an identity is deleted, all related posts on the BulletinBoard about that identity are cleared. Setting this property to `true` enables the references to identities that do not have any ClusterObservers registered on them to be deleted.

Typically there are many destinations defined in a Service Integration Bus (SIB) hierarchical scenario. If this property is set to `false`, and a product, such as the WebSphere Process Server is installed on top of WebSphere Application Server, the Workload Management (WLM) does not properly allow data stored on the Bulletin Board to be garbage collected. This situation can cause a slow memory leak if certain tasks, such as the installing and uninstalling applications, are repeated without restarting the process. If the process is restarted all posts associated with that server are automatically removed, thus preventing the memory leaks.

Property	IBM_CLUSTER_PURGE_NOTIFICATIONS
Data type	Boolean
Default	false

IBM_CLUSTER_RIPPLESTART_NOTIFICATION_TIMEOUT: Specify a value, in milliseconds, to indicate the amount of time the ripplestart function waits for processes to shut down before restarting them. If you attempt a ripplestart and the processes have not shutdown before the start operation begins, one or more of the processes will not restart.

Property	IBM_CLUSTER_RIPPLESTART_NOTIFICATION_TIMEOUT
Data type	Integer
Default	300000 milliseconds (5 minutes)

IBM_CLUSTER_USE_LEGACY_COMPRESSOR: Starting with Version 6.1.0.37, Workload management (WLM) uses a new procedure for compressing and extracting data being sent between processes. This procedure reduces compression overhead in large WebSphere Process Server and service integration bus topologies, that have 2,000 or more destinations. This procedure also prevents compressed data from being lost if a series of bytes to be compressed could not actually be compressed because every byte is unique. This situation could potentially cause a loss of data because of how WLM previously handled the uncompressed data.

This optimized compression and decompression procedure is appropriate for most environments. However, if this procedure causes problems in your environment, add the IBM_CLUSTER_USE_LEGACY_COMPRESSOR custom property to your cell settings, and set it to `true`.

When you set this property to `true`, WLM handles data compression and decompression the same way that it did before Version 6.1.0.37 or later was installed.

If you add this custom property to your cell settings, you must synchronize the nodes, and restart all the processes in the cell before this change goes into effect.

Property	IBM_CLUSTER_USE_LEGACY_COMPRESSOR
Data type	Boolean
Default	false

IBM_CLUSTER_WBI_SUPPORT: Specifies whether your system supports interaction with either WebSphere Business Integration Version 5.1 clients or WebSphere Process Server for Multiplatforms Version 5.0.2 clients. If WebSphere Business Integration Version 5.1 clients, or WebSphere Process Server for Multiplatforms Version 5.0.2 clients need to send data to your system you must set this property to `true`. When this property is not included in your cell configuration settings, or if it is set to `false`, there is an interoperability issue where the data stream being passed back and forth becomes corrupted, which results in the following exception:

```
Exception stack trace: javax.naming.NamingException:
Error during resolve.  Root exception is rg.omg.CORBA.NO_IMPLEMENT:Trace from server:
  server_name at host host_name
```

gotcha: This property can also be specified as an application server custom property. However, if this property is specified at both levels, the value specified for the property at the server level takes precedence over the value specified for this property at the cell level. To enable the property at the server level, in the administrative console click **Servers** > **Server Type** > **WebSphere application servers**, and then under Server Infrastructure, click **Administration** > **Custom properties**.

Property	IBM_CLUSTER_WBI_SUPPORT
Data type	Boolean
Default	false

Cell settings for deployment managers

Use this page to set the discovery protocol and address end point for an existing cell. A cell is a configuration concept, a way for an administrator to logically associate nodes according to whatever criteria make sense in the administrator's organizational environment.

To view this administrative console page, click **System administration** > **Cell**.

Name:

Specifies the name of the existing cell.

A cell name must be unique in any circumstance in which the product is running on the same physical machine or cluster of machines, such as a sysplex. Additionally, a cell name must be unique in any circumstance in which network connectivity between entities is required either between the cells or from a client that must communicate with each of the cells. Cell names must also be unique if their namespaces are federated. Otherwise, you might encounter symptoms such as a javax.naming.NameNotFoundException error, in which case, create uniquely named cells.

Short Name:

Specifies the short name of the cell. The name is 1-8 characters, alphanumeric or national language. It cannot start with a numeric.

The short name property is read only. It was defined during installation and customization.

Cell Discovery Protocol:

Specifies the protocol that the nodes use to contact and discover the deployment manager in the cell.

Select one of these protocol options:

UDP   User Datagram Protocol (UDP)
TCP   Transmission Control Protocol (TCP)

Default   TCP

Deleting the Internet Protocol Version 4 or the Internet Protocol Version 6 multicast port

This topic describes how to delete the Internet Protocol Version 4 (IPv4) or the Internet Protocol Version 6 (IPv6) for multicast ports so that the node agent runs more efficiently.

Before you begin

You must install the WebSphere Application Server, Network Deployment product before you can delete a multicast port.

About this task

To allow node installation to run out-of-the box, both IPv4 and IPv6 are initially defined in the node agent configuration. To make the node agent run more efficiently, delete the multicast port that the node is not using.

Procedure

1. Click **System administration** > **Node agents***node_agent* > **Ports**.
2. On the Ports page, delete a multicast port.
3. Click **Delete**.

Results

What to do next

You can continue to administer your WebSphere Application Server, Network Deployment product by doing such tasks as managing nodes, node agents, and node groups.

Working with deployment managers - centralized cell management

A deployment manager is an administration application that is created when you add a cell or deployment manager management profile to a Network Deployment product. With the deployment manager, you can administer multiple nodes.

Configuring deployment managers

Configure deployment managers for a single, central point of administrative control for all elements in a WebSphere Application Server distributed cell.

Before you begin

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

About this task

Deployment managers are administrative agents that provide a centralized management view for all nodes in a cell, as well as management of clusters and workload balancing of application servers across one or several nodes in some editions. Each cell contains one deployment manager.

WebSphere Application Server for z/OS uses workload management (WLM) as the primary vehicle for workload balancing.

A deployment manager hosts the administrative console.

The Java virtual machine (JVM) for the deployment manager runs in 64-bit addressing mode by default. The maximum amount of virtual memory available to each JVM on the z/OS operating system in 31-bit addressing mode is 2 gigabytes (GB). However, because of product requirements, the amount of virtual memory that is available to each JVM is somewhat less. To obtain more virtual memory for the deployment manager server, or any other server, run the server in 64-bit addressing mode. If you have applications that require large amounts of virtual memory, the applications might need to run on servers configured for 64-bit addressing mode.

Note: You should eventually convert all of your servers to run in 64-bit addressing mode because support for running servers in 31-bit addressing mode is deprecated.

When you create a deployment manager profile, a deployment manager is created. You can run the deployment manager with its default settings. However, you can change the deployment manager configuration settings, such as the ports that the process uses, custom services, logging and tracing settings, and so on. To view information about managing a deployment manager, use the settings page for a deployment manager.

Procedure

1. Click **System administration** > **Deployment manager** from the navigation tree of the administrative console to access the settings page for a deployment manager.
2. Configure the deployment manager by clicking a property, such as **Custom services**, and specifying settings.
3. If you specify the server short name as eight characters, follow the directions to convert the default seven-character short name to eight characters.
4. Optionally register or unregister the deployment manager with the job manager.

   A job manager allows you to submit administrative jobs asynchronously for deployment managers and for application servers registered to administrative agents. Click **System administration** > **Deployment manager**. Under Additional Properties, click **Job managers** > **Register/unregister with job manager**.

Results

You configured a deployment manager with options that you selected.

What to do next

You can continue to administer your product by doing such tasks as configuring cells and managing nodes, node agents, and node groups.

You can use the deployment manager Diagnostic Provider to test connectivity between the deployment manager and the node agents. Click **Troubleshooting** > **Diagnostic Provider** > **Tests** > **dmgr** > **DeploymentManagerDP** > **ping.-*** from the navigation of the administrative console. After running the diagnostic provider, click on the message text to get to the detail page that shows the node agent status. If the value field of the node agent is `j2ee.state.stopped`, the value usually means that the node agent is stopped. However, it can also mean that the deployment manager has lost network connectivity with the node agent. The deployment manager cannot distinguish between these two cases.

Deployment manager settings

Use this page to stop the deployment manager, and to link to other pages that you can use to define additional properties for the deployment manager. A deployment manager provides a single, central point of administrative control for all of the elements in the WebSphere Application Server distributed cell.

To view this administrative console page, click **System administration** > **Deployment manager**.

Name:

Specifies a logical name for the deployment manager. The name must be unique within the cell.

Data type	String

Short name:

Specifies the short name of the deployment manager server.

The server short name must be unique within a cell. The short name identifies the server to the native facilities of the operating system, such as workload manager (WLM), Automatic Restart Manager, System Authorization Facility (SAF) (for example, Resource Access Control Facility (RACF)), started task control, and others.

The name can be 1-8 alphanumeric or national language characters and cannot start with a numeric.

The system assigns a cell-unique, default short name.

Note: If you change a 7-character deployment manager short name to an 8-character short name, you must update the start command arguments for the servant to use the new 8-character short name. See Converting a 7-character server short name to 8 characters.

Data type	String

Unique ID:

Specifies the unique ID of this deployment manager server.

The unique ID property is read only. The system automatically generates the value.

Data type	String

Run in 64 bit JVM mode:

Specifies whether to run the deployment manager in 64-bit addressing mode to obtain more than the virtual memory that is available to the deployment manager when the deployment manager runs in 31-bit addressing mode, which is 2 gigabytes (GB).

Note: You should eventually convert all of your servers to run in 64-bit addressing mode because support for running servers in 31-bit addressing mode is deprecated.

Default	true (checked)

Start components as needed: Select this property if you want the server components started as they are needed for applications that run on this server.

When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

gotcha: If you are running other WebSphere products on top of the this product, make sure that those other products support this functionality before you select this property.

Process ID:

Specifies a string that identifies the process.

Data type	String
Default	None

Cell name:

Specifies the name of the cell for the deployment manager. The default is the name of the host computer on which the deployment manager is installed with `Cell##` appended, where *##* is a two-digit number.

Data type	String
Default	*host_name*Cell01

Node name:

Specifies the name of the node for the deployment manager. The default is the name of the host computer on which the deployment manager is installed with `CellManager##` appended, where *##* is a two-digit number.

Data type	String
Default	*host_name*CellManager01

State:

Indicates the state of the deployment manager. The state is *Started* when the deployment manager is running and *Stopped* when the deployment manager is not running.

Data type	String
Default	Started

Renaming deployment manager nodes

This topic describes how to rename a deployment manager node by modifying the `setupCmdLine` file for the node.

Before you begin

Make sure that you create a backup of the deployment manager configuration including its node configurations. For more information, see the documentation about the backupConfig command.

About this task

You can use the renameNode command to modify managed nodes. However, the rename command does not modify the node name information within the deployment manager configuration. If you use the renameNode command to modify a deployment manager node, the deployment manager does not restart. This task resolves the problem by modifying the `setupCmdLine` file so that it references the new node name.

Procedure

1. Start the deployment manager.
2. Start the wsadmin scripting tool within the deployment manager profile. For more information, see the documentation about starting the wsadmin scripting client.
3. Use the renameNode command in the interactive mode to rename the managed node.
   - Using Jacl:
     ```
     $AdminTask renameNode {-interactive}
     ```
   - Using Jython string:
     ```
     AdminTask.renameNode ('[-interactive]')
     ```
   - Using Jython list:
     ```
     AdminTask.renameNode (['-interactive'])
     ```
4. Save the configuration changes. For more information, see the documentation about saving configuration changes with the wsadmin tool.

   At this point, the renameNode command renames the managed node, but does not modify the deployment manager node configuration. Thus, the deployment manager does not restart.
5. Modify the *WAS_NODE* variable in the *profile_root*/*deployment_manager_name*/`bin/setupCmdLine` file so that it references the new managed node name.

Results

After you save the `setupCmdLine` file, the deployment manager starts successfully.

Starting and stopping the deployment manager

The deployment manager is an administration application that runs in a special application server, which is created when you install the WebSphere Application Server, Network Deployment product or when you create a management profile using the deployment manager profile template. With the deployment manager, you can administer multiple WebSphere Application Server nodes. This topic describes how you start and stop the deployment manager.

Before you begin

Before you can start or stop the deployment manager, you must first install the WebSphere Application Server WebSphere Application Server, Network Deployment product.

About this task

Start the deployment manager so that you can manage all the elements of the WebSphere Application Server cell. Stop the deployment manager as needed, such as when migrating to a new version of the WebSphere Application Server, Network Deployment product, when uninstalling the product, and so on.

Procedure

- Start the deployment manager.

  Use one of these methods to start a deployment manager:

  – Use the startManager command:

  `startManager`

  > For more information, see the startManager command topic in the *Administering applications and their environment* PDF.

- Stop the deployment manager.

  Use one of these methods to stop a deployment manager:

  – Use the stopManager command:

  `stopManager`

  > For more information, see the stopManager command topic in the *Administering applications and their environment* PDF.

  – Use the WebSphere Application Server, Network Deployment deployment manager administrative console.

  > To stop the deployment manager from the administrative console:
  >
  > 1. Click **System Administration** > **Deployment manager**.
  > 2. On the Configuration tab of the deployment manager settings, click **Stop**.

Results

You have started the deployment manager and have optionally stopped it.

What to do next

After you start a deployment manager, run the startNode command to start federated application server nodes of the deployment manager. After the deployment manager and nodes are running, you can administer servers and applications on the nodes.

After you stop a deployment manager, run the stopNode command to stop federated application server nodes if they are running. After you stop product processes, the product is no longer running.

Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root

> Refers to the top directory for a WebSphere Application Server node.
>
> The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.
>
> The default varies based on node type. Common defaults are `configuration_root`/`AppServer` and `configuration_root`/`DeploymentManager`.

configuration_root

> Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.
>
> The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.

The default is /wasv8config/*cell_name*/*node_name*.

plug-ins_root
> Refers to the installation root directory for Web Server Plug-ins.

profile_root
> Refers to the home directory for a particular instantiated WebSphere Application Server profile.
>
> Corresponding product variables are *server.root* and *user.install.root.*
>
> In general, this is the same as *app_server_root*/profiles/*profile_name*. On z/OS, this will always be *app_server_root*/profiles/default because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root
> Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.
>
> The corresponding product variable is *smpe.install.root*.
>
> The default is /usr/lpp/zWebSphere/V8R0.

Administering stand-alone nodes using the administrative agent

You can configure an administrative agent, view or change stand-alone application server nodes registered to the administrative agent, and view or change job manager configurations for a registered node. An administrative agent provides a single interface to administer application servers in, development, unit test, or server farm environments, for example.

Before you begin

Install the WebSphere Application Server product.

About this task

The administrative agent provides a single interface to administer multiple unfederated (stand-alone) application server nodes in, for example, development, unit test, or server farm environments. By using a single interface to administer your application servers, you reduce the overhead of running administrative services in every application server.

You can use the administrative console of the administrative agent to configure the administrative agent, view and change properties for nodes registered to the administrative agent, register and unregister application server nodes with job managers, and view and change job manager configurations for a registered node. A job manager allows you to asynchronously submit and administer jobs for a node registered to the administrative agent when the node is also registered to the job manager.

Procedure

- Set up the administrative agent environment.

  Create an administrative agent profile and one or more stand-alone application server profiles, called *nodes*, on the same computer and then register the node profiles with the administrative agent.

- Start and stop the administrative agent as needed.

  The administrative agent must be running to check for jobs for its registered stand-alone application server nodes.

- View and change properties for the administrative agent.

  1. Click **System Administration** > **Administrative agent** from the navigation of the administrative agent administrative console.

     – Optionally view the administrative agent properties on the Configuration tab and the Runtime tab.

- Optionally select **Start components as needed** on the Configuration tab. Click **Apply**, and then click **OK**.

    Selecting the setting allows administrative agent components to start dynamically as needed for applications.

- View and change properties for a node registered to the administrative agent.

    1. Click **System Administration** > **Administrative agent** > **Nodes**.

        - You can view the nodes registered to the administrative agent.

        - Optionally click **Register with Job manager** to register an application server node with a job manager.

        - Optionally click **Unregister from a Job Manager** to unregister an application server node from a job manager.

    2. Click **System Administration** > **Administrative agent** > **Nodes** > *node_name*.

        - Optionally select **Poll jobs from job manager** to have the administrative agent retrieve jobs from the job manager for this node.

        - To change other properties for the node, click the links under Additional Properties.

- View and change job manager configurations for a registered node.

    1. Click **System Administration** > **Administrative agent** > **Nodes** > *node_name* > **Job managers** to view the job managers to which the node is registered.

    2. Click **System Administration** > **Administrative agent** > **Nodes** > *node_name* > **Job managers** > *job_manager_UUID* to change job manager-related properties for the registered node.

        - Optionally change the polling interval by entering an integer value.

            The administrative agent uses the polling interval to check for jobs from the job manager for this registered node.

        - Optionally change the web address of the job manager that the administrative agent polls for this registered node.

- Unregister a stand-alone application server node from the administrative agent.

    Unregister nodes if you no longer need the node in the administrative agent environment or if you intend to delete the stand-alone application server node profile. Run the deregisterNode command to unregister a node.

Results

Depending on the tasks that you completed, you might have configured the administrative agent, registered and unregistered application server nodes with job managers, viewed or changed properties for a node registered to the administrative agent, or viewed and changed the job manager configuration for a registered node.

What to do next

You can continue to administer registered nodes from the administrative agent. You can further configure the administrative agent using the links on the configuration tab of the administrative agent panel. You can register more nodes with the administrative agent using the registerNode command. You can unregister nodes from the administrative agent using the deregisterNode command.

You can register and unregister nodes with a job manager.

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere

Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

Administrative agent

An administrative agent provides a single interface to administer multiple unfederated application server nodes in environments such as development, unit test or that portion of a server farm that resides on a single machine.

The administrative agent and application servers must be on the same sysplex, but you can connect to the sysplex from a browser or the wsadmin tool on another machine.

Multiple customers administer application servers in their development, test, and production environments by federating the application server nodes into a cell and administering the application servers from the deployment manager. However, if you have development and unit test environments, then you might prefer to run application servers whose nodes have not been federated. These application servers have some administrative disadvantages. The application servers lack a common administrative interface. Remote administration is limited to installing applications and changing application server configurations. As an alternative, you can register these application servers with an administrative agent to administer application servers from a single interface and to more fully administer application servers remotely.

You can register an application server node with the administrative agent or federate the node with a deployment manager, but not both.

gotcha: Registered nodes must have the same products as the administrative agent, and the products must be at the same version levels on the registered node and the administrative agent. This requirement is enforced because the administrative agent must have a matching environment in order to handle all of the administrative capabilities of the registered node. A node is not allowed to register with an administrative agent unless that node has an identical set of products and versions.

transition: If you were previously running on Version 7.0.0.11 or earlier, and have an administrative agent with a managed node that has mismatched products or versions, when you when you migrate to Version 8.0, that administrative agent will not be able to start the subsystem for any mismatched nodes. You must update these nodes to have the same products and versions as the administrative agents, restart the servers on the node and then restart the administrative agent, before the administrative agent can resume managing these registered nodes

An administrative agent can monitor and control multiple application servers on one or more nodes. Use the application servers only to run your applications. By using a single interface to administer your application servers, you reduce the overhead of running administrative services in every application server.

You can use the administrative agent to install applications on application servers, change application server configurations, stop and restart application servers, and create additional application servers.

You can start the administrative agent on any logical partitions (LPAR) if the administrative agent configuration is on a shared file system.

Example topology of multiple application servers managed by an administrative agent

The following example topology shows machine A with an administrative agent and two application servers, Profile01 and Profile02, that are registered with the administrative agent. The application servers on machine A each communicate with a remote web server on machine B through the web server plug-in. Firewalls provide additional security for the machines. Read the topic on planning to install WebSphere

Application Server for further information on the topology.
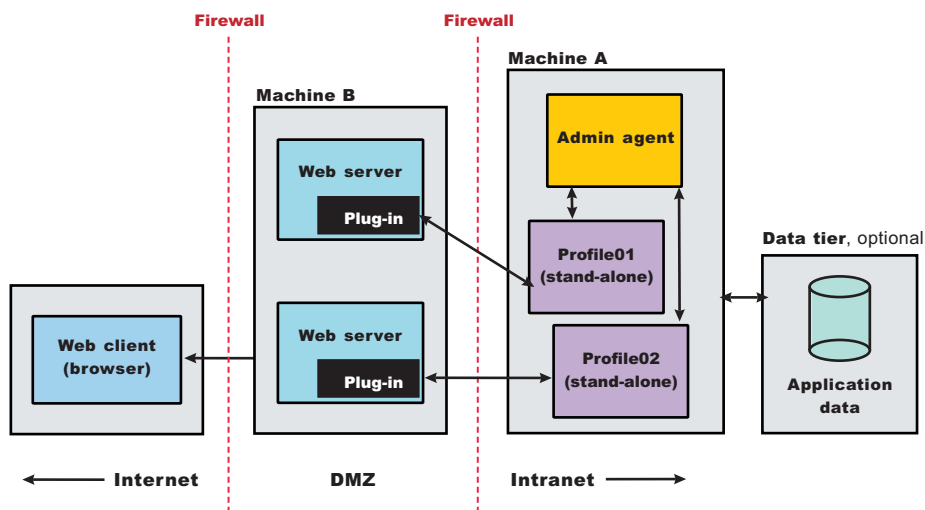
Administrative jobs for application servers registered to an administrative agent and for deployment managers

You can use a job manager to submit administrative jobs asynchronously for application servers registered to an administrative agent and for deployment managers. To make application servers registered to an administrative agent and deployment managers known to the job manager, you must register them with the job manager. Many of the management tasks that you can perform with the job manager are tasks that you can already perform with the product, such as application management, server management, and node management. However, with the job manager, you can aggregate tasks and perform those tasks across multiple application servers or deployment managers.

Administrative agent security

In a flexible management environment, a user ID must have the required authorization to use the administrative agent and to work with registered nodes.

Required security roles

You need the following roles to use the administrative agent:

Table 4. Required security roles for administrative agent tasks. Roles include administrator and roles required for the operation or node.

Administrative tasks	Required security roles
Register or unregister a base (stand-alone) node with the administrative agent	administrator
Work with the administrative agent:	Administrative roles required for the operation being performed
Work with the administrative subsystem, such as registered nodes	Administrative roles required for the registered base node

Same security domain configuration

The administrative agent supports a security configuration where all the cells in the topology share the same user registry, and therefore, the same security domain.

For the administrative agent topology, when a user logs in to the JMX connector port of an administrative subsystem, or chooses the registered node from the administrative console, the authorization table for the chosen node is used.

For example, suppose two stand-alone application servers, Node1 and Node2, are registered with an administrative agent. User1 is authorized as administrator for Node1, but is not authorized for Node2. User2 is authorized as configurator for Node2, but is not authorized for Node1. User1 can administer, operate and configure Node1 and its resources. User2 can monitor and configure Node2 and its resources. Only User1 can register or unregister a node, Node1, with the administrative agent.

Do not use DMZ proxy

A DMZ proxy does not work with the administrative agent when security is enabled. Keep security enabled and do not use the administrative agent in a DMZ proxy environment.

Setting up the administrative agent environment

An administrative agent environment consists of an administrative agent and the stand-alone application servers that it manages. Setting up an administrative agent environment involves creating an administrative agent profile and one or more stand-alone application server profiles, called *nodes*, on the same computer and then registering the node profiles with the administrative agent.

Before you begin

Install the WebSphere Application Server product.

Make sure that the nodes that you want the administrative agent to manage have the same products as the administrative agent, and the products are at the same version levels on these nodes and the administrative agent. This requirement is enforced because the administrative agent must have a matching environment to handle all the administrative capabilities of the registered node. A node cannot register with an administrative agent unless that node has an identical set of products and versions.

A DMZ proxy does not work with the administrative agent when security is enabled. Keep security enabled and do not use the administrative agent in a DMZ proxy environment.

transition: If you were previously running on Version 7.0.0.11 or earlier, and have an administrative agent with a managed node that has mismatched products or versions, when you migrate to Version 8.0, that administrative agent will not be able to start the subsystem for any mismatched nodes. You must update these nodes to have the same products and versions as the administrative agents, restart the servers on the node and then restart the administrative agent, before the administrative agent can resume managing these registered nodes.

About this task

You can use an administrative agent to manage base (stand-alone) application servers that are on the same computer.

Administrative agents and the managed nodes are part of the flexible management environment.

To add an administrative agent to your environment, create an administrative agent profile using the manageprofiles command or the Profile Management Tool. To add a node, create a stand-alone application server profile and then register the stand-alone application server with the administrative agent.

The node must be on the same computer as the administrative agent.

On a Network Deployment product, you also can add job managers to your flexible management environment. A job manager is a single management server from which you can remotely manage multiple administrative agents, deployment managers, and stand-alone application servers. From an administrative agent, you can register stand-alone application server nodes with a job manager. Nodes that register with a job manager maintain their own administrative capabilities. Additionally, the nodes periodically poll the job managers to determine whether there are jobs posted there that require action. The advantage to a job manager configuration is the ability to coordinate management actions across multiple varied environments.

Ensure that the profiles in the flexible management environment either all have security enabled or all have security disabled.

Procedure

1. Determine the topology for your administrative agent environment.

   Determine which computers, stand-alone application server nodes, and node resources such as applications that you want to use.

   To manage stand-alone application servers, use an administrative agent on each computer where the stand-alone application servers reside. For more information, see Scenarios 5 in the Planning to install WebSphere Application Server topic.

2. Determine the security roles needed for your administrative agent environment.

   For an administrative agent environment, you typically have one administrative agent profile and one or more stand-alone application server profiles on the same computer. The stand-alone application server nodes are registered to the administrative agent. Profiles in the environment must either all have security enabled or all have security disabled. When you create the profiles, you can specify security options, user names, and passwords.

   You must have security roles that authorize you to work with an administrative agent and to manage registered nodes and resources on those nodes. For more information, see the administrative agent security topic.

3. Create a management profile for the administrative agent.

   You can use the Profile Management Tool or the manageprofiles command.

   For example, in the Profile Management Tool, select the **Management** environment and click **Next**, select the **Administrative agent** server type, and select options that create the profile. By default, an administrative agent has its own administrative console, administrative security is enabled, and the console port is 9065. To disable administrative security, to specify a security certificate, or to change the default ports, use the advanced profile creation option when creating the administrative agent profile.

   By default, the first administrative agent profile in a product installation is named `AdminAgent01` and its server name is `adminagent`.

   For more information, see the topic on creating management profiles for administrative agents.

   For manageprofiles examples, see the topic on the manageprofiles command. For `-templatePath`, specify the `management` template. For `-serverType`, specify `ADMIN_AGENT`.

4. Create profiles for the stand-alone application server nodes that you intend to have in your flexible management environment.

   Create profiles for one or more stand-alone application server nodes that reside on the same computer as the administrative agent profile. You can use the Profile Management Tool or the manageprofiles command.

For example, in the Profile Management Tool, select the **Application server** environment and click **Next**, and then select options that create the profile. By default, an application server has its own administrative console, administrative security is enabled, and the console port is 9060. To disable administrative security, to specify a security certificate, to specify to install sample application, or to change the default ports, select the advanced profile creation option when creating the application server profile.

By default, the first application server profile in a product installation is named `AppSrv01` and its server name is `server1`.

For more information, see the topic on creating application server profiles.

For manageprofiles examples, see the topic on the manageprofiles command. For `-templatePath`, specify the `default` template. Do not specify a `-serverType` parameter.

5. Start the administrative agent server.
   - Run the startServer command.

     For example, suppose the AdminAgent01 profile has the server name `adminagent`. Run the following command from the `bin` directory of the AdminAgent01 profile:

     ```
     startServer adminagent
     ```
   - Use the START command to start the administrative agent.

     ```
     START administrative_agent_proc_name,JOBNAME=server_short_name,
         ENV=cell_short_name.node_short_name.server_short_name
     ```

   If the administrative agent starts successfully, the `open for e-business` message displays and is written to the administrative agent `startServer.log` file:

   ```
   Server launched.  Waiting for initialization status.
   Server adminagent open for e-business; process id is 1932.
   ```

   For more information, see the topic on starting and stopping the administrative agent.

6. Register the stand-alone application server nodes with the administrative agent.

   Run the registerNode command of the administrative agent.

   When you run the registerNode command, you can optionally specify parameters such as `-node` to assign a node name and `-port` to assign an administrative agent connector port. If security is enabled for the node that you are registering and the node user name and node password are different than those used for the administrative agent, specify values for `-nodeusername` and `-nodepassword`. For more information, see the topic on the registerNode command.

   To register the AppSrv01 profile with the administrative agent, run the following command from the `bin` directory of the administrative agent profile:

   ```
   registerNode.sh -profilePath app_server_root/profiles/default
   ```

   For more information, see the topic on the registerNode command.

7. Verify that the nodes have been registered to the administrative agent.

   You can use the administrative agent console or wsadmin scripting commands to see a list of nodes that are registered with the administrative agent.
   - Use the administrative agent console to see a list of managed nodes.
     a. Start the administrative agent console.
     b. On the opening page of the administrative agent console, select to administer the administrative agent. The administrative agent has a name such as *host_name*AANode01.
     c. Log in to the administrative agent console.
     d. Examine the Nodes page.
        1) Click **System administration** > **Administrative agent**.
        2) On the Configuration tab of the Administrative agent page, click **Nodes**.
     e. Ensure that the Nodes page lists nodes that have been registered with the administrative agent.
   - Use the AdminConfig list command to see a list of managed nodes. Run the following wsadmin scripting commands from the administrative agent `bin` directory.
     – To use the Jython scripting language, enter the following two commands in succession:

```
wsadmin -lang jython

print AdminConfig.list('ManagedNode')
```

– To use the Jacl scripting language, enter the following two commands in succession:

```
wsadmin
```

```
$AdminConfig list ManagedNode
```

After you verify that the stand-alone application server nodes are registered with the administrative agent, enter `quit` to exit the wsadmin scripting tool.

8. Start the stand-alone application server nodes.

Run the startServer command.

```
startServer server1
```

If the server starts successfully, the `open for e-business` message displays and is written to the `startServer.log` file.

For more information, see topics on the startServer command and on starting application servers.

Results

The administrative agent environment is set up and the nodes are running.

What to do next

Use the administrative agent to monitor and configure the stand-alone application server nodes.

From the administrative agent, you can register the stand-alone application server nodes with a job manager. After the nodes are registered with a job manager, you can remotely manage the administrative agent and the stand-alone application servers. The nodes periodically poll the job manager to determine whether there are jobs posted that pertain to the nodes.

You can use the administrative agent console to register a stand-alone application server node with a job manager:

1. Click **System administration** > **Administrative agent**.
2. On the Configuration tab of the Administrative agent page, click **Nodes**.
3. On the Nodes page, select the node to register with the job manager and click **Register with Job Manager**.
4. On the Register with Job Manager page, specify a node name, specify a job manager administrative console port number, optionally specify other parameters such as the job manager user name and password, and click **OK**.

   Note: For **Port**, if security is not enabled, specify 9960 for an unsecure job manager administrative console port. If no port number is specified, the default secure port number 9943 is used.

To unregister a node later, you can use the same Nodes page, except click **Unregister with Job Manager**.

Instead of using the administrative agent console to register and unregister with a job manager, you also can use the ManagedNodeAgent registerWithJobManager wsadmin command. To unregister a node, use the ManagedNodeAgent unregisterWithJobManager wsadmin command.

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere

Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

Starting and stopping the administrative agent

In your flexible management environment, you can start the administrative agent by using the startServer command. You can stop the administrative agent by using the stopServer command.

Before you begin

Before you can start or stop the administrative agent, you must first install the product.

About this task

Start the administrative agent so that you can manage multiple application servers. Stop the administrative agent as needed, such as when migrating to a new version of the product, when uninstalling the product, and so on.

Procedure

- Start the administrative agent.

  Use one of these methods to start the administrative agent:

  - Use the startServer command:

    ```
    startServer <administrative_agent>
    ```

    where *administrative_agent* is name of the administrative agent that you want to start.

- Stop the administrative agent.

  Use one of these methods to stop the administrative agent:

  - Use the stopServer command:

    ```
    stopServer <administrative_agent>
    ```

    where *administrative_agent* is name of the administrative agent that you want to stop.

Results

You have started the administrative agent and have optionally stopped it.

What to do next

Administer application servers using the administrative agent. You can do such tasks as configure the administrative agent, view or change properties for a node registered to the administrative agent, or view and change the job manager configuration for a registered node.

Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root
　　　　　Refers to the top directory for a WebSphere Application Server node.

　　　　　The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.

　　　　　The default varies based on node type. Common defaults are `configuration_root`/AppServer and `configuration_root`/DeploymentManager.

configuration_root

> Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.
>
> The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.
>
> The default is `/wasv8config/`*cell_name*`/`*node_name*.

plug-ins_root

> Refers to the installation root directory for Web Server Plug-ins.

profile_root

> Refers to the home directory for a particular instantiated WebSphere Application Server profile.
>
> Corresponding product variables are *server.root* and *user.install.root.*
>
> In general, this is the same as *app_server_root*`/profiles/`*profile_name*. On z/OS, this will always be *app_server_root*`/profiles/default` because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root

> Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.
>
> The corresponding product variable is *smpe.install.root.*
>
> The default is `/usr/lpp/zWebSphere/V8R0`.

Administrative agent settings

Use this page to configure the administrative agent and view its properties.

To view this page in the administrative agent console, click **System administration** > **Administrative agent**.

Name

Specifies the administrative agent server name. The name is read-only.

Node

Specifies a name for the administrative agent node. The node name is unique within the cell. The node name is read-only.

By default, a node name is the hostname appended with Node01. For example, a node on a computer with the host name of MyComputer is named MyComputerNode01 by default.

However, the node name is a purely logical name for a group of servers. The node name does not have to contain the host name.

Short name

Specifies the short name of the administrative agent server.

The server short name must be unique within a cell. The short name identifies the server to the native facilities of the operating system, such as workload manager (WLM), Automatic Restart Manager, System Authorization Facility (SAF), for example, Resource Access Control Facility (RACF), started task control, and others.

The name can be 1-8 alphanumeric or national language characters and cannot start with a numeral.

The system assigns a cell-unique, default short name.

Data type   String

Unique ID

Specifies the unique ID of this administrative agent server.

The unique ID property is read-only. The system automatically generates the value.

Data type   String

Start components as needed

Select this property if you want the server components started as they are needed for applications that run on this server.

When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

gotcha: If you are running other WebSphere products on top of this product, make sure that those other products support this functionality before you select this property.

Process ID

Specifies the read-only process ID of the administrative agent.

Cell name

Specifies the read-only cell name of the administrative agent.

Node name

Specifies the read-only node name of the administrative agent.

State

Specifies the read-only state of the administrative agent, such as started or stopped.

Node collection for the administrative agent

Use this page to view the application server nodes that are registered to the administrative agent. The administrative agent provides a single interface to the registered nodes.

To view this administrative console page, click **System administration** > **Administrative agent** > **Nodes**.

On multiple-server products, application server nodes already registered to an administrative agent also can be registered to job managers. Job managers enable you to asynchronously submit and administer jobs for large numbers of unfederated application servers, deployment managers, and host computers over a geographically dispersed area.

Table 5. Button descriptions. Use the buttons to register and unregister nodes with a job manager.

Button	Resulting action
Register with Job Manager	Registers the selected node with a job manager. The node must already be registered with an administrative agent. Also, the node must be at an equal or lesser version number than the job manager. A Version 8 job manager can manage Version 8 and 7 nodes. A Version 7 job manager can manage Version 7 nodes. The fix pack portion of the version number does not matter; for example, a Version 7.0.0.3 job manager can manage a node at Version 7.0.0.9, which is Version 7 with fix pack 9 installed.
Unregister from a Job Manager	Unregisters the selected node from a job manager.

Name

Specifies a name for an application server node that is registered to the administrative agent. The name is read-only.

Unique ID

Specifies the unique ID of this application server.

The unique ID property is read-only. The system automatically generates the value.

Data type   String

Registered nodes settings

Use this page to view properties for a node registered to the administrative agent. The properties are name, unique ID, and poll jobs from job manager.

To view this administrative console page, click **System administration** > **Administrative agent** > **Nodes** > *node_name*.

Name: Specifies the name of an application server registered to the administrative agent. The name is read-only.

Unique ID:

Specifies the unique ID of this application server.

The unique ID property is read only. The system automatically generates the value.

Data type   String

Poll jobs from job manager: Select the option so that the administrative agent retrieves jobs from the job manager for this node. The jobs start when the administrative agent retrieves them. The polling interval is defined on the Job manager page of the administrative console for the administrative agent and controls how often the administrative agent checks for new jobs. The default is to retrieve the jobs.

Unregistering nodes of the administrative agent

You can unregister application server nodes so that they are no longer registered to an administrative agent. Unregister nodes if you no longer need the node in the administrative agent environment or if you intend to delete the application server node profile. After you unregister a node from an administrative agent, you can use the node stand-alone, register the node with another administrative agent, or delete the application server node profile.

Before you begin

The application server node that you want to remove from the administration agent environment must be registered with the administrative agent. Start the administrative agent if it is not running already.

If the application server node is registered with a job manager, unregister the node from the job manager. You can use the administrative agent console or wsadmin commands to unregister the node:

- Use the administrative agent console to unregister the stand-alone application server node from a job manager.
  1. Log in to the administrative agent console. The administrative agent name resembles *host_name*AANode01.
  2. Click **System administration** > **Administrative agent**.
  3. On the Configuration tab of the Administrative agent page, click **Nodes**.

4. On the Nodes page, select the node to unregister from the job manager and click **Unregister from a Job Manager**.

5. On the Unregister from a Job Manager page, specify a node name, specify a job manager administrative console port number, optionally specify other parameters such as the job manager user name and password, and click **OK**.

   For **Port**, if security is not enabled, specify `9960` for an unsecure job manager administrative console port. If no port number is specified, the default secure port number 9943 is used.

- Run the wsadmin unregisterWithJobManager command in the ManagedNodeAgent command group to unregister the stand-alone application server node from a job manager.

  When you run the unregisterWithJobManager command, specify the name of the stand-alone application server node that is managed by the job manager for the required `-managedNodeName` parameter. Other parameters are optional.

  ```
  AdminTask.unregisterWithJobManager('[-host myJobMgrHostname -port 8989
   -managedNodeName myAppServerNodeName]')
  ```

  The default value for the `-host` parameter is `localhost`.

  The default value for the `-port` parameter is `9943`, the job manager administrative console secure port number. If security is disabled, specify `9960`, the default unsecure port number.

  For more information about the unregisterWithJobManager command and parameters, see the topic on the ManagedNodeAgent command group for the AdminTask object.

  If the system fails when unregistering a stand-alone application server from a job manager, run the cleanupTarget command in the JobManagerNode group to clean up job manager registration information. See the topic on the JobManagerNode command group for the AdminTask object.

About this task

To unregister a node, run the deregisterNode command from the `bin` directory of the administrative agent. Step 1 describes how to run the deregisterNode command.

When you unregister a node, the node configuration is retained, but is marked as not registered with the administrative agent. If the node that you unregister had the administrative console or management Enterprise JavaBeans (EJB) applications installed before registering the node, they are re-enabled.

Running the deregisterNode command might result in a null pointer exception if the application server node profile is corrupted or unusable. If you receive the null pointer exception, the process to unregister the application server from the administrative agent failed. You receive ADMU0116I, ADMU0128I, ADMU0211I, ADMU0113E, and ADMU1211I messages in the error log. Step 2 describes how to remove a node and related end points if there is a null pointer exception.

If the application server node profile is deleted before the node is unregistered, running the deregisterNode command is ineffective. Because the profile no longer exists, the administrative agent does not recognize the profile. Complete Step 2 to remove the node and related end points from the administrative agent environment.

Procedure

1. Unregister a node using the deregisterNode command.

   If the node that you want to unregister exists, run the deregisterNode command, specifying the profile path of the node to unregister:

   ```
   deregisterNode -profilePath profile_root/profile_name
   ```

   For example, to unregister the AppSrv02 profile from the administrative agent environment, run the following command:

   ```
   deregisterNode -profilePath profile_root/AppSrv02
   ```

   See the topic on the deregisterNode command for information about command parameters.

2. If a null pointer exception results from running the deregisterNode command or if the node profile has been deleted, run wsadmin commands that remove the registered node and related end points.

   a. On a command line, run a command to start the wsadmin scripting tool from the administrative agent `bin` directory.

      To use the Jython scripting language, enter:

      ```
      wsadmin -lang jython
      ```

      To use the Jacl scripting language, enter:

      ```
      wsadmin
      ```

   b. If you do not know the name of the node to remove, run the AdminConfig list command to list nodes that are registered with the administrative agent and find the node to remove in the list.

      For Jython:

      ```
      print AdminConfig.list('ManagedNode')
      ```

      For Jacl:

      ```
      $AdminConfig list ManagedNode
      ```

      The list of registered nodes that is displayed resembles the following:

      ```
      nodeA(cells/myAACell01/managednodes/nodeA|managednode.xml#ManagedNode_1239121412703)
      nodeB(cells/myAACell01/managednodes/nodeB|managednode.xml#ManagedNode_1239121498500)
      ```

      This list shows that `nodeA` and `nodeB` are registered nodes of the `myAACell01` administrative agent.

   c. Issue wsadmin commands that remove the node.

      To remove `nodeA` and save the changes, run the following commands in succession.

      For Jython:

      ```
      mn = AdminConfig.getid('/ManagedNode:nodeA/')

      AdminConfig.remove(mn)

      AdminConfig.save()
      ```

      For Jacl:

      ```
      set mn [$AdminConfig getid /ManagedNode:nodeA/]

      $AdminConfig remove $mn

      $AdminConfig save
      ```

   d. Run wsadmin commands that remove end points that were generated for the subsystem when the node profile was registered.

      Run the following commands sequentially to remove end points for `nodeA`. The `for` command in Jython and the `foreach` command in Jacl are one-line commands that are shown on multiple lines for publication.

      For Jython:

      ```
      import java.lang.System as System

      lineSeparator = System.getProperty("line.separator")

      neps = AdminConfig.list("NamedEndPoint").split(lineSeparator)

      for nep in neps:
        set name = AdminConfig.showAttribute(nep, "endPointName")
        if (name.endswith("nodeA") == 1):
          AdminConfig.remove(nep)

      AdminConfig.save()

      quit
      ```

      For Jacl:

```
set neps [$AdminConfig list NamedEndPoint]

foreach nep $neps {set name [$AdminConfig showAttribute $nep endPointName];
if {[string last "nodeA" $name] != -1} {$AdminConfig remove $nep}}

$AdminConfig save

quit
```
   e. Restart the administrative agent.

   To restart an administrative agent named `adminagent`, run the following commands from a
   command prompt at the `bin` directory of the administrative agent profile:
```
stopServer adminagent
```
```
startServer adminagent
```
   f. Verify that the node is no longer registered with the administrative agent.

Results

The application server node is no longer registered with the administrative agent.

What to do next

You can use the unregistered node stand-alone or register the node with another administrative agent.
Optionally, use the manageprofiles command to delete the application server profile.

Register or unregister with job manager settings

Use this page to either register a node to a job manager or unregister a node from a job manager. The
node can be a deployment manager or a node registered to an administrative agent.

You can use the administrative agent administrative console to register or unregister a node with the job
manager. The node that you register with the job manager is already registered to the administrative
agent. For example, click **System administration > Administrative agent > Nodes**. Select **Register with
Job Manager** to register a node to a job manager or **Unregister from a Job Manager** to unregister a
node from a job manager.

You can use the deployment manager administrative console to register or unregister a deployment
manager with the job manager. Click **System administration > Deployment manager**. Under Additional
properties, click **Job Managers > Register/unregister with job manager**.

Managed node name
A required setting that specifies the name of the managed node. For a deployment manager, the managed
node name is the name of the deployment manager node, usually *host_name*CellManager01.

Alias
An optional setting that specifies the alias of the managed node to enroll. Specify an alias if the managed
node name is in use by another node.

Host name
An optional setting that specifies the host name to use to identify the job manager. The default value is
`localhost`.

Port
An optional setting that specifies the job manager administrative console port number. If security is
enabled, use the secure port number. If security is disabled, use the unsecure port number. The default
secure port number is 9943, and the default unsecure port number is 9960. If no port number is specified,
9943 is used.

User name

Specifies the user name to log into the job manager when security is enabled.

Password

Specifies the password to log into the job manager. This setting is required when the user name setting is required.

Confirm password

Specifies the password a second time. This setting is required when a user name and a password are required.

Job manager collection

Use this page to view the job managers to which this node is registered. The job managers enable you to asynchronously submit and administer jobs, such as manage applications, for this node.

To view this page on the administrative agent administrative console, click **System administration** > **Administrative agent** > **Nodes** > *node_name* > **Job managers**.

To view this page on the deployment manager administrative console, click **System administration** > **Deployment manager** > **Job managers**.

UUID

Specifies the Universal Unique Identifier (UUID), which uniquely identifies the job manager to which the administrative agent connects.

URL

Specifies the web address of the job manager to which the node is registered.

Job manager settings

Use this page to view the Universal Unique Identifier (UUID) of the job manager, specify the polling interval to check for jobs on the job manager, and specify the web address of the job manager.

To view this page on the administrative agent administrative console, click **System administration** > **Administrative agent** > **Nodes** > *node_name* > **Job managers** > *job_manager_UUID*.

To view this page on the deployment manager administrative console, click **System administration** > **Deployment manager**. Under Additional Properties click **Job managers** > *job_manager_UUID*.

UUID:

Specifies a Universal Unique Identifier (UUID), which uniquely identifies the job manager.

In an administrative agent console, the UUID identifies the job manager to which the administrative agent connects.

In a deployment manager console, the UUID identifies the job manager to which the deployment manager connects.

The UUID is read-only.

Polling interval:

Specifies in seconds the time that elapses during a polling interval.

In an administrative agent console, the administrative agent uses the polling interval to determine how often to poll a job manager for new jobs for the registered node.

In a deployment manager console, the deployment manager uses the polling interval to determine how often to poll a job manager for new jobs.

The default value is 30 seconds.

URL:

Specifies a web address that is used to connect to the job manager.

In an administrative agent console, the URL specifies the web address that the administrative agent uses to connect to the job manager.

In a deployment manager console, the URL specifies the web address that the deployment manager uses to connect to the job manager.

The formats are `http://`*`host`*`:`*`port`*`/otis/OMADMServlet` or `https://`*`host`*`:`*`port`*`/otis/OMADMServlet`, where *host* is the host name of the job manager and *port* is the port of the job manager.

Administering nodes remotely using the job manager

In a flexible management environment, you can asynchronously submit and administer jobs for large numbers of stand-alone application servers, deployment managers, and host computers over a geographically dispersed area. At the remote machines, you can use jobs to manage applications, modify the product configuration, or do general purpose tasks such as run a script.

Before you begin

Install the WebSphere Application Server product.

About this task

A job manager is a single management server from which you can remotely manage multiple administrative agents, deployment managers, stand-alone (unfederated) application servers, and host computers.

In a flexible management environment, the job manager enables you to asynchronously submit and administer jobs for large numbers of stand-alone application servers, deployment managers, and host computers over a geographically dispersed area. Many of the management tasks that you can perform with the job manager are tasks that you can already perform with the product, such as application management, server management, and node management. However, with the job manager, you can aggregate the tasks and perform the tasks across multiple application servers, deployment managers, or host computers.

In contrast to a deployment manager, the job manager does not exclusively inherit the administrative functions of its registered targets. Targets that register with a job manager maintain their own administrative capabilities. Additionally, the targets periodically poll the job managers to determine whether there are jobs posted there that require action. You can administer all registered targets separately from the job manager. The advantage to a job manager is that you can administer targets in multiple varied environments.

To administer targets, you submit jobs using the job manager. You can submit jobs for individual targets or for groups of targets that you define. After you submit a job, you can check the job status, check the status of targets, and check the status of target resources. The status of managed resources is not necessarily up-to-date. Status in the job manager administrative console is updated only when a status job or an inventory job for the target containing the resource completes successfully. You can view target resources

for targets and groups of targets that you administer. You can configure the job manager and view its properties.

Procedure

- Set up the job manager environment.

  Create a job manager profile and any other profiles that are needed for the environment, synchronizing the clocks on all environment computers, and then registering the target profiles with the job manager. You can register stand-alone application servers that are already registered with an administrative agent or deployment manager profiles. You can also add remote host computers as targets.
- Start and stop the job manager as needed.

  The job manager must be running to submit jobs and to enable targets to poll the job manager for jobs.
- Configure job managers.

  You can specify settings such as the default job expiration, the job manager web address, and the mail provider Java Naming and Directory Interface (JNDI) name for the job manager. You can view job manager properties such as the process ID and the state of the job manager.
- View information on targets using a job manager.

  You can view targets with their version numbers based on the results of the Find option and view target resources for targets that you select. You can also view the properties and property values for a particular target.
- View information on target resources.

  You can view server, application, node, and cluster resources that are associated with targets and groups of targets registered to the job manager. You can also view the status of specific resources at each target and view properties for a particular target resource as a name-value pair.
- Submit jobs to administer servers, files, and applications.

  You can submit jobs to remote targets to manage applications, modify the product configuration on remote machines, or do general purpose tasks such as run a script. You can specify when the jobs start, whether they are recurring, and when they are no longer available for submission.
- Check the status of jobs.

  You can check the status of jobs, the status of jobs at their targets, and the job history of targets. You can suspend, resume, or delete jobs on the Job status collection page.
- Administer groups of targets using a job manager.

  You can create, modify, delete, and view groups of targets. Groups of targets make job submission simpler because you can submit a job for a group of targets instead of entering multiple target names for a job submission.
- Change the polling interval.

  You can increase or decrease the polling interval that a target uses to poll the job manager for jobs. The default polling interval is 30 seconds.

Results

Depending on the tasks that you completed, you might have submitted jobs, checked the status of jobs, viewed targets and target resources, or administered groups of targets.

What to do next

If you no longer need a target, unregister the target. You can unregister targets from a job manager in the following ways:

- To deregister application servers or deployment managers, run the wsadmin unregisterWithJobManager command in the ManagedNodeAgent command group or click **Unregister from a Job Manager** on the Register or unregister with job manager settings page of a deployment manager or administrative agent console.

- To deregister hosts, run the wsadmin unregisterHost command in the JobManagerNode command group or click **Delete Host** on the Targets page of a job manager or deployment manager console.

Use a deployment manager to unregister the deployment manager from a job manager. Use an administrative agent to unregister a stand-alone application server. To fully remove a stand-alone application server from the flexible management environment, you must first unregister the stand-alone application server from a job manager and then unregister it from an administrative agent.

Note: Unregister a node before deleting its profile. For example, `AppSrv02` is a stand-alone application server that is registered as `nodeB`. Use the administrative agent to unregister `nodeB` before deleting profile `AppSrv02`. For more information, see the topic on unregistering nodes of the administrative agent.

If the system fails when unregistering a target from a job manager, run the cleanupTarget command in the JobManagerNode group to clean up job manager registration information. The command does not remove the job history of the node that you are unregistering. Jobs in progress continue to run, but new jobs do not start for the node. See the topic on the JobManagerNode command group for the AdminTask object.

Job manager

In a flexible management environment, a job manager allows you to submit administrative jobs asynchronously for application servers registered to administrative agents, for deployment managers, and for host computers. You can submit these jobs to a large number of servers over a geographically dispersed area.

You can register stand-alone application servers that are registered to administrative agents, deployment managers, and host computers with the job manager. After you register stand-alone application servers, deployment managers, or host computers as targets, you can queue administrative jobs directed at the targets through the job manager.

To register application server nodes and deployment managers with the job manager, use an administrative console or the wsadmin registerWithJobManager command. The command is in the ManagedNodeAgent command group.

To register hosts with the job manager, use an administrative console or the registerHost command. The command is in the JobManagerNode command group.

Note: You can complete job manager actions and run jobs from a deployment manager. The deployment manager administrative console has **Jobs** navigation tree choices similar to those in the job manager administrative console.

Use the job manager to asynchronously administer job submissions. You can complete the following tasks:
- Set the job submission to take effect at a specified time.
- Set the job submission to expire at a specified time.
- Specify that the job submission occur at a specified time interval.
- Notify the administrator through email that the job has completed.

Each application server, deployment manager, or host registered with the job manager is known as a target to the job manager. Groups of targets are those groups that you create so that you can make job submission easier. You can submit a job for a group of targets instead of entering multiple target names for a job.

Many of the management tasks that you can perform with the job manager are tasks that you can already perform with the product, such as application management, server management, and node management. However, with the job manager, you can aggregate tasks and perform those tasks across multiple targets.

Example uses of job manager

The following hypothetical company environments are examples of situations where a job manager is useful:

Branch office environment
> A business has a thousand stores geographically dispersed across the continent. Each store contains either a few application servers, or a small WebSphere Application Server, Network Deployment cell consisting of two or three machines. Each store is managed locally for daily operations. However, each store is also connected to the data center at the company headquarters, potentially thousands of miles away. Some connections to the headquarters are at modem speeds. The headquarters uses the job manager to periodically submit administrative jobs for the stores.

Environment consisting of hundreds of application servers
> An administrator sets up hundreds of low-cost machines running identical clones of an application server. Each application server node, which is registered with an administrative agent, is registered with the job manager. The administrator uses the job manager to aggregate administration commands across all the application servers, for example, to create a new server, or to install or update an application.

Environment consisting of dozens of deployment manager cells
> An administrator sets up hundreds of application servers, which are divided into thirty different groups. Each group is configured within a cell. The cells are geographically distributed over five regions, consisting of three to seven cells per region. Each cell is used to support one to fifteen member institutions, with a total of 230 institutions supported. Each cell contains approximately thirty applications, each running on a cluster of two for failover purposes, resulting in a total of 1800 application servers. The administrator uses the job manager to aggregate administration commands across all the cells, for example, to start and stop servers, or to install or update an application.

Example topology

The following example topology shows a deployment manager and a federated node that is managed by the deployment manager on machine A; two application servers, Profile01 and Profile 02, registered with an administrative agent on machine C; a job manager on machine D; and a web server on machine B. Firewalls provide additional security for the machines. The administrative agent and the deployment manager are registered to the job manager. The administrative agent and deployment manager periodically poll the job manager to determine whether the job manager posted jobs that require action. Read the topic on planning to install WebSphere Application Server for further information on the topology.

Machine D

Job manager

Firewall   Firewall

Machine B   Machine A   Machine C

Web client   Web server   dmgr   Admin agent
(browser)   (deployment
                    Plug-in   manager)
                                                                Profile01
                                         Node agent   (stand-alone)

                                                                                Data tier, optional
                                         Profile01   Profile02
                                         (managed   (stand-alone)
                                         node)   Application
                                                                                data

←——— Internet   DMZ   Intranet ———→
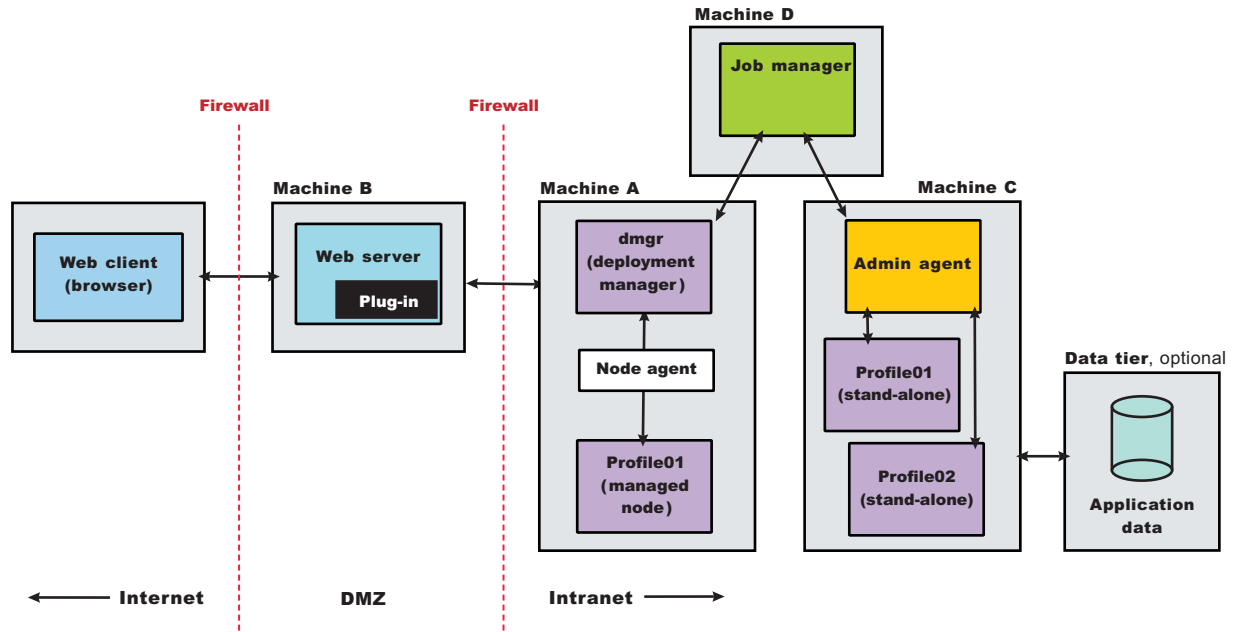
Job manager security

In a flexible management environment, a user ID must have the required authorization to use the job manager and to work with registered nodes.

Required security roles

You need the following roles to use the job manager:

Table 6. Required security roles for job manager tasks.   Roles include administrator, operator, configurator, and monitor.

Administrative tasks	Required security roles
Register or unregister with the job manager	administrator
Submit a job	operator
Change the job manager configuration	configurator
Read the job manager configuration or job history	monitor

If base (stand-alone) application server nodes that are managed by an administrative agent are registered to a job manager, you need the following roles to use the administrative agent and manage its nodes:

Table 7. Required security roles for administrative agent tasks.   Roles include administrator and roles required for the operation or node.

Administrative tasks	Required security roles
Register or unregister a base (stand-alone) node with the administrative agent	administrator
Work with the administrative agent:	Administrative roles required for the operation being performed
Work with the administrative subsystem, such as registered nodes	Administrative roles required for the registered base node

When a job runs on a target, the user must have privileges that include the role required for that job. For example, a job to create an application server requires a minimum configurator role on either the base node or WebSphere Application Server, Network Deployment cell.

Basic security configuration

The administrative agent and job manager support two different basic security configurations:

- Same security domain

  In this configuration, all the cells in the topology share the same user registry, and therefore, the same security domain. The same is true of the administrative agent and its registered base nodes, and also any job manager or WebSphere Application Server, Network Deployment cells in the topology.

- Different security domains

  In this configuration, all the cells are configured with different user registries, and therefore different security domains.

For the administrative agent topology, when a user logs in to the JMX connector port of an administrative subsystem, or chooses the registered node from the administrative console, the authorization table for the base node is used.

For example, suppose User1 is authorized as administrator for the first base node, but is not authorized for the second node. User2 is authorized as configurator for the second node, but is not authorized for the first node. The Same user registry figure illustrates this example:
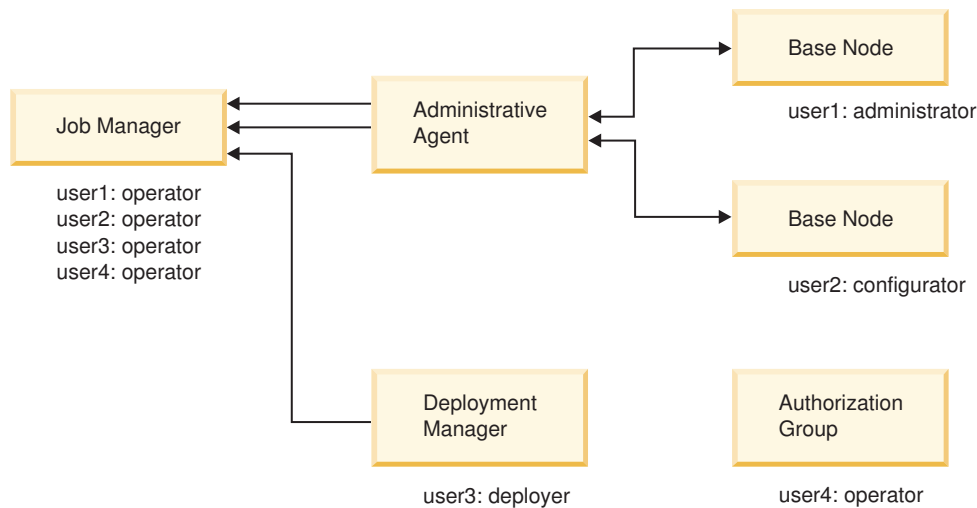
Further suppose User1 can log in to job manager as an operator with a user name and password. User1 can also log in to the deployment manager as a monitor with a user name and password. The Different user registry figure illustrates this example:

Base Node

Job Manager

Administrative
Agent

user1@jmgr: administrator

user1@jmgr: operator
user2@jmgr: operator
user3@jmgr: operator
user4@jmgr: operator

Base Node

user2@jmgr: configurator

Deployment
Manager

Authorization
Group

user1: monitor

user4@jmgr: operator
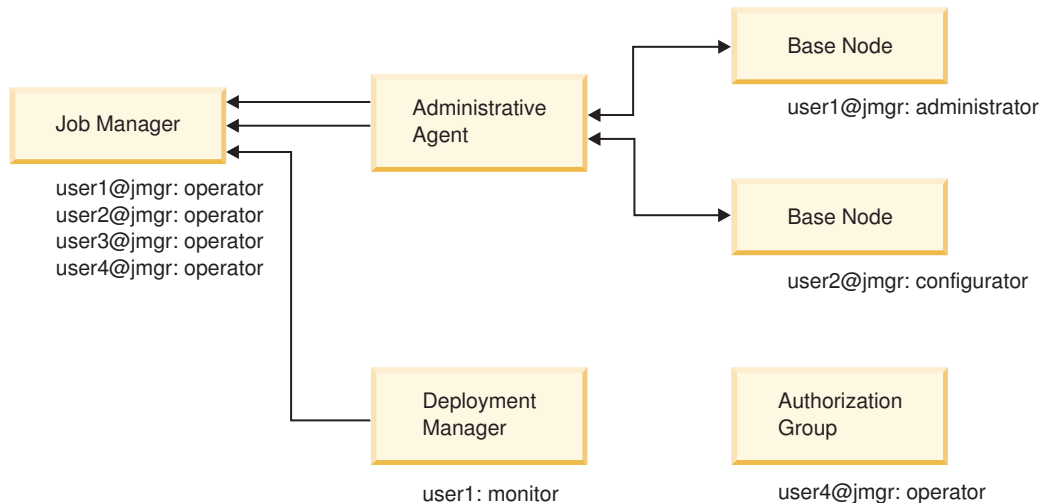
Although User1 has the same user name for both the job manager and the deployment manager, User1 might as likely have different user names and passwords.

Transfer of security information

When the product transfers a job from the job manager to the administrative agent, deployment manager or host computer, the product also transfers security information about the job submitter. This transfer authenticates and authorizes the user while running the job. The following user security information might be passed with a submitted job:

- User name and password

  When submitting a job, the user might specify a user name and password. When the job reaches the administrative subsystem or the deployment manager, the user name and password are used to log in.

  For the same user registry configuration, if John submits a job against the first base node, he can specify his user name and password as part of the job. The user name and password are used to log in against the first administrative subsystem, and the job runs. If John submits a job against the deployment manager cell or the second base node, the job fails because John is not authorized.

  For the different user registry configuration, John can submit a job against the deployment manager cell, specifying his user name and password for the deployment manager cell. When the job reaches the deployment manager, login succeeds, and the job runs. If John submits a job against the base nodes, access is denied, and the job fails.

- Security token

  If the user does not specify the user name and password with a job, the credential of the user is automatically persisted as a security token in the job database. The token contains the user security attributes, including groups. When a job is fetched, the token is used to authenticate and authorize against the administrative subsystem or the deployment manager.

  For the same user registry configuration, John can submit a job against the first base node without specifying user name and password. The job runs because John's credential is automatically propagated as a security token to the administrative subsystem, and used to authenticate and authorize him for the job. If John submits a job against the second base node or the deployment manager cell, the job fails because his security token is not authorized in these two environments.

  For the different user registry configuration, a user's security token does not automatically allow the submitted job to run against the administrative subsystem or the deployment manager. To enable a user token for a different realm, you must use the multiple security domain feature. First, you must establish the job manager realm as a trusted realm for the registered base nodes and the deployment manager

cell. In addition, you must import the access ID of the user from the job manager into the local authorization table and give the access ID a role. Then, the user can submit a job without passing user name and password.

Suppose John is an operator on the job manager, but his access ID is imported as an administrator in administrative authorization table of the first base node. Although John does not exist in the user registry of the base node, by passing the security token and the administrative authorization table definition, John is authorized as an administrator on the base node. John can submit a job for the first base node without having to specify a user name or password.

If John submits a job against the deployment manager, the job fails. John's security token is from the job manager realm, and John's access ID has not been authorized for the deployment manager. In this case, the administrator can export John's access ID from the job manager and import it to the deployment manager. Or, John can submit a job passing user name and password that he had with the deployment manager, which enables John to run jobs with monitor role.

The same mechanism works if the fine grained security feature is in use. You must be authorized in the authorization table for a new authorization group. The authorization table might also contain an external access ID.

Mixed registries configuration

In a more complex topology, where some cells share the same user registry and some cell do not, the following rules apply:

- You can always specify a user name and password during job submission if the target node or deployment manager recognizes your user name and password.
- If the job manager and the target node or deployment manager have the same user registry, then job submission does not require a user name and password. However, you must be authorized for the target node or deployment manager.
- If the job manager and the target node or deployment manager have different user registries, trusted realms have been established, and the access ID of the job submitter has been imported into the administrative authorization table of the target node or deployment manager, then job submission does not require a user name and password.

Setting up a job manager environment

A job manager environment consists of a job manager and the targets that it manages. The job manager targets can be deployment managers, stand-alone application server nodes that are managed by administrative agents, and host computers. Setting up a job manager environment involves creating a job manager profile and any other profiles that are needed for the environment, synchronizing the clocks on all environment computers, and then registering the targets with the job manager.

Before you begin

Install the WebSphere Application Server product.

About this task

Before you use the job manager, you must create a job manager profile and a profile for each target node that you want managed by the job manager.

Job managers are part of the flexible management environment. Job managers can manage stand-alone application server nodes that are registered to an administrative agent. Those nodes and administrative agents are also part of the flexible management environment.

Ensure that the profiles in the flexible management environment either all have security enabled or all have security disabled. Depending on your environment, you might need profiles for administrative agents, the nodes registered to the administrative agents, deployment managers, and the nodes federated with the deployment manager.

Job managers can manage Version 8 and Version 7 target nodes. A job manager can manage a node at an equal or lesser version number than the job manager. For example, a Version 8 job manager can manage Version 8 and 7 nodes. A Version 7 job manager can manage Version 7 nodes. The fix pack portion of the version number does not matter; for example, a Version 7.0.0.3 job manager can manage a node at Version 7.0.0.9, which is Version 7 with fix pack 9 installed.

Further, a job manager can manage a Version 8 or Version 7 deployment manager that has a Version 8, Version 7, or Version 6 federated node. A deployment manager that is registered with a job manager can manage a mixed version cell. Using the job manager, you can submit jobs that manage any resources in the mixed version cell, including resources on a Version 6 federated node.

Procedure

1. Determine the topology for your flexible management environment. Flexible management encompasses administrative agents and job managers.

    Determine which machines, targets, and target resources such as servers and applications to be in the flexible management environment.

    To manage stand-alone application servers, use an administrative agent on each computer where the stand-alone application servers reside. For more information, see topics on the administrative agent and Scenarios 5 in the Planning to install WebSphere Application Server topic.

    To collectively manage deployment managers and stand-alone application servers on the same or different computers, use a job manager. The stand-alone application servers must be registered with an administrative agent before you can manage them using a job manager. For more information, see Scenarios 5 and 10 in the Planning to install WebSphere Application Server topic.

2. Determine the security roles needed for your flexible management environment.

    Depending on your environment, you might need profiles for administrative agents, the nodes registered to the administrative agents, deployment managers, the nodes federated with the deployment manager, and job managers. Profiles in the flexible management environment must either all have security enabled or all have security disabled. When you create the profiles, you can specify security options, user names, and passwords.

    You must have security roles that authorize you to work with a job manager and to manage registered targets and resources on those targets. If the environment includes stand-alone application server target nodes, then you must be authorized to work with an administrative agent and its nodes.

    For more information, see the job manager security topic.

3. Create a management profile for the job manager.

    You can use the Profile Management Tool or the manageprofiles command.

    For example, in the Profile Management Tool, select the **Management** environment and click **Next**, select the **Job manager** server type, and select options that create the profile. By default, a job manager has its own administrative console, administrative security is enabled, and the console port is 9960. To disable administrative security, to specify a security certificate, or to change the default ports, use the advanced profile creation option when creating the job manager profile.

    By default, the first administrative agent profile in a product installation is named `JobMgr01` and its server name is `jobmgr`.

    For more information, see the topic on creating management profiles for job managers.

    For manageprofiles examples, see the topic on the manageprofiles command. For `-templatePath`, specify the `management` template. For `-serverType`, specify `JOB_MANAGER`.

4. Create profiles for any administrative agents and stand-alone application server nodes that you intend to have in your flexible management environment. Then, register the stand-alone application server nodes with the administrative agent.

   Stand-alone nodes are also called unfederated or base application servers. They are not managed by a deployment manager. Stand-alone application servers typically have a profile name such as AppSvr01. An administrative agent must be on the same computer as its stand-alone nodes. Registering the stand-alone nodes with the administrative agent enables the administrative agent to manage the nodes.

   Note: You must register stand-alone application servers with an administrative agent before you can register the stand-alone application servers with the job manager.

   For details on creating the profiles and registering with an administrative agent, see the topic on setting up the administrative agent environment.

5. Create profiles for any deployment managers and federated nodes that you intend to have in your flexible management environment.

   Federated nodes are managed by a deployment manager. Federated application servers typically have a profile name such as AppSvr01, however you cannot administer them individually. You must administer federated nodes using the deployment manager.

   See topics on creating cell profiles, management profiles for deployment managers, or the manageprofiles command.

6. Synchronize the clocks on all involved systems.

   If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

7. Start the job manager server.

   - Run the startServer command.

     For example, suppose the JobMgr01 profile has the server name `jobmgr`. Run the following command from the `bin` directory of the JobMgr01 profile:

   ```
   startServer jobmgr
   ```

   - Use the START command to start the job manager:

   ```
   START job_manager_proc_name,JOBNAME=server_short_name,
       ENV=cell_short_name.node_short_name.server_short_name
   ```

     If the job manager starts successfully, the message `open for e-business` displays and is written to the job manager `startServer.log` file:

   ```
   Server launched.  Waiting for initialization status.
   Server jobmgr open for e-business; process id is 1932.
   ```

     For more information, see the topic on starting and stopping the job manager.

8. Register stand-alone application server target nodes with a job manager.

   Registering stand-alone nodes with a job manager enables the job manager to administer stand-alone application server nodes.

   Note: Before you can register the stand-alone application server nodes with the job manager, the stand-alone application servers must be registered with an administrative agent. Ensure that the administrative agent version number (excluding fix pack level) is not higher than any job manager with which you are registering the administrative agent. For details on registering stand-alone application servers with an administrative agent, see the topic on setting up the administrative agent environment.

   To register stand-alone nodes with a job manager, you can use the administrative agent console or the wsadmin registerWithJobManager command.

- Use the administrative agent console to register stand-alone application server nodes.
  a. Click **System administration** > **Administrative agent**.
  b. On the Configuration tab of the Administrative agent page, click **Nodes**.
  c. On the Nodes page, select the node to register with the job manager and click **Register with Job Manager**.
  d. On the Register with Job Manager page, specify a node name, specify a job manager administrative console port number, optionally specify other parameters such as the job manager user name and password, and click **OK**.

     Note: For **Port**, specify the job manager administrative console port. If security is not enabled, specify the unsecure job manager administrative console port; the default is `9960` for an unsecure job manager administrative console port. If no port number is specified, the default secure port number `9943` is used.

- Use the wsadmin registerWithJobManager command to register stand-alone application server nodes. The command is in the ManagedNodeAgent command group.
  – Open a command window on the `bin` directory of the administrative agent profile.
  – Run a wsadmin command to start the wsadmin tool, connect the wsadmin tool to the administrative agent process, for example *adminagent*, and optionally use the Jython language:

    `wsadmin -profileName adminagent -lang jython`

  – Run the registerWithJobManager command to make a stand-alone application server a managed node of the job manager:

    `AdminTask.registerWithJobManager('[-host jobmgr_host -port jobmgr_console_port -managedNodeName application_server_node_name]')`

    jobmgr_host is the host name of the job manager. The default value is `localhost`.

    jobmgr_console_port specifies the job manager administrative console port number. If security is disabled, the default unsecure port number is `9960`. If security is enabled, optionally specify the secure port number. The default secure port number is 9943. If no port number is specified, 9943 is used.

    application_server_node_name is the host name of the stand-alone application server, for example, `myHostNode01`.

    Alternatively, you can run the *registerWithJobManager* command in interactive mode:

    `AdminTask.registerWithJobManager('-interactive')`

    If the command is successful, wsadmin displays the unique ID (UUID) of the job manager. For example:

    `'JobMgr01-JOB_MANAGER-74cdda0c-68f6-4970-a959-6f6800b9f22d'`

  For more information, see the topic on registering target nodes with the job manager using scripting.
9. Register deployment managers with the job manager.

  To register deployment managers, you can use the deployment manager console or the wsadmin registerWithJobManager command.

  Note: A deployment manager version number must not be higher than the version number of any job manager with which you are registering the deployment manager.

- Use the deployment manager administrative console to register deployment managers.
  a. Click **System administration** > **Deployment manager** > **Job managers** > **Register with Job Manager**.
  b. On the Register with Job Manager page, specify the deployment manager node name, optionally specify other parameters such as the job manager user name and password, and click **OK**.

     For **Port**, specify the job manager administrative console port number. If security is not enabled, specify the unsecure job manager administrative console port; the default is `9960`. If no port number is specified, the default secure port number `9943` is used.

- Use the wsadmin registerWithJobManager command to register deployment managers. The command is in the ManagedNodeAgent command group.
  - Open a command window on the `bin` directory of the deployment manager profile.
  - Run the wsadmin command to start the wsadmin tool and, optionally, use the Jython language:

    `wsadmin -lang jython`
  - Run the *registerWithJobManager* command to make the deployment manager a managed target node of the job manager:

```
AdminTask.registerWithJobManager('[-host jobmgr_host -port jobmgr_console_port -managedNodeName deployment_manager_node_name]')
```

    jobmgr_host is the host name of the job manager. The default value is `localhost`.

    jobmgr_console_port specifies the job manager administrative console port number. If security is not enabled, specify the unsecure job manager administrative console port; the default is `9960`. If no port number is specified, the default secure port number `9943` is used.

    deployment_manager_node_name is the host name of the deployment manager, for example, `CellManager01`.

    Alternatively, you can run the *registerWithJobManager* command in interactive mode:

    `AdminTask.registerWithJobManager('-interactive')`

    If the command is successful, wsadmin displays the unique ID (UUID) of the job manager. For example:

    `'JobMgr01-JOB_MANAGER-74cdda0c-68f6-4970-a959-6f6800b9f22d'`

  For more information, see the topic on registering targets with the job manager using scripting.

10. Register host computers with the job manager.

    A remote host target is not required to have any WebSphere Application Server products installed. There are no software requirements for this host beyond its operating system. To register remote hosts, you can use the Targets page of an administrative console or the wsadmin registerHost command.

    - Use the Targets page of the job manager console or the deployment manager console to register hosts.
      a. Click **Jobs** > **Targets** > **New Host**.
      b. On the New target page, complete the following steps:
         1) Specify the host computer name in one of the following formats:
            - Fully qualified domain name servers (DNS) host name string, such as `xmachine.manhattan.ibm.com`
            - Default short DNS host name string, such as `xmachine`
            - Numeric IP address, such as `127.1.255.3`

            The host can be the same computer on which the product is installed or a different computer.
         2) Specify the administrative user name for the host.
         3) Specify the password or private key file for the administrative user so that the job manager can access and run jobs on the host. If the host does not require a password, you can specify a null String value of "".
         4) Specify other parameters as needed.

            best-practices: Select **Save security information** and you will not need to enter the user name and password for every job manager action on the host.
         5) Click **OK**.
    - Use the wsadmin registerHost command to register hosts. The command is in the JobManagerNode command group.
      - Open a command window on the `bin` directory of the job manager profile.
      - Run the wsadmin command to start the wsadmin tool and, optionally, use the Jython language:

```
wsadmin -lang jython
```

– Run the *registerHost* command to make the host computer a target of the job manager:

```
AdminTask.registerHost('[-host host_computer -hostProps [ [osType operating_system][username administrative_user]
 [privateKeyFile key_file_path] [passphrase passphrase][saveSecurity true] ]')
```

> *host* is the computer name of the host that you want to register with the job manager. You must specify a *host* value.
>
> *hostProps* specifies properties of the host.

Table 8. registerHost -hostProps defined properties.   You can specify one or more defined properties for the registerHost command, or specify undefined properties for the command.

Property name	Property description
osType	The operating system type. Specify osType to enable the command to complete faster. This optional property determines the means for connecting with the host. Valid values are: • os390
username	A user with authority to log in to the host. This property is required.
password	The password for the given username. A value for password or privateKeyFile must be specified. If the host does not require a password, you can specify a null String value of "".
privateKeyFile	The path to the private keyfile. If you do not specify a value for password, then you must specify a value for privateKeyFile.
passphrase	A passphrase for the privateKeyFile, if needed.
saveSecurity	Specifies whether to store security properties (username, password, privateKeyFile, passphrase) with the host and used as default values for job submissions. If this property is given a value of `true`, then the security properties are stored with the host and used for subsequent job submissions to this host.

> Alternatively, you can run the *registerHost* command in interactive mode:
>
> ```
> AdminTask.registerHost('-interactive')
> ```
>
> If the command is successful, wsadmin displays the unique ID (UUID) of the host.

For more information, see topics on the New target settings or the JobManagerNode command group.

11. Verify that the targets are registered with the job manager.

    You can use an administrative console or wsadmin scripting commands to see a list of targets that are registered with the job manager.

    • In the job manager console or deployment manager console, click **Jobs** > **Targets**. The Targets page lists targets that are registered with the job manager.

    • Run the AdminConfig list command to see a list of managed targets. Run the following wsadmin scripting commands from the administrative agent `bin` directory to list stand-alone application server targets or from the deployment manager `bin` directory to list other targets.

      – To use the Jython scripting language, enter the following two commands in succession:

        ```
        wsadmin -lang jython
        ```

        ```
        print AdminConfig.list('JobManagerRegistration')
        ```

      – To use the Jacl scripting language, enter the following two commands in succession:

        ```
        wsadmin
        ```

        ```
        $AdminConfig list JobManagerRegistration
        ```

    After you verify that the targets are registered with the job manager, enter `quit` to exit the wsadmin scripting tool.

12. Ensure that the servers in your flexible management environment are running.

    In the job manager console or deployment manager console, click **Jobs** > **Target resources** > ***server_name***. On the Target resource page, a server status of `Started` shows that the server is running.

Results

The flexible management environment is set up and the job manager is configured.

What to do next

Submit jobs using the job manager.

Starting and stopping the job manager

In your flexible management environment, you can start the job manager by using the startServer command. You can stop the job manager by using the stopServer command.

Before you begin

Before you can start or stop the job manager, you must first install the product.

About this task

Start the job manager so that you can administer jobs for large numbers of unfederated application servers and deployment managers. Stop the job manager as needed, such as when migrating to a new version of the product, when uninstalling the product, and so on.

Procedure

- Start the job manager.

  Use one of these methods to start the job manager:

  - Use the startServer command:

    ```
    startServer <job_manager>
    ```

    where *job_manager* is name of the job manager that you want to start.

  - Use the START command to start the job manager:

    ```
    START job_manager_proc_name,JOBNAME=server_short_name,
          ENV=cell_short_name.node_short_name.server_short_name
    ```

- Stop the job manager.

  Use one of these methods to stop the job manager:

  - Use the Job manager administrative console.
    1. Click **System Administration** > **Job manager**.
    2. Click **Stop** on the Configuration tab.
  - Use the stopServer command:

    ```
    stopServer <job_manager>
    ```

    where *job_manager* is name of the job manager that you want to stop.

Results

You have started the job manager and have optionally stopped it.

What to do next

Administer jobs using the job manager. You can do such tasks as submit jobs, check the status of jobs, view nodes and node resources, or administer node groups.

Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root

Refers to the top directory for a WebSphere Application Server node.

The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.

The default varies based on node type. Common defaults are `configuration_root`/`AppServer` and `configuration_root`/`DeploymentManager`.

configuration_root

Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.

The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.

The default is `/wasv8config/`*cell_name*`/`*node_name*.

plug-ins_root

Refers to the installation root directory for Web Server Plug-ins.

profile_root

Refers to the home directory for a particular instantiated WebSphere Application Server profile.

Corresponding product variables are *server.root* and *user.install.root.*

In general, this is the same as `app_server_root`/`profiles`/*profile_name*. On z/OS, this will always be `app_server_root`/`profiles`/`default` because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root

Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.

The corresponding product variable is *smpe.install.root*.

The default is `/usr/lpp/zWebSphere/V8R0`.

Configuring job managers

In a flexible management environment, you can specify settings such as the default job expiration, the job manager web address, and the mail provider Java Naming and Directory Interface (JNDI) name for the job manager. You can view job manager properties such as the process ID and the state of the job manager.

Before you begin

Before you can configure the job manager, you must have started the job manager.

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

About this task

When you create a job manager profile, a job manager is created. You can run the job manager with its default settings. However, you can change the job manager configuration settings, such as the expiration

time of jobs, the maximum number of database results to display, and so on.

Procedure

1. Click **System administration > Job manager** from the navigation tree of the job manager administrative console to access the settings page for a job manager.

2. Configure the job manager by clicking a property and specifying settings.

   a. Optionally change the default job expiration by specifying an integer value.

   b. Optionally change the maximum number of rows that a query can return to the job manager by specifying an integer value on the **Maximum database results** setting.

   c. Optionally change the web address of the job manager that the administrative agent uses to retrieve jobs by specifying a web address for the **Job manager URL** setting.

   The web address is used only when the job manager is configured as a proxy server.

   d. Optionally specify an email address on the email sender's address setting.

   The email address that you specify is the email address of the sender of the notification message that the job manager provides when jobs have completed.

   e. Optionally select the **Start components as needed** setting to start components dynamically as needed for applications.

   f. Optionally stop the job manager by clicking **Stop**.

   Selecting this option stops the job manager and its console.

3. If you changed any of the settings, click **Apply**, and then **OK**.

Results

You configured the job manager with options that you selected.

What to do next

You can do other job management tasks such as administer node groups, view nodes, submit jobs, and view node resources.

Job manager settings

Use this page to configure the job manager server and view its properties. You can specify the default job expiration, the job manager web address, and the mail session Java Naming and Directory Interface (JNDI) name.

To view this administrative console page, click **System administration** > **Job manager**.

Name:

Specifies the job manager server name. The name is read-only.

Default job expiration:

Specifies the default job expiration time in days.

Data type	Integer
Default	60

Maximum database results:

Specifies the maximum number of records that can be retrieved during a job manager find operation. Find operations might be for records on jobs, nodes, and node resources.

This maximum number of records can be reduced by the maximum results setting on a find operation. For example, assume that you specify the maximum results to display for finding nodes at 50, but maximum database results is set to 10000. If you have 20000 jobs, the find operation finds 50 nodes.

Data type	Integer
Default	10000

Job manager URL:

Specifies the web address of the job manager that the administrative agent uses to fetch jobs.

The web address that you specify is used only when the job manager is configured as a proxy server. The web address overrides the default web address. If you modify the web address, you must reregister the nodes with the job manager. The change affects only the nodes previously registered.

Data type	String
Default	http://*host*:*port*/otis/OMADMServlet
	The *host* and *port* are those of the job manager unless you use a web server. In that case, change the *host* and *port* to that of the web server.

Mail session JNDI name:

Specifies an optional mail session JNDI name to be used for email notifications on job completion.

Data type	String
Default	None

Email sender's address: Specifies the email address of the sender of the notification message that the job manager provides when jobs have completed. This setting is required if you specify a JNDI mail session on the Mail session JNDI name setting.

Start components as needed:

Specifies whether to start the server components as they are needed for applications that run on this server.

Select this property if you want the server components started as they are needed.

When this property is not selected, all the server components are started during the startup process. Therefore, selecting this property typically results in improved startup time because fewer components are started during the startup process.

gotcha: If you are running other WebSphere products on top of this product, make sure that those other products support this functionality before you select this property.

Process ID:

Specifies the read-only process ID of the job manager.

Cell name:

Specifies the read-only cell name of the job manager.

Node name:

Specifies the read-only node name of the job manager.

State:

Specifies the read-only state of the job manager, such as started or stopped.

Viewing target information using the job manager

In a flexible management environment, you can view targets with their version numbers based on the results of the Find option, and view target resources for targets that you select. You can also view the properties and property values for a particular target.

Before you begin

Before you can view information about targets, you must have registered at least one target with the job manager.

To display resources for selected targets, your ID must be authorized for the monitor role.

About this task

The first time you access the Target collection page, no targets are listed. You must enter parameters for the Find option to obtain a list of targets based on the parameter information that you provide. The next time you select **Jobs** > **Targets**, a list of targets are displayed based on the parameters you last specified on the Find option for this administrative console page. You can then optionally modify the Find option criteria to display a different set of targets. After at least one target is displayed, you can view information about the targets. You can display all targets by setting the string for the target name on the Find option to *.

The Target collection page is in the administrative console for a job manager and for a deployment manager.

Procedure

- Optionally use the Find option to display a set of targets.

  If no targets are displayed, you must use the Find option to display targets based on the parameter information that you enter.

  1. Click **Jobs** > **Targets** in the administrative console navigation.
  2. If you want to run the Find operation on specific parameters, specify a valid operator and a text string. You can use the asterisk (*) character on the target name, job type, and unique identifier parameters to represent unspecified characters in conjunction with the characters that you specify. The target Find option has some advanced Find options that you can view by selecting the plus (+) character. You can enter partial search strings for the advanced find options as well.
  3. Click **Find**.

  The list of targets along with their version number is displayed in the collection table.

- Optionally display resources for selected targets.

  1. Click **Jobs** > **Targets** in the administrative console navigation.
  2. Select the check box next to targets for which you want to display resources.
  3. Click **Display resources**.
  4. Choose the type of resources to display.

  The resources are displayed for the targets that you selected.

- Optionally display properties and property values for a particular target by clicking **Jobs** > **Targets** > *target_name* in an administrative console.

Results

Depending on the tasks that you completed, you might have viewed targets with their version numbers based on the results of the Find option, viewed target resources for targets that you selected, or viewed the properties and property values for a particular target.

What to do next

You can continue to view targets and do other job management tasks such as submit jobs, create target groups for job submission, and view target resources.

Target collection for Find results

Use this page to find targets for jobs. The page displays target names and version numbers based on the results of the Find option. You can additionally display target resources for targets that you select.

To view this administrative console page, click **Jobs** > **Targets**.

Table 9. Button descriptions. Select a button to register a remote target host with a job manager, to display information about target resources, or to unregister a host.

Button	Description
New Host	Displays a page that you can use to register a remote target host with a job manager. To register a host, you must be authorized for the administrator role.
Display Resources	Displays the available resources of selected targets. Example values are **All**, **Application**, **Server**, and **Cluster**. The choices in the list depend on the targets that are registered. For example, if you do not have a deployment manager registered to a job manager, clusters are not in the list. To display target resources, you must be authorized for the monitor role.
Delete Host	Unregisters a target host from the job manager. You must be authorized for the administrator role.

Find:

Use the Find option to determine the targets to display. By default, the product searches all targets. After you click **Find**, the results are displayed in the table. The table follows the Find and Preferences options. Click **Reset** to assign the parameters the default values.

Table 10. Find parameters. Specify Find parameters to limit the search for targets.

Parameter name	Operators	Search strings
Target type	Valid operators include `All`, `Host`, and `Node`. By default, all target types are searched.	Not applicable

Table 10. Find parameters (continued).   Specify Find parameters to limit the search for targets.

Parameter name	Operators	Search strings
Target name	Valid operators are = (equal to) and != (not equal to). The default operator is =.	The search string specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, to find all jobs with a target name that starts with Node, set the target name parameter to Node*. To search for an exact match for multiple items, include comma-separated items. For example, to search on two target names, specify Node1, Node2. When you search for more than one item, you cannot use the asterisk. **Example:** If the targets are AppSvr01, AppSvr02, AppSvr03, and Test01, you can specify the = operator and the App* search string for the **Target name** parameter, then click **Find**. The target names displayed are AppSvr01, AppSvr02, and AppSvr03.
Job type	Valid operators are = (equal to), != (not equal to), is null, and is not null. The default operator is =.	
Unique identifier		
Advanced find options Lists searchable target properties, such as **username**, **host**, **password**, and **Operating system**.		

Table 10. Find parameters (continued).   Specify Find parameters to limit the search for targets.

Parameter name	Operators	Search strings
Resources Enables you to further narrow the list of targets to find targets that do or do not have specific resources. For example, to find targets that have a server named `server1`.	Specify query conditions to find resources. 1. Specify **With** (button shows **With** | Without), or click the button to toggle to **Without** (button shows With | **Without**). The button specifies whether to search for resources that meet the query conditions (**With**), or do not meet the query conditions (**Without**). 2. Select a resource type, such as `application`, `server`, `cluster`, `Installation Manager`, `Package`, or `Profile`. The list of resource types depends upon whether the registered targets are stand-alone application servers, deployment managers, or hosts. 3. Select a property for the specified resource type. 4. Select an operator. Valid operators are = (equal to), != (not equal to), < (less than), > (greater than), <= (less than or equal to), >= (greater than or equal to), `is null`, and `is not null`. The default operator is =. 5. Specify a value for the resource property; for example, an asterisk (*). To add more conditions to the first query, click , the Add icon. If you click , you cannot change the **With | Without** toggle button. Click , the Delete icon, to remove an added condition. To narrow the search and add a query that changes the **With | Without** toggle button, click ⊕ , the Add Search Clause icon below the **With | Without** toggle button. In the dialog that opens, select **Perform subset search** to perform a subset search that refines the results of the first query and click **OK**. If you want the query to simply use a different **With | Without** selection and not perform a subset search, then do not select **Perform subset search**. Then, specify another query condition.	The value that you specify for the resource property is like a search string. You can use an asterisk (*) to designate a partial string. You can also specify comma-separated items to search on multiple values. When you search for more than one item, do not use the asterisk. **Example:** To find targets that have stopped applications, specify the following query conditions: 1. **With** 2. `application` resource type 3. `status` resource property 4. = (equal to) operator 5. `STOPPED` property value Then click **Find**. The table lists targets that have stopped applications. To narrow the search to targets that have stopped applications and that do not have a server named `server1`, specify the previous query. Then, click ⊕ , the Add Search Clause icon below the **With | Without** button. In the dialog that opens, select **Perform subset search** and click **OK**. Another query row is displayed, with parentheses around the **With | Without** button. The parentheses indicate the query is a subset search; only the targets resulting from the previous query are searched. In the new query row, specify the following query conditions: 1. **Without** 2. `Server` resource type 3. `serverName` resource property 4. = (equal to) operator 5. `server1` property value Click **Find**.
Maximum results	Not applicable	The search string specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration. The default value is `50`.

Target name: Specifies the names of targets that are found as a result of the Find option.

Version:

Specifies the name and version number of the product on which the target runs.

The product version is the version of a WebSphere Application Server.

The base edition of WebSphere Application Server (base) is listed in the version column as `Base`. The WebSphere Application Server, Network Deployment product is listed in the version column as `ND`.

The product in the version column indicates the product that you used to create the profile, not the type of profile that you installed. For example, if you use the WebSphere Application Server, Network Deployment product to install a profile type of application server, the version column indicates `ND`.

New target settings:

Use this page to register a remote computer as a target with the job manager. To register a target with the job manager, you must be authorized for the administrator role.

To view this administrative console page, click **Jobs** > **Targets** > **New Host**.

Host name: Specifies the name of the computer that you want to register as a target with the job manager. The target host must be a remote computer.

Specify the host name value in one of the following formats:
- Fully qualified domain name servers (DNS) host name string, such as `xmachine.manhattan.ibm.com`
- Default short DNS host name string, such as `xmachine`
- Numeric IP address, such as `127.1.255.3`

Operating system: Specifies the operating system of the target host. The default value is **Any** for any operating system. Other supported values are **AIX**, **HPUX**, **IBM i**, **IBM zOS**, **Linux**, **Solaris**, and **Windows**. Specifying the correct operating system value enables operations to run faster.

Administrative user with install authority: Specifies a user name that provides access to the target host. The user must have the authority to log in to the target host.

Password authentication: Specifies the password for the user name on the target host. Specify the same password for the **Password** and **Confirm password** fields.

If the target host does not require a password, leave the fields blank.

Public-private key authentication: Specifies the full path to the keystore and, if required for the keystore, the passphrase.

To use public-private key authentication, first generate a pair of keys using a key generation tool such as ssh-keygen. Next, add the public key to the `authorized_keys` file of the user on the target host. Then, on this page, specify the user name, fully qualified private key file, and optionally the fully qualified passphase.

Save security information: Specifies whether to save the security information that is entered on this page. The host uses any saved properties as defaults for subsequent job submissions to the host.

The default is not to save security information.

Installation Manager data location path(s): Specifies one or more paths on the host to use as the Installation Manager data location. The default is not to specify a data location path.

Enter the fully qualified path of the Installation Manager data locations. You can enter multiple paths separated by a semi-colon. This property is useful if you have non-default Installation Manager data

locations on your targets. For example, if you specified data locations when installing Installation Manager, the inventory job might not find the non-default data locations. Therefore, you might need to specify additional data locations when registering a host. If an unsupported data location is specified, it is not saved. If the specified data location can be detected by the inventory job, it is not saved.

Note: Ensure that the path is not longer than 256 characters. If the path is longer than 256 characters, the product truncates the path. The product does not warn you if the path is longer than 256 characters. You must check the length of the path. Truncated paths are not valid and, after validation checking, are removed from this field.

You can use the find data location job to search for data locations on the system. The find data location job automatically updates this property. For example:

```
AdminTask.registerHost('[-host hostname -hostProps [[imDataLocations datalocation1; datalocation2][password ****][saveSecurity true][username username]]]')
```

Target property settings:

Use this page to view the properties and property values for a particular target.

To view this administrative console page, click **Jobs** > **Targets** > ***managed_target_name***.

Target name: Specifies the target name that you selected from the target collection.

Name: Specifies the name of each property for the target. The list of names varies from one runtime environment to another and is read-only.

Value: Specifies a value of the specified name. The value is read-only.

Viewing target resource information using the job manager

In a flexible management environment, you can view server, application, cluster, and host resources associated with targets and target groups registered to the job manager. You can also view the status of specific resources at each target, and view properties for a particular target resource as a name-value pair.

Before you begin

Before you can view information about target resources, you must have registered at least one target with the job manager.

About this task

The type of resources you can view depends on your topology. For example, you cannot view clusters if a deployment manager that you registered to the job manager does not have a defined cluster.

The first time you access the Target resource collection page, no target resources are listed. You must enter parameters for the Find option to obtain a list of target resources based on the parameter information that you provide. The next time you select **Jobs** > **Target resources**, a list of target resources are displayed based on the parameters you last specified on the Find option for this console page. You can then optionally modify the Find option criteria to display a different set of target resources. After at least one target resource displays, you can view information about the target resources.

The Target resource collection page is in the administrative console for a job manager and for a deployment manager.

Procedure

- Optionally use the Find option to display a set of target resources.

If no target resources are displayed, you must use the Find option to display target resources based on the parameter information that you enter.

1. Click **Jobs** > **Target resources** in the administrative console navigation.

2. Choose the resource type in the **Type** list.

3. If you want to do a Find operation on specific parameters, specify a valid operator and a text string.

4. Click **Find**.

The list of target resources along with the number of target resources for a given target resource in the list and the target for the resource are displayed.

- Optionally display the status of specific target resources for each target.

  1. Click **Jobs** > **Target resources** > *resource* in an administrative console.

  The resource ID, target name, and resource status are displayed.

- Optionally display the properties of a target resource by clicking **Jobs** > **Target resources** > *resource* > *resource_ID* in an administrative console.

Results

Depending on the tasks that you completed, you might have viewed server, application, and cluster resources associated with targets and target groups registered to the job manager. You might have also viewed the status of specific resources at each target, and viewed properties for a particular target resource as a name-value pair.

What to do next

You can continue to view target resources and do other job management tasks such as submit jobs, create target groups for job submission, and view targets.

Target resources collection

Use this page to display resources on targets or target groups such as servers, applications, cluster, and profiles.

To view this administrative console page, click **Jobs** > **Target resources**.

Find:

You can use the Find option to determine the resources to display. After you click **Find**, the Find results are displayed in the table that follows the Find and Preference options. Click **Reset** to assign the parameters the default values.

Table 11. Find results. *Specify Find parameters to limit the search for target resources.*

Parameter names	Operators	Search strings
Type Specifies the type of resource.	For target nodes, valid operators include `All`, `Application`, `Server`, and `Cluster`. For target hosts, valid operators include `All`, `Installation Manager`, `Package`, `Profile` and `Package Group`.	Not applicable

Table 11. Find results (continued). Specify Find parameters to limit the search for target resources.

Parameter names	Operators	Search strings
Resource name Specifies the name of an instance of a resource type. For example, a server resource could have a resource name of server1.	Valid operators include = (equal to), != (not equal to), `is null`, and `is not null`.	Specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). You can search for an exact match for multiple items by including comma-separated items. When you search on more than one item, you cannot use the asterisk. **Example:** If the resource names are server1, server2, application01, and application03, you can select the `!=` operator for the resource name parameter and a resource name of `server*` for the search string. The results of the find operation display the application01 and application03 resource information.
Status Specifies the status of the resource. Valid values for the status vary because different resources have different status. Examples of status for a server are started and stopped.		
Target name Specifies the name of the target on which the resource resides.	Valid operators include = (equal to) and != (not equal to).	
Group name Specifies the name of the target group.	The valid operator is = (equal to).	
Context Specifies topology information of the resource, such as cell/application.	Valid operators include = (equal to) and != (not equal to).	
Maximum results The maximum number of results that display after the find option completes.	Not applicable	Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration. The default is `50`.

Resources: Specifies the ID of the resource.

Quantity: Specifies the number of targets with the same resource name.

Target name: Specifies the name of the target for the resource. When the resource is defined on multiple targets, the target name is displayed as `multiple`, and the quantity is greater than one. When you click on the resource name, you can see the details of the targets where this resource resides.

Target resources for targets collection:

Use this page to display the status of specific resources at each target, and includes the resource ID, the name of the target, and the resource status.

To view this administrative console page, click **Jobs** > **Target resources** > *resource_ID*.

Resource ID: Specifies a unique identifier for the resource in the form of context and values. For example, servers display as
- `server/`*server_name* for targets registered with an administrative agent
- `node/`*node_name*`/server/`*server_name* for targets that are deployment managers

Target name: Specifies the target on which the resource is located.

Status: Specifies the status of the resource. Valid values for the status vary because different resources have different status. Examples of status for a server are started and stopped. Status on **Jobs** > **Status** administrative console pages is updated only when a status job or an inventory job for the target containing the resource completes successfully. The status is not updated on every polling interval. You can view up-to-date resource status in the deployment manager console or the administrative agent console for the target that you want to view.

Target resource properties:

Use this page to display a read-only view of the properties for a particular target resource as a name-value pair. Updates that you make to the target resource properties must be done at the administrative agent.

To view this administrative console page, click **Jobs** > **Target resources** > *resource_ID* > *resource_ID*.

Name:

Specifies the name of the property. The property name is unique.

Value:

Specifies the value paired with the specified name.

Submitting jobs

In a flexible management environment, you can submit jobs to remote targets to manage applications, modify the product configuration on remote machines, or do a general purpose task such as run a script. You can specify when the jobs start, whether they are recurring, and when they expire.

Before you begin

Before you can submit a job, you must have registered at least one target with the job manager. A target can be an application server node that was first registered with an administrative agent, a deployment manager node, or a host computer.

Start the job manager and the targets. If a target is a stand-alone application server, also start the administrative agent.

Your ID for the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. If you do not provide a user name and password in the job parameters, the credentials for the job submitter at the job manager are used for this purpose. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply to all the job targets.

You can simplify administration of multiple targets by submitting jobs against groups of targets. Each group of targets represents a group of targets. Before you can submit a job for a group of targets, you must have created the group of targets.

Job manager functionality exists in a job manager and in a deployment manager. For simplicity, this documentation refers to the functionality as the *job manager*.

About this task

You can use the administrative console of the job manager or deployment manager to submit jobs to do tasks such as manage applications, modify the product configuration on remote workstations, or do general-purpose tasks such as run a script. To complete the job submission, choose the type of job, choose the targets on which you want the job to run, specify the job parameters that are specific to the job type, schedule the job, review the summary, and submit the job.

The topics in this section describe how to submit jobs using a job manager console or a deployment manager console. Instead of using a console, you can submit jobs from the command line using the wsadmin submitJob command in AdministrativeJobs command group. See the topic on administrative job types.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the job manager console or deployment manager console.
2. Choose the job type.
   a. Select the job type from the list.

      The list of job types varies based on the targets that you have registered with the job manager. The values displayed in the list are retrieved from the getJobTypes and getJobTypeMetadata commands of the AdminTask object. You can have job types that manage applications, modify the product configuration on remote machines, or do general-purpose tasks such as run a script.

      The following job types exist:
      - Collect file
      - Configure properties
      - Create application server
      - Create cluster
      - Create cluster member
      - Create proxy server
      - Delete application server
      - Delete cluster
      - Delete cluster member
      - Delete proxy server
      - Distribute file
      - Install application
      - Install IBM Installation Manager on a remote host
      - Install SSH Public Key
      - Inventory
      - Manage offerings
      - Manage profiles
      - Remove file
      - Run command on remote host
      - Run wsadmin script
      - Start application
      - Start cluster
      - Start server
      - Status
      - Stop application

- Stop cluster
- Stop server
- Test connection
- Uninstall application
- Update application
- Uninstall IBM Installation Manager
- Update IBM Installation Manager

b. Optionally specify a description of the job.

The description is a string that can be up to 256 characters. The default description is the job type. You can change or add to the default description. The description is useful when using the Find option to view existing jobs.

c. Click **Next**.

3. Choose the job targets.

You are determining the targets on which you want the job to run.

a. Select a group of targets from the list, or select **Target name**.

Only groups of targets that are valid for the job type that you selected are displayed in the list of groups of targets.

b. If you selected **Target name**, then enter a target name, and click **Add**, or generate a list of targets by using the **Find** option.

Target name that you enter
If you enter a target name, it must be a target that has been registered to the job manager. The target name is validated when you click **Next**.

List of target names
1) Click **Find**.

The Find targets page is displayed.

2) For **Target type**, select **All**, **Host**, or **Node**. The default value is **All**.

3) If you want to run the Find operation on specific keywords, specify a valid operator and a text string.

The list of keywords is dynamic. Valid operators are = (equal to), != (not equal to), `is null`, and `is not null`. The text string can be complete or partial and can contain an asterisk (*) to include variable or unknown characters.

4) Click **Find**.

The results are displayed in the Excluded targets list and are selected.

5) Move targets that you want to target from the Excluded targets list to the Chosen targets list.

- To move specific targets from the Excluded targets list to the Chosen targets list, select targets in the Excluded targets list and click >.
- To move specific targets from the Chosen targets list to the Excluded targets list, select targets in the Chosen targets list and click <.

6) After you have a list of the wanted targets in the Chosen targets list, click **OK**.

The targets display on the Choose job targets page.

c. If the target requires authentication, specify a user name and password so that the target can run the job.

For example, to access a target host, you typically specify values for **User name** and **Password authentication**. The user name and password are the login values for the host. If the target host does not require a password, leave the fields blank.

If you want to use public-private key authentication, select **Public-private key authentication** and then specify the full path to the keystore and, if required for the keystore, the passphrase.

> **best-practices:** To use public-private key authentication, first generate a pair of keys using a key generation tool such as ssh-keygen. Next, add the public key to the `authorized_keys` file of the user on the target host. Then, on this Choose job targets page, specify the user name, fully qualified private key file, and optionally the fully qualified passphase.

    d. Click **Next**.

4. Specify the job parameters.

    The list of job parameters is dynamic and based on the job type. For example, if the job type is to install an application, specify the application name, the location of the application to install, and optionally the name of the server where the system installs the application.

    When you submit a job to multiple targets, the parameter values must apply to all the job targets.

    The following table describes the types of parameters.

Parameter Type	Description
String	You can enter text for the appropriate parameters. The text is not validated until the job is submitted.
Target resource	You can select a target resource. The Find option is available for you to search for the resource, depending on the job type that you selected in the first step.

    a. Optionally click **Find** if it is available.

       The Find target resources page is displayed.

    b. If you want to run the Find operation on specific keywords, specify a valid operator and a text string.

       The list of keywords is dynamic. Valid operators are = (equal to), != (not equal to), `is null`, and `is not null`. The text string can be complete or partial and can contain an asterisk (*) to include variable or unknown characters.

    c. Click **Find**.

       The results are displayed in the Available resources common to all selected endpoints list.

    d. Click **OK** to save the results and return to the page on specifying job parameters.

    e. Click **Next**.

5. Schedule the job.

    The times and dates that you specify are relative to the job manager.

    a. Optionally specify one or more email addresses where notifications are sent when the job is done.

       If you specify multiple email addresses, separate them with commas. The email addresses are saved in your console preferences. Each email address is validated for format errors.

    b. Select when the job is available for submission.

       You can submit the job to be available now, or specify a time and date that the job is retrieved from the job manager.

    c. Select the job expiration.

       The job expiration is the time at which the job is no longer available for targets to run. You can use the default expiration, specify a time and date for the job expiration, or specify an amount of time in which the job expires. The default expiration is defined on the Job manager configuration page.

    d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

    e. Click **Next**.

6. Review the summary, and submit the job.

    a. If you want to change the options, click **Previous** until you reach the page that you want to change. Make the correction on that page, and then proceed through the pages until you review the Summary and submit the job page.

b. When you are satisfied with the options, click **Finish** to submit the job.

The Job status collection page is displayed where only the status for the job that you submitted is displayed.

Results

After you submit a job, the job might not be run immediately. The job manager queues submitted jobs. The administrative agents and deployment managers poll the job manager for jobs when they are online, based on their configured polling intervals. The default polling interval is 30 seconds. It takes at least two polling cycles for administrative agents and deployment managers to retrieve jobs and then return results to the job manager. Depending on how long it takes for the target to process the job, it might take more cycles to complete the job.

What to do next

After you submit a job, the Job status page shows a unique job ID; for example, 1227633809125763341. You can use the job ID to query, suspend, resume, or delete the job. When you click a job ID, you see the specific properties of that job, including activation and expiration time of the job and its status. If you click the job status link, you see the job history for each job target. Click the status refresh icon to refresh the displayed status.

You can check the state and status of a job using the job manager console or a wsadmin command.

The job state shows where the job is in the execution process from the job manager perspective. Table 1 lists the job states.

Table 12. Job states. The state indicates whether the job is active.

Job states	Description
Pending	You submitted the job, but the job is not available yet to be run on the targets.
Active	One or more targets have started running the job.
Expired	The job has expired. If a target started to run the job before it expired, the job continues running. After a job expires, a target cannot start running the job.
Suspended	The job suspended operation. If a target started to run the job before it is suspended, the job continues running. After a job is suspended, a target cannot start running the job.

The job status shows a history of the job processing on a managed target. A typical job history is for the status to progress from Distributed to In progress to Succeeded. Table 2 shows the job status values.

Table 13. Job status descriptions. The status indicates whether the job completed successfully.

Job status	Description
Not attempted	The target has not received the job. The status is NOT_ATTEMPTED.
Distributed	The target has received the job. The status is DISTRIBUTED.
In progress	The target is running the job concurrently with other jobs. The status is ASYNC_IN_PROGRESS.
Failed	The job failed and is no longer running. The status is FAILED.
Rejected	The target rejected the job because, for example, the target does not support the job type. The status is REJECTED.
Succeeded	Job completed successfully. The status is SUCCEEDED.
Partially succeeded	Applies only to startCluster and stopCluster jobs where the cluster has multiple cluster members and to startApplication and stopApplication jobs where the application is installed on multiple targets. If only some cluster members are started or stopped or the application does not start on all application targets, the status of the job is PARTIALLY_SUCCEEDED.

By default, submitted jobs remain active for one day (24 hours). An active job is a running Java process that consumes machine resources. Delete jobs that you no longer need. You can use the job manager console Job status page. Click **Jobs** > **Status**, select the jobs, and click **Delete**.

Submitting jobs to manage servers

In a flexible management environment, you can submit jobs to create and administer servers on managed targets of the job manager. The servers can be a stand-alone server or a federated node of a deployment manager.

Before you begin

Before submitting a job, start the job manager and the targets. If a target is a stand-alone application server, also start the administrative agent.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must be applicable to all the job targets.

About this task

The topics in this section describe how to create and administer servers by running jobs in the job manager console or the deployment manager console.

The jobs that you can run depend on the jobs supported by managed targets and your security credentials. To run jobs that administer clusters, a deployment manager target must be registered with the job manager. To run jobs that administer proxy servers, a target that supports proxy servers must be registered with the job manager.

Instead of using a console, you can run wsadmin commands in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure
- Run the Create application server job.
- Run the Create cluster job.
- Run the Create cluster member job.
- Run the Create proxy server job.
- Run the Configure properties job to apply properties files to application servers.
- Run the Start cluster job.
- Run the Start server job.
- Run the Stop cluster job.
- Run the Stop server job.
- Run the Delete application server job.
- Run the Delete cluster job.
- Run the Delete cluster member job.
- Run the Delete proxy server job.

What to do next

On the Job status page, click the ID of the job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, click **Jobs** > **Target resources** to see the server in the list of resources.

Creating application servers using the job manager:

In a flexible management environment, you can submit the **Create application server** job to create a server. The job can create an application server on a stand-alone node or on a federated node of a deployment manager.

Before you begin

Start the job manager and the targets. If a target is a stand-alone application server, also start the administrative agent.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must be applicable to all the job targets.

About this task

You can use the administrative console of the job manager or the deployment manager to create an application server on one or more managed targets. From the console, choose the **Create application server** job, specify server and job options, review the summary, and submit the job.

Instead of using a console, you can run the createApplicationServer job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure
1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to create a server.
    a. Select the **Create application server** job type from the list.
    b. Optionally specify a description of the job.
    c. Click **Next**.
3. Choose the job targets on which to add the server.
    a. Select a group of targets from the list, or select **Target names**.

       Only groups of targets that are valid for the job type that you selected are displayed in the list of target groups.
    b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
    c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
    d. Click **Next**.
4. Specify parameters for the create server job.
    a. For **Server name**, specify a server name that is unique on the targets.

       To see the names of existing servers on the targets, click **Find** on the Specify job parameters page. On the Find node resources page, specify the targets and click **Find**. For example, suppose a managed target, `nodeA`, has one server, `server1`. Specify a name that is unique to `nodeA`:

       `server2`
    b. If the target is in a WebSphere Application Server, Network Deployment cell, for **Node name** specify the target name in the cell on which to create the server.

You do not have to specify the target name for a base (stand-alone) node. For example, suppose the managed node, `nodeA`, is a federated node in a WebSphere Application Server, Network Deployment cell. Specify the node name:

`nodeA`

   c. To use a template other than the default server template, specify the name and location of the template to use.

   d. To use a specific port number and not use the unique port number generated by the product, deselect **Generate unique ports**.

   e. If you are creating a server on a z/OS target, optionally specify short names and bit mode.

      If you do not specify values, the product generates unique short names and uses the default bit mode.

   f. Click **Next**.

5. Schedule the job.

   The times and dates that you specify are relative to the job manager.

   a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

      If you specify multiple email addresses, separate them with commas.

   b. Select when the job is available for submission.

   c. Select the job expiration.

   d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

   e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to create the application server.

What to do next

On the Job status page, click the ID of the create application server job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, click **Jobs** > **Target resources** to see the new application server in the list of resources.

After creating the server, you can run a job that starts the server.

Creating clusters using the job manager:

In a flexible management environment, you can submit the **Create cluster** job. A cluster is a set of servers that you manage together to balance workload. The job creates a cluster that runs on a deployment manager node. The cluster becomes a managed resource of the job manager.

Before you begin

Start the job manager, the deployment manager, and the federated node to which you want to add a cluster.

If the deployment manager is not a managed node of the job manager, register the deployment manager with the job manager. Registering enables the job manager to manage the deployment manager and its federated nodes. To submit the **Create cluster** job, a deployment manager node must be a managed node of the job manager.

Determine how you want to configure the cluster:
- Review topics on clusters and workload management, especially the information about setting cluster weights.
- Determine whether you want a cluster to group application servers, proxy servers, or on-demand routers.
- Determine the node to which to add the first cluster member. The target node must be a managed node of the job manager.
- For a cluster of servers that spans multiple systems in a Sysplex and has stateful session beans with an activation policy of Transaction deployed in them, define the passivation directory on a hierarchical file system (HFS) that is shared across the multiple systems in the Sysplex.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply to all the job targets.

About this task

A cluster enables you to manage a group of application servers as a single unit, and distribute client requests among the application servers that are members of the cluster. You might want to create a cluster if you need to:
- Balance your client requests across multiple application servers.
- Provide a highly available environment for your applications.

You can use the administrative console of the job manager or the deployment manager to create a cluster on one or more managed nodes. From the console, choose the **Create cluster** job, specify job options, review the summary, and submit the job. This topic describes how to use the job manager console or the deployment manager console to submit the job.

Instead of using a console, you can run the createCluster job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to create a cluster.
   a. Select the **Create cluster** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets on which to add the cluster.
   Select one or more deployment manager managed nodes to which you can add clusters.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the create cluster job.

a. For **Cluster name**, specify a cluster name that is unique on the targets.

To see the names of existing clusters on the targets, click **Jobs** > **Target resources**. The Target resources page lists the clusters, servers, and applications on managed targets. After you determine what cluster names are already used, return to the Specify job parameters page and specify a unique name for the cluster that you want to create. For example, suppose a managed deployment manager node, `myNode01`, has one cluster, `cluster1`. Specify a name that is unique to `myNode01`:

`cluster2`

b. For , if you do not want to enable node-scoped routing optimization for the cluster, deselect **Prefer local** (`false`). The default value is selected (`true`).

When enabled, node-scoped routing optimization routes requests to the target on which the cluster resides.

c. For **Cluster type**, specify the type of server cluster to create.

If you want to create a cluster to group application servers, leave the field empty. The default type is `APPLICATION_SERVER`.

Otherwise, specify `PROXY_SERVER` or `ONDEMAND_ROUTER`.

d. If you are creating a cluster on a z/OS target, optionally specify a cluster short name.

If you do not specify a value, the product generates a unique short name.

e. Optionally, expand **Additional job parameters** and specify values that further define the cluster.

Table 14. Additional Create cluster job parameters.   Specify job parameters as needed.

Parameter name	Description
Create domain	To create a replication domain with a name set to the name of the new cluster, select **Create domain** (`true`). The product uses the replication domain for HTTP session data replication. The default value is clear (`false`).
Server node	If you want to convert an existing server to the first member of the cluster, specify the name of a managed node that has the existing server.
Server name	If you want to convert an existing server to the first member of the cluster, specify the name of the existing server that resides on the node specified for **Server node**.
Member weight	If you want the new cluster member to have a weight value other than 2, the default, specify a different weight value. A valid value is a number between 0 and 100. The weight controls the amount of work directed to a server. If the weight is greater than the weight assigned to other cluster members, the server receives a larger share of the workload.
Node group	If you want all cluster member to belong to a node group, specify the name of the node group.
Replication entry	If **Create domain** is selected, optionally specify `true` to create a replicator entry for this member in the cluster replication domain. The default value is `false`.

f. Click **Next**.

5. Schedule the job.

The times and dates that you specify are relative to the job manager.

a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

If you specify multiple email addresses, separate them with commas.

b. Select when the job is available for submission.

c. Select the job expiration.

d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to create a cluster.

What to do next

On the Job status page, click the ID of the create cluster job and view the job status. Click the status
refresh icon [image] to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error
condition, and submit the job again.

If the job is successful, click **Jobs** > **Target resources** to see the new cluster in the list of resources. The
status of the cluster is `Stopped`.

After creating the cluster, you can run a job that creates a cluster member. You cannot start the cluster
unless it has at least one cluster member.

Creating cluster members using the job manager:

In a flexible management environment, you can submit the **Create cluster member** job to create a server
to add to a cluster. A cluster is a set of servers that you manage together to balance workload. The cluster
can group application servers, proxy servers, or on-demand routers. Each server in the cluster is a cluster
member. The job creates a cluster member that runs on a deployment manager node. The cluster member
becomes a managed resource of the job manager.

Before you begin

Start the job manager, the deployment manager on which the cluster resides, and federated nodes of the
deployment manager. A cluster must exist on a deployment manager node.

Determine how you want to configure the cluster member:
- Review topics on clusters and workload management, especially the information about setting cluster
  weights.
- Determine the target to which to add the first cluster member. The target must be a managed target of
  the job manager.
- Determine the appropriate configuration settings for the first cluster member. A copy of the first cluster
  member that you create is stored as part of the cluster data and becomes the template for all additional
  cluster members that you create.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit
jobs. When you submit a job, you can specify a user name and password for authentication and
authorization at the target or targets. When you submit a job to multiple targets, the user name and
password or the credentials for the submitter must apply to all the job targets.

About this task

You can use the administrative console of the job manager or the deployment manager to create a cluster
member for one or more managed clusters. From the console, choose the **Create cluster member** job,
select the clusters to which to add members, specify job options, review the summary, and submit the job.

Instead of using a console, you can run the createClusterMember job script in the AdministrativeJobs
command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to create a cluster member.
   a. Select the **Create cluster member** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets on which to add the cluster member.

   Select one or more managed deployment manager nodes that have the target cluster.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the create cluster member job.
   a. For **Cluster name**, specify the name of the cluster to which you want to add a member. For example, suppose a managed deployment manager cell, `myCellmanager01`, has a managed cluster, `cluster2`. Specify the name of the cluster that is managed by the job manager:

   `cluster2`
   b. For **Member node**, specify the name of the node on which the target cluster resides. For example, suppose the Target resources page shows that the deployment manager node on which `cluster2` resides is named `myNode01`. Specify the member node name:

   `myNode01`
   c. For **Member name**, specify a cluster member name that is unique on the target cluster. For example, suppose a target cluster, `cluster2`, has one cluster member, `cluster_member_1`. Specify a name that is unique to `cluster2`:

   `cluster_member_2`
   d. If you want the new cluster member to have a **Member weight** value other than 2, the default, specify a different weight value.

   A valid weight value is a number between `0` and `100`. The weight controls the amount of work directed to the server. If the weight is greater than the weight assigned to other cluster members, the server receives a larger share of the workload.
   e. Optionally, specify a universally unique identifier (UUID) for the cluster member.

   If you do not specify a value, the product generates a UUID.
   f. If you do not want the product to generate unique port numbers for the cluster member, clear **Generate unique ports**.

   By default, the product generates unique port numbers for HTTP transports defined in the server. The default value is selected (`true`).
   g. If the cluster has a replication domain and you want to create a replicator entry for the new cluster member in the cluster replication domain, select **Replicator entry** (`true`).

   The product uses a replication domain for HTTP session data replication. By default, the product does not create a replicator entry for a cluster member in the cluster replication domain. The default value is clear (`false`).
   h. If you are creating a cluster on a z/OS target, optionally specify a cluster member short name.

   If you do not specify a value, the product generates a unique short name.
   i. Optionally, expand **Additional job parameters** and specify values that further define the cluster member.

Table 15. Additional Create cluster member job parameters.   Specify job parameters as needed.

Parameter name	Description
Template name	If you want to use a specific cluster member template, specify the name of the template.
Template node name	If you want to use an existing cluster member as a template, specify the name of the node that has a cluster member to use as a template for the new cluster member.
Template server name	If you want to use a specific server as a template for the new cluster member, specify the server name.
Node group	If you want the cluster member to belong to a node group, specify the name of the node group.
Core group	If you want the cluster member to belong to a core group, specify the name of the core group.

   j. Click **Next**.
5. Schedule the job.

   The times and dates that you specify are relative to the job manager.

   a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

       If you specify multiple email addresses, separate them with commas.

   b. Select when the job is available for submission.

   c. Select the job expiration.

   d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

   e. Click **Next**.
6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to create a cluster member of the target cluster.

What to do next

On the Job status page, click the ID of the create cluster member job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, click **Jobs** > **Target resources** to see the new cluster member in the list of resources. The status of the member is Stopped.

After creating the cluster member, you can run a job that starts the cluster. Starting the cluster also starts the cluster member.

Creating proxy servers using the job manager:

In a flexible management environment, you can submit the **Create proxy server** job. A proxy server routes requests to application server nodes. The job creates a proxy server that runs on a deployment manager or stand-alone node. The proxy server becomes a managed resource of the job manager.

Before you begin

Start the job manager. If you are adding the proxy server to a deployment manager node, start the deployment manager and the target federated node. If you are adding the proxy server to a stand-alone node, start the stand-alone node and the administrative agent. The target nodes must be managed by the job manager.

To submit the **Create proxy server** job, the target must support proxy servers.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply to all the job targets.

About this task

You can use the administrative console of the job manager or the deployment manager to create a proxy server on one or more managed targets. From the console, choose the **Create proxy server** job, specify job options, review the summary, and submit the job. This topic describes how to use the job manager console to submit the job.

Instead of using a console, you can run the createProxyServer job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to create a proxy server.
   a. Select the **Create proxy server** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets on which to add the proxy server.

   Select one or more managed targets to which you can add proxy servers, such as deployment manager nodes.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the create cluster job.
   a. For **Server name**, specify a proxy server name that is unique on the targets. For example, suppose a managed deployment manager node, `myNode01`, has one proxy server, `proxy_server_1`. Specify a name that is unique to `myNode01`:

      `proxy_server_2`
   b. If you are creating the proxy server on a deployment manager node, for **Node name** specify the name of the target node. For example, suppose you want to add `proxy_server_2` to a managed deployment manager node named `myNode01`, specify the node name:

      `myNode01`
   c. Click **Next**.
5. Schedule the job.

   The times and dates that you specify are relative to the job manager.

a. Optionally specify one or more email addresses where notifications are sent when the job finishes. If you specify multiple email addresses, separate them with commas.

b. Select when the job is available for submission.

c. Select the job expiration.

d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to create the proxy server.

What to do next

On the Job status page, click the ID of the create proxy server job and view the job status. Click the status refresh icon ![icon] to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, click **Jobs** > **Target resources** to see the new proxy server in the list of resources. The status of the proxy server is `Stopped`.

After creating the proxy server, you can run a job that starts the proxy server.

Starting clusters using the job manager:

In a flexible management environment, you can submit the **Start cluster** job to start a cluster that is on a managed target of the job manager.

Before you begin

Start the job manager and the managed targets. The cluster that you want to start must be a resource managed by the job manager. The target cluster must have at least one cluster member. You cannot start a cluster unless it has a cluster member.

You might need to synchronize the nodes of the deployment manager to start the cluster successfully. You can use the Nodes page of the deployment manager administrative console to synchronize the nodes. Click **System administration** > **Nodes** > **Synchronize**.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply to all the job targets.

About this task

You can use the administrative console of the job manager or the deployment manager to start clusters on one or more managed targets. From the console, choose the **Start cluster** job, specify job options, review the summary, and submit the job.

When you start a cluster, all the application servers that are members of that cluster start automatically.

Instead of using a console, you can run the startCluster job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to start a cluster.
   a. Select the **Start cluster** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target on which a cluster resides and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the start cluster job.
   a. For **Cluster name**, specify the name of cluster to start.

      To see the names of existing clusters on the targets, click **Find** on the Specify job parameters page. On the Find target resources page, specify the targets and click **Find**. For example, suppose a managed target, `myCellManager01`, has a cluster named `cluster2` that you want to start. Specify the name:

      `cluster2`
   b. If you want the cluster start operation to ripple across the cluster one member at a time, select **Ripple start** (`true`).

      Otherwise, leave the field empty. The default is clear (`false`).
   c. Optionally, for **Timeout in minutes** specify the maximum amount of time in minutes to wait for the cluster to start before returning the state of the cluster.
   d. Click **Next**.
5. Schedule the job.

   The times and dates that you specify are relative to the job manager.
   a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

      If you specify multiple email addresses, separate them with commas.
   b. Select when the job is available for submission.
   c. Select the job expiration.
   d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.
   e. Click **Next**.
6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to start the clusters and the cluster members.

What to do next

On the Job status page, click the ID of the start server job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job. You might need to synchronize the nodes of the deployment manager before you can start the cluster successfully. After you correct the error condition, submit the job again.

If the job is successful, the status on the Target resources page of the cluster is RUNNING. Click **Jobs** > **Target resources** > *resource_name* to see the resource status.

When a cluster starts, its cluster members also start. If the Target resources page shows that the cluster is RUNNING but the status of cluster members is Stopped, submit the Status or Inventory job. Both jobs refresh data on managed resources. If the status of cluster members remains Stopped, submit the Start server job to start a cluster member.

Starting servers using the job manager:

In a flexible management environment, you can submit the **Start server** job to start an application server on a managed target. The job can start a stand-alone server or a server on a federated node of a deployment manager.

Before you begin

Start the job manager and the targets. If a target is a stand-alone application server, also start the administrative agent.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must be applicable to all of the job targets.

About this task

You can use the administrative console of the job manager or the deployment manager to start application servers on one or more managed targets. From the console, choose the **Start server** job, specify server and job options, review the summary, and submit the job.

gotcha: You cannot start or stop the deployment manager or any node agent by using the job manager console with the submit job option. Performing such a start/stop job submission for the deployment manager or any node agent results in an error message similar to the following.

```
CWWSY0334E: Unable to locate node agent MBean when processing
 job startServer for server dmgr on node CellManager
```

The deployment manager or any node agent is unaffected by this action.

This topic describes how to run the **Start server** job using the job manager console or the deployment manager console. Instead of using a console, you can run the startServer job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure
1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to start a server.
   a. Select the **Start server** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets.
   a. Select a group of targets from the list, or select **Target names**.

Only groups of targets that are valid for the job type that you selected are displayed in the list of target groups.

b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.

c. If user authentication is required, specify a user name, password, or any other authentication values as needed.

d. Click **Next**.

4. Specify parameters for the start server job.

a. For **Server name**, specify the name of server to start.

To see the names of existing servers on the targets, click **Find** on the Specify job parameters page. On the Find target resources page, specify the targets and click **Find**. For example, suppose a managed node, `nodeA`, has a server named `server2` that you want to start. Specify the name:

`server2`

b. If the target node is in a WebSphere Application Server, Network Deployment cell, specify the node name in the cell.

You do not have to specify the node name for a base (stand-alone) node. For example, suppose the managed node, `nodeA`, is a federated node in a WebSphere Application Server, Network Deployment cell. Specify the node name:

`nodeA`

c. Click **Next**.

5. Schedule the job.

The times and dates that you specify are relative to the job manager.

a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

If you specify multiple email addresses, separate them with commas.

b. Select when the job is available for submission.

c. Select the job expiration.

d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to start the application server.

What to do next

On the Job status page, click the ID of the start server job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, the status on the Target resources page of the server is `RUNNING`. Click **Jobs** > **Target resources** > *resource_name* to see the resource status.

After starting the server, you can run the following jobs:
• Install application
• Update application
• Uninstall application

- Stop server

Stopping servers using the job manager:

In a flexible management environment, you can submit the **Stop server** job to stop an application server on a managed target. The job can stop a stand-alone server or a federated node of a deployment manager.

Before you begin

Start the job manager if it is not already running. The target application server must also be running. If a target is a stand-alone application server, also start the administrative agent.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must be applicable to all the job targets.

About this task

You can use the administrative console of the job manager or the deployment manager to stop application servers on one or more managed targets. From the console, choose the **Stop server** job, specify server and job options, review the summary, and submit the job.

Instead of using a console, you can run the stopServer job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure
1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to stop a server.
   a. Select the **Stop server** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets.
   a. Select a group of targets from the list, or select **Target names**.

      Only groups of targets that are valid for the job type that you selected are displayed in the list of target groups.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the stop server job.
   a. For **Server name**, specify the name of server to stop.

      To see the names of existing servers on the targets, click **Find** on the Specify job parameters page. On the Find target resources page, specify the targets and click **Find**. For example, suppose a managed target, `nodeA`, has a server named `server2` that you want to stop. Specify the name:

      `server2`
   b. If the target node is in a WebSphere Application Server, Network Deployment cell, specify the node name in the cell.

You do not have to specify the target name for a base (stand-alone) target. For example, suppose the managed target, `nodeA`, is a federated node in a WebSphere Application Server, Network Deployment cell. Specify the node name:

`nodeA`

  c. Click **Next**.

5. Schedule the job.

The times and dates that you specify are relative to the job manager.

  a. Optionally specify one or more email addresses where notifications are sent when the job finishes. If you specify multiple email addresses, separate them with commas.

  b. Select when the job is available for submission.

  c. Select the job expiration.

  d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

  e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to stop the application server.

What to do next

On the Job status page, click the ID of the stop server job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, the status on the Target resources page of the server is `Stoppped`. Click **Jobs** > **Target resources** > *resource_name* to see the resource status.

After stopping the server, you can run the following jobs:
- Start server
- Delete application server

Stopping clusters using the job manager:

In a flexible management environment, you can submit the **Stop cluster** job to stop a running cluster that is on a managed target of the job manager.

Before you begin

Start the job manager if it is not running already.

To submit the **Stop cluster** job, the deployment manager node on which the cluster resides must be a managed target of the job manager. The cluster must be running.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must be applicable to all the job targets.

About this task

You can use the administrative console of the job manager or the deployment manager to stop clusters on one or more managed targets. From the console, choose the **Stop cluster** job, specify job options, review the summary, and submit the job.

Instead of using a console, you can run the stopCluster job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to stop a cluster.
   a. Select the **Stop cluster** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets.
   a. Select a group of targets from the list, or select **Target names**.

      Only groups of targets that are valid for the job type that you selected are displayed in the list of target groups.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the stop cluster job.
   a. For **Cluster name**, specify the name of cluster to stop.

      To see the names of existing clusters on the targets, click **Find** on the Specify job parameters page. On the Find target resources page, specify the targets and click **Find**. For example, suppose a managed target, `myNode01`, has a cluster named `cluster2` that you want to stop. Specify the name:

      `cluster2`
   b. Optionally, for **Timeout in minutes** specify the maximum number of minutes to wait for the cluster to stop before returning the cluster state.
   c. Click **Next**.
5. Schedule the job.

   The times and dates that you specify are relative to the job manager.
   a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

      If you specify multiple email addresses, separate them with commas.
   b. Select when the job is available for submission.
   c. Select the job expiration.
   d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.
   e. Click **Next**.
6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to stop the cluster.

What to do next

On the Job status page, click the ID of the stop cluster job and view the job status. Click the status refresh icon ![icon] to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, the status on the Target resources page of the cluster is `Stopped`. Click **Jobs** > **Target resources** > *resource_name* to see the resource status.

After stopping the cluster, you can run the following jobs:
- Start cluster
- Delete cluster

Applying properties files to configure servers using the job manager:

In a flexible management environment, you can submit the **Configure properties** job to apply properties files to application servers managed by the job manager. After you edit a properties file that is intended for a managed application server target, use the job manager to distribute the properties file to the managed target and apply the changed file to update the application server configuration.

Before you begin

Properties files provide a way to query and change the product configuration. The files list the most commonly used properties in name and value pairs.

Before you can submit a job to apply a properties file, do the following:

1. Start the job manager. If you are applying the properties file to a deployment manager target, start the deployment manager and the federated node of the deployment manager. If you are applying the properties file to a stand-alone application server target, start the administrative agent. The targets must be managed by the job manager.

2. Use the extractConfigProperties command in the PropertiesBasedConfiguration command group for the AdminTask object to extract a properties file of an application server that is managed by the job manager. You must run the extractConfigProperties command locally or run a wsadmin script that extracts the properties file in a **Run wsadmin script** job. The job manager does not have a job that specifically extracts a properties file.

   For example, suppose you want to extract the server configuration of an application server target that has a server named `server1` and a profile named `AppSrv02`. Run the following wsadmin commands from the `bin` directory of the AppSrv02 profile:

   ```
   wsadmin -lang jython
   ```

   ```
   AdminTask.extractConfigProperties('[-propertiesFileName server.props -configData Server=server1 ]')
   ```

   The product extracts the server configuration to a file named `server.props` in the `bin` directory of the AppSrv02 profile.

3. Open an editor on the properties file, change the value of one or more properties, and save the file.

   Important: The properties file must be in UTF-8. The generated file is in UTF-8 automatically. Ensure that the file remains in UTF-8 after any edits. Use an editor that handles UTF-8, or US-ASCII if the file does not have characters outside the 7-bit US-ASCII character set.

4. Copy the properties file to the `/config/temp/JobManager` directory of the job manager profile.

   If the `JobManager` directory does not exist, create the `JobManager` directory in the job manager profile `/config/temp` directory. To create and access the directory, you must have the appropriate authority.

If the properties file exists on a managed target, you can run the **Collect file** job to copy the properties file from the managed target to the *job_manager_profile*/config/temp/JobManager/*jobToken*/*targetName* directory. See the topic on the collect file job.

5. Run the **Distribute file** job to distribute the properties file from the job manager to one or more application server targets. Remember any value that you specify for **Destination** because you use that location for the job that applies the properties file.

   Note: You must distribute the properties file to the targets before you can run the **Configure properties** job. The distribute file job copies the properties file in the /config/temp/JobManager directory of the job manager profile to the targets. The name of the properties file on the targets becomes whatever value that you specify for the destination when distributing the file. See the topic on the distribute file job.

6. If the properties file uses a variable map file, run the **Distribute file** job to distribute the variable map file. Remember any value that you specify for **Destination**.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must be applicable to all the job targets.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that applies a properties file and configures an application server. From the console, choose the **Configure properties** job, specify the properties file to apply, specify job options, schedule the job, review the summary, and submit the job.

The job makes the following changes to an application server configuration:
- Sets the attribute corresponding to each property specified in the properties file to the new value.
- If an attribute corresponding to the specified property does not exist in the configuration, creates an attribute in the configuration.
- If a configuration object specified in the properties file does not exist in the configuration, creates a configuration object.

Optionally, you can specify that the job manager use variables that are set in a variable map file when applying the properties file. Specify the location of a variable map file to include with the properties file.

This topic describes how to use the job manager console or the deployment manager to submit the job. Instead of using a console, you can submit the job by running the configProperties command in the AdministrativeJobs command group to configure the properties for the application server target. See the Administrative job types topic.

The configure properties job uses the applyConfigProperties command in the PropertiesBasedConfiguration command group for the AdminTask object to configure the properties for the target.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to apply a properties file to one or more application server targets.
   a. Select the **Configure properties** job type from the list.
   b. Optionally describe the job.
   c. Click **Next**.

3. For the job target, choose the application server targets that you want to configure by applying a properties file.

    a. Select **Target names**.

    b. Specify the target names to which you previously distributed the properties file and click **Add**, or click **Find** and specify the application server targets as chosen targets on the Find targets page.

    c. Click **Next**.

    d. If user authentication is required, specify a user name, password, or any other authentication values as needed.

    e. Click **Next**.

4. Specify parameters for the configure properties job.

    a. Specify the location of the properties file that you want to apply.

        The properties file location is the destination value that you specified for the **Distribute file** job.

    b. If the properties file uses a variable map file, specify the location of the variable map file.

        Use the destination value that you specified for the **Distribute file** job.

    c. Click **Next**.

5. Schedule the job.

    The times and dates that you specify are relative to the job manager.

    a. Optionally specify one or more email addresses where notifications are sent when the job is done.

        If you specify multiple email addresses, separate them by commas. The email addresses are saved in your console preferences. Each email address is validated for format errors.

    b. Select when the job is available for submission.

        You can submit the job to be available now, or specify a time and date that the job is retrieved from the job manager.

    c. Select the job expiration.

        The job expiration is the time at which the job will no longer be available for targets to run. You can use the default expiration, specify a time and date for the job expiration, or specify an amount of time in which the job expires. The default expiration is defined on the Job manager configuration page.

    d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

    e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The job manager performs the following operations:

*   Makes the **Configure properties** job available to the targets.
*   Reports on the status of the job at each target.

What to do next

On the Job status page, click the job ID and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job status is Succeeded, check a target to ensure that the properties file was applied.

Deleting application servers using the job manager:

You can submit the **Delete application server** job to remove a server from the flexible management environment. The job can delete a stand-alone server or a server on a federated node of a deployment manager.

Before you begin

Each server that you want to delete must be on a managed target of the job manager.

Start the job manager. If the server that you want to delete is on a federated node of a deployment manager, also start the deployment manager and the federated node. If the target is a stand-alone application server, also start the administrative agent.

About this task

You can use the administrative console of the job manager or the deployment manager to delete an application server from one or more managed targets. From the console, choose the **Delete application server** job, specify server and job options, review the summary, and submit the job.

Instead of using a console, you can run the deleteApplicationServer job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to delete a server.
   a. Select the **Delete application server** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets from which to delete a server.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the delete server job.
   a. For **Server name**, specify the name of the server to delete.

      To see the names of existing servers on the targets, click **Find** on the Specify job parameters page. On the Find target resources page, specify the targets and click **Find**. For example, suppose a managed target, `nodeA`, has a server named `server2` that you want to delete. Specify the server name:

      `server2`
   b. If the target is in a WebSphere Application Server, Network Deployment cell, specify the node name in the cell from which to delete the server.

      You do not have to specify the node name for a base (stand-alone) node. For example, suppose the managed target, `nodeA`, is a federated node in a WebSphere Application Server, Network Deployment cell. Specify the node name:

      `nodeA`
   c. Click **Next**.
5. Schedule the job.

a. Optionally specify one or more email addresses where notifications are sent when the job finishes. If you specify multiple email addresses, separate them with commas.

b. Select when the job is available for submission.

c. Select the job expiration.

d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to delete the application server.

What to do next

On the Job status page, click the ID of the delete application server job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, click **Jobs** > **Target resources**. The names of deleted servers are not in the list of resources.

Deleting clusters using the job manager:

You can submit the **Delete cluster** job to remove a cluster from the flexible management environment.

Before you begin

Start the job manager, the deployment manager, and the targets on which the cluster resides.

Each cluster that you want to delete must be a managed resource of the job manager.

About this task

You can use the administrative console of the job manager or the deployment manager to delete a cluster from one or more managed targets. From the console, choose the **Delete cluster** job, specify job options, review the summary, and submit the job.

The job stops the cluster, if it is running, and removes the cluster from the flexible management environment and from the deployment manager node.

Instead of using a console, you can run the deleteCluster job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.

2. Choose a job to delete a cluster.

a. Select the **Delete cluster** job type from the list.

b. Optionally specify a description of the job.

c. Click **Next**.

3. Choose the job targets from which to delete a cluster.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.

      For example, suppose a managed target, `myNode01`, has a cluster named `cluster2` that you want to remove. Specify the target name:

      `myNode01`
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the delete cluster job.
   a. For **Cluster name**, specify the name of cluster to remove.

      To see the names of existing clusters on the targets, click **Find** on the Specify job parameters page. On the Find target resources page, specify the targets and click **Find**. To continue with the example, to remove `cluster2` from the `myNode01` target, specify the cluster name:

      `cluster2`
   b. If the cluster belongs to a replication domain, optionally select **Delete replication domain** (`true`) to remove the cluster replication domain.

      By default, the product does not delete the cluster replication domain when the cluster is deleted. The default value is clear (`false`).
   c. Click **Next**.
5. Schedule the job.
   a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

      If you specify multiple email addresses, separate them with commas.
   b. Select when the job is available for submission.
   c. Select the job expiration.
   d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.
   e. Click **Next**.
6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and delete the cluster.

What to do next

On the Job status page, click the ID of the delete cluster job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again. correct the error condition, and submit the job again.

If the job is successful, click **Jobs** > **Target resources**. The names of deleted clusters are not in the list of resources.

Deleting cluster members using the job manager:

You can submit the **Delete cluster member** job to remove a cluster member from the flexible management environment.

Before you begin

Start the job manager and the managed targets. The cluster member that you want to delete must be a managed resource of the job manager.

About this task

You can use the administrative console of the job manager or the deployment manager to delete a cluster member from one or more managed clusters. From the console, choose the **Delete cluster member** job, specify job options, review the summary, and submit the job.

The job stops the cluster member, if it is running, and removes the cluster member from the flexible management environment.

Instead of using a console, you can run the deleteClusterMember job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to delete a cluster member.
   a. Select the **Delete cluster member** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose one or more managed targets on which the cluster member resides.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the delete member job.
   a. For **Cluster name**, specify the name of the cluster that groups the cluster member.

   To see the names of existing clusters on the targets, click **Find** on the Specify job parameters page. On the Find target resources page, specify the targets and click **Find**. For example, suppose the cluster member that you want to delete, `cluster_member_2`, belongs to the `cluster2` cluster. Specify the cluster name:

   `cluster2`
   b. For **Member node**, specify the name of the node on which the cluster member resides. For example, suppose the Target resources page shows that the deployment manager node on which `cluster_member_2` resides is named `myNode01`. Specify the member node name:

   `myNode01`
   c. For **Member name**, specify the name of the cluster member that you want to delete. For example, specify the cluster member name:

   `cluster_member_2`
   d. If the cluster has a replication domain and you want to remove a replicator entry for the cluster member, select **Delete entry** (`true`).

   By default, the product does not delete the replicator entry having the server name of this cluster member from the cluster replication domain. The default value is clear (`false`).
   e. Click **Next**.
5. Schedule the job.
   a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

If you specify multiple email addresses, separate them with commas.

b. Select when the job is available for submission.

c. Select the job expiration.

d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to delete the cluster member.

What to do next

On the Job status page, click the ID of the delete cluster member job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, click **Jobs** > **Target resources**. The names of deleted cluster members are not in the list of resources.

Deleting proxy servers using the job manager:

You can submit the **Delete proxy server** job to remove a server from the flexible management environment. The job can delete a proxy server from a stand-alone server or a federated node of a deployment manager.

Before you begin

Start the job manager and the targets. If a target is a stand-alone application server, also start the administrative agent.

To submit the **Delete proxy server** job, the target must support proxy servers.

Each proxy server that you want to delete must be a managed resource of the job manager.

About this task

You can use the administrative console of the job manager or the deployment manager to delete a proxy server from one or more managed targets. From the console, choose the **Delete proxy server** job, specify server and job options, review the summary, and submit the job.

The job stops the proxy server, if it is running, and removes the proxy server from the flexible management environment.

Instead of using a console, you can run the deleteProxyServer job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.

2. Choose a job to delete a proxy server.

a. Select the **Delete proxy server** job type from the list.

b. Optionally specify a description of the job.

c. Click **Next**.

3. Choose the job targets from which to delete the proxy server.

   a. Select a group of targets from the list, or select **Target names**.

   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.

   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.

   d. Click **Next**.

4. Specify parameters for the delete proxy server job.

   a. For **Server name**, specify the name of the proxy server to delete.

      To see the names of existing servers on the targets, click **Find** on the Specify job parameters page. On the Find target resources page, specify the targets and click **Find**. For example, suppose a managed target, `myNode01`, has a server named `proxy_server_2` that you want to delete. Specify the server name:

      `proxy_server_2`

   b. If the target is in a WebSphere Application Server, Network Deployment cell, specify the node name in the cell from which to delete the server.

      You do not have to specify the node name for a base (stand-alone) node. For example, suppose the managed node, `myNode01`, is a federated node in a WebSphere Application Server, Network Deployment cell. Specify the node name:

      `myNode01`

   c. Click **Next**.

5. Schedule the job.

   a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

      If you specify multiple email addresses, separate them with commas.

   b. Select when the job is available for submission.

   c. Select the job expiration.

   d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

   e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to delete a proxy server.

What to do next

On the Job status page, click the ID of the delete proxy server job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, click **Jobs** > **Target resources**. The names of deleted servers are not in the list of resources.

Submitting jobs to manage files

In a flexible management environment, you can submit jobs to collect files from managed targets and copy them to the job manager, to distribute files to managed targets, and to remove files from managed targets.

Before you begin

Before submitting a job, start the job manager and the targets. If a target is a stand-alone application server, also start the administrative agent.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must be applicable to all of the job targets.

About this task

You must distribute files to managed targets before you can submit jobs that use the files. For example, you must distribute an enterprise application before you can install or update the application on managed application server targets. Similarly, you must distribute a properties file to managed application server targets before you can apply the file to configure the application servers. You also must distribute a wsadmin script file before submitting a job that run the script.

The topics in this section describe how to administer files by running jobs in the job manager console or the deployment manager console. Instead of using a console, you can run wsadmin commands in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

- Run the Collect file job to copy either a single file or a directory of files from managed targets to the job manager.
- Run the Distribute file job to copy a single file to managed targets.
- Run the Remove file job to delete a file or directory of files from managed targets.

Results

The collect file job copies a file or directory into the *job_manager_profile*/config/temp/JobManager/ *jobToken*/*targetName* directory.

The distribute file job copies a file into the downloadedContent directory of the administrative agent or deployment manager profile.

The Remove file job deletes a file from the downloadedContent directory.

What to do next

On the Job status page, click the ID of the job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

Collecting files for the job manager:

In a flexible management environment, you can submit the **Collect file** job to copy a single file or a directory of files from managed targets, which can be hosts or nodes, to the job manager. You might collect files before submitting jobs that use the files. For example, you might collect and then distribute an enterprise application before you install or update the application on managed application server targets. Similarly, you might collect and then distribute a properties file to managed application server targets before you apply the file to configure the application servers.

Before you begin

Start the job manager. Ensure that the targets from which you want to copy a file are registered with the job manager.

If you are collecting a file on a host, you must first register the host with the job manager.

If a target is deployment manager, start the deployment manager. If a target is a stand-alone application server, start the administrative agent.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply to all the job targets.

You can simplify administration of a large number of targets by submitting jobs against groups of targets. Before you can submit a job for a group of targets, you must create the group of targets.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that transfers a file or directory of files from targets to the job manager. From the console, choose the **Collect file** job, specify the file or directory to transfer, specify job options, schedule the job, review the summary, and submit the job. When the job runs, the job manager copies the file from the application server or deployment manager profile directory to the job manager profile `config/temp/JobManager/` `jobToken/targetName` directory.

When collecting a directory from a node, the job recursively zips the directory contents and copies the resulting compressed format file. When collecting multiple files from a host, you can specify wildcards in the filename.

This topic describes how to run the **Collect file** job using the job manager console or the deployment manager console. Instead of using a console, you can run the collectFile job script in the AdministrativeJobs command group. See the Administrative job types topic.

Note: The collect file job can transfer files that are accessible from the z/OS UNIX shell only. Files are transferred in binary mode, therefore there is no conversion of character sets or encoding during the transfer.

Procedure

1. For nodes, determine the location of the file or directory that you want to collect relative to the *profile_root* directory of the target.

   For example, suppose that you want to collect the `DynaCacheEsi.ear` file from the *profile_root*`/AppSrv01/config/cells/myNode01Cell/applications/DynaCacheEsi.ear` directory. AppSrv01 is the target node profile name. AppSrv01 is a stand-alone application server. myNode01Cell is the AppSrv01 cell name. The location of the `DynaCacheEsi.ear` file relative to the profile root, AppSrv01, is `config/cells/myNode01Cell/applications/DynaCacheEsi.ear`

   For hosts, you can collect one or more files from any location of the hosts if your user ID has the correct permissions.

2. Click **Jobs** > **Submit** from the navigation tree of the administrative console.

3. Choose the job.

   a. Select the **Collect file** job type from the list.

   b. Optionally describe the job.

   c. Click **Next**.

4. Choose the targets from which you want to collect the file or directory.

    a. Select a group of targets from the list, or select **Target names**.

        Only target groups that are valid for the job type that you selected are displayed in the list of target groups.

        The target can be a host. However, you can only collect a file from a host. You cannot collect a directory from a host.

    b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.

        Select the target from which you want to collect a file or directory. To continue with the example in step 1, suppose the AppSrv01 profile is registered with the job manager as nodeA. Select `nodeA`.

    c. If user authentication is required, specify a user name, password, or any other authentication values as needed.

    d. Click **Next**.

5. Specify parameters for the collect file job.

    a. For **Source**, you have different options depending on if you are collecting files from a host or from a node:

        • **Node**: specify the path of the file or directory that you want to collect, relative to the *profile_root* directory of the target.

            For example, to collect the `DynaCacheEsi.ear` file from the *profile_root*`/AppSrv01/config/cells/myNode01Cell/applications/DynaCacheEsi.ear` directory, specify the following path:

`config/cells/myNode01Cell/applications/DynaCacheEsi.ear/DynaCacheEsi.ear`

            To collect a directory such as the `logs` directory under the *profile_root* of the target, specify `logs` in the **Source** field.

        • **Host**: Specify a fully qualified path of the file. There are no restrictions on the source. You can specify files using wildcards. Supported wildcard characters are (*) and (?).

            – Asterisk (*) – for multiple unknown or variable characters in the term.

            – Question mark (?) – for a single unknown or variable character in the term.

    b. Optional: For **Destination**, specify a destination name for the file or directory that is being copied to the job manager.

        By default, the file or directory is placed in the *job_manager_profile*`/config/temp/JobManager/`*jobToken*`/`*targetName* directory. The file or directory retains its name unless you specify a different name in the **Destination** field. For the `DynaCacheEsi.ear` example, if you do not specify a value for **Destination**, the product copies the `DynaCacheEsi.ear` file to a directory such as *profile_root*`/JobMgr01/config/temp/JobManager/124517860634322577/nodeA`. In this example, `JobMgr01` is the name of the job manager profile, `124517860634322577` is the job token identifier, and `nodeA` is the target from which the file was copied. The `DynaCacheEsi.ear` file is placed in the nodeA directory.

        • Node

            Suppose that you want the `DynaCacheEsi.ear` file to have a name such as `dynacache_esi_sample` in the nodeA directory. If you specify `dynacache_esi_sample` for **Destination**, the `DynaCacheEsi.ear` file is copied to the *profile_root*`/JobMgr01/config/temp/JobManager/`*jobToken*`/nodeA` directory, where it has the name `dynacache_esi_sample`.

        • Host

            If you chose to specify a destination for a host, you must specify a directory. When collecting a file from a host, the destination cannot be a file name.

        If you specify a **Destination** value, remember the value. If you later run a job that references the collected file or directory, you can use the destination value to identify the file or directory.

    c. For **Distribution provider**, if you use a distribution provider other than the default distribution provider, specify the name of the distribution provider. For the `DynaCacheEsi.ear` file or `logs` directory example, do not specify a value and use the default distribution provider.

d. Click **Next**.

6. Schedule the job.

The times and dates that you specify are relative to the job manager.

a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

If you specify multiple email addresses, separate them with commas. The email addresses are saved in your console preferences. Each email address is validated for format errors.

b. Select when the job is available for submission.

You can submit the job to be available now, or specify a time and date that the job is retrieved from the job manager.

c. Select the job expiration.

The job expiration is the time at which the job will no longer be available for targets to run. You can use the default expiration, specify a time and date for the job expiration, or specify an amount of time in which the job expires. The default expiration is defined on the Job manager configuration panel.

d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

e. Click **Next**.

7. Review the summary, and click **Finish** to submit the job.

Results

The job manager copies the file or directory into the *job_manager_profile*/config/temp/JobManager/ *jobToken*/*targetName* directory. The name of the collection file or directory is the destination. If you did not specify a destination value, then the file or directory retains its original name.

For the DynaCacheEsi.ear example, the file is copied to the *job_manager_profile*/config/temp/ JobManager/*jobToken*/nodeA directory.

What to do next

On the Job status page, click the ID of the collect file job and view the job status. Click the status refresh

icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job manager successfully copies the file or directory from the targets to the job manager, you can submit jobs that distribute collected files.

Distributing files from the job manager to targets:

In a flexible management environment, you can submit the **Distribute file** job to copy files to managed targets of the job manager. You must distribute files before you can submit jobs that use the files. For example, you must distribute an enterprise application before you can install or update the application on managed application server targets. Similarly, you must distribute a properties file to managed application server targets before you can apply the file to configure the application servers.

Before you begin

Start the job manager. Ensure that the targets to which you want to copy a file are registered with the job manager.

- To register a stand-alone application server with the job manager, first use the administrative agent registerNode command to register the stand-alone application server with the administrative agent. Then, use the administrative agent console or registerWithJobManager command to register the stand-alone application server target with the job manager.
- To register a deployment manager with the job manager, use the deployment manager console or the registerWithJobManager command.

If you are collecting a file on a host, you must first register the host with the job manager.

Start the targets. If a target is a stand-alone application server, also start the administrative agent.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply all the job targets.

You can simplify administration of multiple targets by submitting jobs against groups of targets. Before you can submit a job for a target group, you must create the target group.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that moves a file to targets. From the console, choose the **Distribute file** job, specify the file to transfer, specify job options, schedule the job, review the summary, and submit the job. When the job runs, the job manager copies the file into the `downloadedContent` directory of the administrative agent profile or deployment manager profile.

You must distribute a file before running the following jobs:
- Install application
- Update application
- Configure properties
- Run wsadmin script

This topic describes how to run the **Distribute file** job using a job manager console or the deployment manager console. Instead of using a console, you can run the distributeFile job script in the AdministrativeJobs command group to distribute the file to targets. See the Administrative job types topic.

Note: The distribute file job can transfer files that are accessible from the z/OS UNIX shell only. Files are transferred in binary mode, therefore there is no conversion of character sets or encoding during the transfer.

Procedure

1. Copy the file that you want to distribute to the `/config/temp/JobManager` directory of the job manager profile.

   If the `JobManager` directory does not exist, create the `JobManager` directory in the job manager profile `/config/temp` directory. To create and access the directory, you must have the appropriate authority.

   For example, copy the `DynaCacheEsi.ear` file from the *app_server_root*/`installableApps` directory to the `/config/temp/JobManager` directory of the job manager profile.

2. Click **Jobs** > **Submit** from the navigation tree of the administrative console.

3. Choose a job to distribute a file.

   a. Select the **Distribute file** job type from the list.

   b. Optionally describe the job.

   c. Click **Next**.

4. Choose the targets to which you want to distribute the file.

   a. Select a group of targets from the list, or select **Target names**.

      Only groups of targets that are valid for the job type that you selected are displayed in the list of target groups.

   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.

      The target can be a host.

      If the targets are stand-alone application servers that are administered by an administrative agent, you can select one stand-alone target instead of all the stand-alone targets. All the targets can use the file that is distributed to the `downloadedContent` directory of the administrative agent.

   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.

   d. Click **Next**.

5. Specify parameters for the distribute file job.

   a. For **Source**, specify the uniform resource locator (URL) for the file that you want to distribute.

      For the default distribution provider, the location is a file URL relative to the job manager profile `config/temp/JobManager` directory. For the `DynaCacheEsi.ear` file, specify the following:

      • **Node:**

        `file:/DynaCacheEsi.ear`

      • **Host:**

        `DynaCacheEsi.ear`

      You can specify files on hosts using wildcards. Supported wildcard characters are (*) and (?).

      • Asterisk (*) – for multiple unknown or variable characters in the term.

      • Question mark (?) – for a single unknown or variable character in the term.

   b. For **Destination**, specify the location on the target where the job manager stores the file. The destination parameter is relative to the `downloadedContent` directory of the administrative agent or deployment manager profile.

      Remember the value that you specify for the file. If you later run a job that references this file, you use the destination value to identify the file. For the `DynaCacheEsi.ear` file, specify a value that identifies the file. For example:

      `dynacache_esi_sample`

      For hosts, there are no restrictions. Specify an absolute path. The path must be a directory. For example:

      `/home/userA`

   c. For **Distribution provider**, if you use a distribution provider other than the default distribution provider, specify the name of the distribution provider. For the `DynaCacheEsi.ear` example, do not specify a value and use the default distribution provider.

   d. Click **Next**.

6. Schedule the job.

   The times and dates that you specify are relative to the job manager.

   a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

      If you specify multiple email addresses, separate them with commas. The email addresses are saved in your console preferences. Each email address is validated for format errors.

   b. Select when the job is available for submission.

      You can submit the job to be available now, or specify a time and date that the job is retrieved from the job manager.

   c. Select the job expiration.

The job expiration is the time at which the job will no longer be available for targets to run. You can use the default expiration, specify a time and date for the job expiration, or specify an amount of time in which the job expires. The default expiration is defined on the Job manager configuration panel.

   d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

   e. Click **Next**.

7. Review the summary, and click **Finish** to submit the job.

Results

The job manager copies the file into the `downloadedContent` directory of the administrative agent or deployment manager profile. The name of the file is the destination.

For the `DynaCacheEsi.ear` example, a file named `dynacache_esi_sample` is copied to the `downloadedContent` directory.

What to do next

On the Job status page, click the ID of the distribute file job and view the job status. Click the status

refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job manager successfully copies the file to the targets, submit jobs that use the distributed file:
- Install application
- Update application
- Configure properties
- Run wsadmin script
- Remove file

For the `DynaCacheEsi.ear` example, you can install the application.

Removing files from targets using the job manager:

You can submit the **Remove file** job to delete a file from managed targets. The job can remove a file that previously was distributed to managed targets of the job manager.

Before you begin

Start the job manager and the managed targets. If a target is a stand-alone application server, also start the administrative agent.

The file that you want to remove must exist on the managed target.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that removes a file. From the console, choose the **Remove file** job, specify the file to remove, specify job options, review the summary, and submit the job. When the job runs, the job manager deletes the file from the `downloadedContent` directory of the administrative agent or deployment manager profile.

Instead of using a console, you can run the removeFile job script in the AdministrativeJobs command group. See the Administrative job types topic.

Note: The remove file job can remove files that are accessible from the z/OS UNIX shell only.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to remove a file from selected targets.
   a. Select the **Remove file** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.

      The target can be a host. However, when the target is a host, a wildcard character is not supported for the remove file job.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify job parameters.
   a. For **Location**, specify the **Destination** value given when distributing the application file.

      For example, the topic on distributing files using the job manager describes how to distribute the `DynaCacheEsi.ear` example file. To remove that file, specify the destination value that identifies the `DynaCacheEsi.ear` file:

      `dynacache_esi_sample`
   b. For **Distribution provider**, if you use a distribution provider other than the default distribution provider, specify the name of the distribution provider.

      For the `DynaCacheEsi.ear` example, do not specify a value and use the default distribution provider.
   c. Click **Next**.
5. Schedule the job.
6. Review the summary, and click **Finish** to submit the job.

Results

The job manager runs the job and removes the file from the `downloadedContent` directory of the administrative agent or deployment manager profile.

For the `DynaCacheEsi.ear` example, the file named `dynacache_esi_sample` is deleted from the `downloadedContent` directory.

What to do next

On the Job status page, click the ID of the remove file job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

You can delete the file from the `/config/temp/JobManager` directory of the job manager profile. The file was placed in that directory to enable the distribute file job.

Submitting jobs that test connections to a remote host

You can submit the **Test connection** job to verify access to remote hosts. After a remote host computer is registered with a job manager, you can use the **Test connection** job to check the connection between the job manager and the remote host.

Before you begin

Before you can run the **Test connection** job, the job manager must be running and the remote host must be a target of the job manager.

If the host is not currently a registered target, complete the following steps to make the host a target of the job manager:

- Start the job manager.
- In the job manager console, click **Jobs** > **Targets** > **New Host** and complete the fields on the New targets page.

  A host is a computer. A remote host typically is a different computer than the one on which the job manager is installed.

  To run the **Test connection** job, a WebSphere Application Server product does not need to be installed on the remote host.

Using the console **New Host** option or a registerHost command to make a remote host a target verifies access to the host. Immediately after making a host a target, you do not need to run the **Test connection** job. The **Test connection** job is useful if the host has been registered for a while and you want to verify that the password specified using the job manager is still valid.

About this task

You can use the job manager console or the deployment manager console to submit a job that checks the connection from the job manager to a host. From the console, choose the **Test connection** job, specify the host computer, specify job options, review the summary, and submit the job.

Instead of using a console, you can run the testConnection job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to manage the profiles of a remote host target.
   a. Select the **Test connection** job type from the list.

      The **Test connection** job is only available in the list if the job manager has a host as a target. Stand-alone application servers or deployment managers registered with the job manager are targets, but not hosts.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.

      On the Find targets page, the job type is automatically set to `testConnection` to filter search results. Click **Find** on this page to view the list of hosts in the **Excluded targets** list. Select the target, click **>** to move the host name to the **Chosen targets** list, and then click **OK**.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify job parameters.

   Click **Next**. There are no parameters for this job.
5. Schedule the job.

6. Review the summary, and click **Finish** to submit the job.

Results

The job manager runs the job and attempts to access the host.

What to do next

On the Job status page, click the ID of the testConnection job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

Submitting jobs to run commands on a remote host

You can submit the **Run command on remote host** job to run command-line utilities such as startServer, wsadmin commands, or operating system commands on a remote host. A WebSphere Application Server installation on the remote host is required to run product commands, but is not required for non-product commands.

Before you begin

Before you can run the **Run command on remote host** job, complete the following steps:
- Start the job manager.
- Make a remote host a target of the job manager. In the job manager console, click **Jobs** > **Targets** > **New Host** and complete the fields on the New targets page.

  A host is a computer. A remote host typically is a different computer than the one on which the job manager is installed. The remote host does not require a WebSphere Application Server installation unless you want to run product commands such as startServer or wsadmin.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that runs commands on a remote host. From the console, choose the **Run command on remote host** job, specify the remote host computer, specify the command to run, review the summary, and submit the job. After the job submission, the job manager runs the specified command and records any messages in job manager `stdErr.txt` or `stdOut.txt` logs.

Instead of using a console, you can run the runCommand job script in the AdministrativeJobs command group. See the Administrative job types topic.

To run wsadmin commands in jobs, you can also use the **Run wsadmin script** job, which additionally enables you to specify script parameters. See information about submitting jobs to run wsadmin scripts.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to run a WebSphere Application Server command on a remote host target.
   a. Select the **Run command on remote host** job type from the list.

      The **Run command on remote host** job is only available in the list if the job manager has a host target. Stand-alone application servers or deployment managers registered with the job manager are targets, but not host targets.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets.
   a. Select a group of targets from the list, or select **Target names**.

b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.

On the Find targets page, the job type is automatically set to `runCommand` to filter search results. Click **Find** on this page to view the list of hosts in the **Excluded targets** list. Select the target, click **>** to move the host name to the **Chosen targets** list, and then click **OK**.

c. If user authentication is required, specify a user name, password, or any other authentication values as needed.

d. Click **Next**.

4. Specify job parameters.

a. For **Command or script**, specify the command or script to run on the host.

For example, to run the versionInfo command in a `bin` directory of a product installation on the remote host, specify the following command:

`versionInfo.sh`

For **Command or script**, you can specify a command with arguments; for example:

`startServer.sh server1`

b. For **Working directory**, specify the fully qualified path of the directory where the command resides.

By default, the working directory is set to the home directory of the user.

For example, specify a `bin` directory of the product installation on a remote host:

`/WAS_v800/IBM/WebSphere/AppServer/bin`

c. Click **Next**.

5. Schedule the job.

6. Review the summary, and click **Finish** to submit the job.

Results

The job manager runs the job. If the job options specify the command name and path correctly, the command runs on the host and any messages resulting from the command are written to job manager logs.

For the versionInfo example, a message such as the following is written to the `stdOut.txt` file in the *profile_root*/JobMgr01/config/temp/JobManager/*job_ID*/*host_name*/logs directory:

```
-----------------------------------------------------------------------------
IBM WebSphere Product Installation Status Report
-----------------------------------------------------------------------------

Report at date and time August 30, 2010 11:20:50 AM EDT

Installation
-----------------------------------------------------------------------------
Product Directory      C:\WAS_v800\IBM\WebSphere\AppServer
Version Directory      C:\WAS_v800\IBM\WebSphere\AppServer\properties\version
DTD Directory          C:\WAS_v800\IBM\WebSphere\AppServer\properties\version\dtd
Log Directory          C:\Documents and Settings\All Users\Application Data\IBM\Installation Manager\logs

Product List
-----------------------------------------------------------------------------
ND                     installed

Installed Product
-----------------------------------------------------------------------------
Name                   IBM WebSphere Application Server - ND
Version                8.0.0.0
ID                     ND
Build Level            build_2464
Build Date             8/17/10
Architecture           x86 (32 bit)
Installed Features     EJBDeploy tool for pre-EJB 3.0 modules
```

```
                Sample applications
                Stand-alone thin clients and resource adapters
--------------------------------------------------------------------------------
End Installation Status Report
--------------------------------------------------------------------------------
```

For the startServer example that starts server1, the command starts server1 on the host. A message such as the following is written to the `stdOut.txt` file in the *profile_root*/JobMgr01/config/temp/JobManager/ *job_ID/host_name/*logs directory:

```
ADMU0116I: Tool information is being logged in file
           C:\WAS_v800\IBM\WebSphere\AppServer\profiles\AppSrv01\logs\server1\startServer.log
ADMU0128I: Starting tool with the AppSrv01 profile
ADMU3100I: Reading configuration for server: server1
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server server1 open for e-business; process id is 5384
```

What to do next

On the Job status page, click the ID of the runCommand job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

Submitting jobs to manage profiles on a remote host

You can submit the **Manage profiles** job to create, delete, augment, back up, list, and restore profiles on remote hosts. Profiles are sets of files that define the runtime environment. On local host computers, you can use the manageprofiles command or the Profile Management Tool to manage profiles. After a remote host computer is registered with a job manager, you can use the **Manage profiles** job to run the manageprofiles command on the remote host. The remote host must have a WebSphere Application Server installation.

Before you begin

Restriction: The **Manage profiles** job is only available on distributed and IBM i operating systems. It is not available on z/OS operating systems.

Before you can run the **Manage profiles** job, complete each of the following steps:

- Start the job manager.
- Make a remote host a target of the job manager. In the job manager console, click **Jobs** > **Targets** > **New Host** and complete the fields on the New targets page.

  A host is a computer. A remote host typically is a different computer than the one on which the job manager is installed. The remote host must have a WebSphere Application Server installation. The product does not need to be running on the remote host for the **Manage profiles** job to complete successfully.

- Create a response file that defines tasks that you want the manageprofiles command of the product installation on the remote host to perform.

  Any of the tasks provided by the manageprofiles command can be included in your response file. For descriptions of create, delete, list, and other profile management tasks, see information about the manageprofiles command.

  For example, to create the profile `test1` in an installation on a remote host, create the following response file named `test_response_file.txt`:

  Specify the command task in the first line of the response file, and use a separate line for each manageprofiles command parameter.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that manages profiles of a remote WebSphere Application Server installation. From the console, choose the **Manage profiles** job, specify the remote host computer, specify job options, review the summary, and submit the job. When the job runs, the job manager performs the profile management tasks described in the response file and changes the profile configuration in the product installation on the remote host.

Instead of using a console, you can run the manageprofiles job script in the AdministrativeJobs command group. See the topic on administrative job types.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to manage the profiles of a remote host target.
   a. Select the **Manage profiles** job type from the list.

      The **Manage profiles** job is only available in the list if the job manager has a host as a target. Stand-alone application servers or deployment managers registered with the job manager are targets, but not hosts.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.

      On the Find targets page, the job type is automatically set to `manageprofiles` to filter search results. Click **Find** on this page to view the list of hosts in the **Excluded targets** list. Select the target, click **>** to move the host name to the **Chosen targets** list, and then click **OK**.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify job parameters.
   a. For **WebSphere Application Server home**, specify the location of the product installation on the host.

      Specify the *app_server_root* directory.

      For example, suppose the remote host computer has a Windows operating system and a WebSphere Application Server Version 8.0 installation with an *app_server_root* directory of `C:\WAS_v800\IBM\WebSphere\AppServer`. Specify the following path:

      `C:\WAS_v800\IBM\WebSphere\AppServer`
   b. For **Response file location**, specify the fully qualified path of the response file.

      The response file contains instructions for the manageprofiles command of the product installation on the host.

      For example, specify the location of the `test_response_file.txt` response file described in the "Before you begin" section of this topic to create the profile `test1` in the installation on the host. In this example, the response file is in the temp directory of the same computer on which the job manager resides:
   c. Click **Next**.
5. Schedule the job.
6. Review the summary, and click **Finish** to submit the job.

Results

The job manager runs the job. The manageprofiles command on the host changes the profile configuration in the product installation as instructed in the response file.

For the `test_response_file.txt` example, a profile named `test1` is created on the host.

What to do next

On the Job status page, click the ID of the manageprofiles job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

Submitting Installation Manager jobs

In a flexible management environment, you can submit jobs to install Installation Manager instances, update Installation Manager with a repository (not supported on z/OS targets), manage Installation Manager offerings, and install WebSphere Application Server Version 8.0 products.

Before you begin

Note: This topic applies to WebSphere Application Server Version 8.0 only.

Start the job manager and make a remote host a target of the job manager. In the job manager console or deployment manager console, click **Jobs** > **Targets** > **New Host** and complete the fields on the New targets page.

A remote host typically is a different computer than the one on which the job manager is installed.

To submit jobs, your ID at the job manager must be authorized for the administrator role or the operator role. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must be applicable to all of the job targets.

SSH code is not automatically included with z/OS operating systems. You must ensure SSH is installed and enabled on any target you want to access using CIM.

Note: CIM jobs to install, uninstall, and update Installation Manager are not supported on z/OS targets. You must first install Installation Manager on z/OS targets before using CIM manage offerings jobs.

Note: For Version 8.0, centralized installation manager (CIM) functions are accessed through the job manager. Using the job manager, you can perform the following functions:
- Install, update, and uninstall WebSphere Application Server offerings on remote machines
- Install, update, and uninstall IBM Installation Manager on remote machines. Not supported on z/OS targets. For z/OS targets, you must install Installation Manager prior to working with CIM.
- Collect, distribute, and delete files on remote hosts
- Run scripts on remote hosts

For Version 8.0, CIM functions are not compatible with CIM Version 6.x or 7.x.

Note: IBM Installation Manager 1.4.3 or above is required.

About this task

You can use the Installation Manager to install and manage installations on remote hosts. Using the job manager, you can run jobs that create and update Installation Manager instances and install the product on remote hosts.

The topics in this section describe how to use the Installation Manager by running jobs in the job manager console or the deployment manager console. Instead of using a console, you can run wsadmin commands in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure
- Run the install Installation Manager job.
- Run the update Installation Manager job.
- Run the uninstall Installation Manager job.
- Run the install SSH public key job.
- Manage your Installation Manager install kits.

What to do next

On the Job status page, click the ID of the job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

To review the Installation Manager license, perform the following steps:
- If you are using the graphical user interface (GUI), run the following command and follow the instructions:
- If you are using the command line, run the following command and follow the instructions:

Submitting jobs to install Installation Manager on remote hosts:

In a flexible management environment, you can submit the **Install IBM Installation Manager** job to install the Installation Manager on registered hosts of the job manager.

Before you begin

Start the job manager and the targets. Ensure that the targets for which you want to install Installation Manager are registered with the job manager.

To submit jobs, your ID at the job manager must be authorized for the administrator role or the operator role. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply to all of the job targets.

To run the job against a large number of targets, optionally create a group of targets and submit the job against the group.

For instructions on updating an existing instance of Installation Manager, see Submitting jobs to update Installation Manager on remote hosts.

Note: CIM jobs to install, uninstall, and update Installation Manager are not supported on z/OS targets. You must first install Installation Manager on z/OS targets before using CIM manage offerings jobs.

About this task

You can use the administrative console of the job manager or the deployment manager to submit the job. From the console, choose the **Install IBM Installation Manager** job, specify the targets, schedule the job, review the summary, and submit the job.

Instead of using a console, you can run the installIM job script in the AdministrativeJobs command group. See the Administrative job types topic.

For Windows targets, CIM sends unzip.exe to the target to unzip the Installation Manager zip file. If you do not want to use unzip.exe from CIM, you can set the JVM parameter:

```
com.ibm.ws.admin.cimjm.unzipOnTheFly=true/TRUE"
```

If this parameter is set to "true", CIM unzips the zip file from the job manager and sends individual files to the target. You must restart the server after changing this parameter.

For Linux/UNIX targets, if CIM detects an instance of unzip, CIM sends the zip file to the target and then unzips the zip file. If CIM does not detect an instance of unzip, CIM unzips the zip file from the job manager and sends individual files to the target. Sending the files individually usually requires more time than unzipping on the target. For IBM i targets, CIM uses the jar command found on the IBM i target to unzip the zip file.

After the centralized installation manager successfully completes the installation process on a remote node, it then deletes the installation image files that are located in the temporary location that you specify during the installation process. If the installation is unsuccessful, the files remain in the temporary location for you to use to determine what caused the installation error. However, you can safely delete the files.

Note: IBM Installation Manager 1.4.3 or above is required.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose the **Install IBM Installation Manager** job and click **Next**.
3. Choose job targets.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. On the Specify the job parameters page, specify the location of the Installation Manager instance that you want to install.

   Note: If you do not specify the IBM Installation Manager installation kit path, the installIM job searches for the most recent IBM Installation Manager installation kit that is suitable for the target platform from the installation kit repository on the Job Manager. By default, the installation kit repository is `<profile_home>/IMKits`. You can change the location from the Job Manager. Click **Jobs** > **Installation Manager installation kits**, then modify 'Installation Manager installation kits location' to a different location. If you are using the command line, you can check for the repository location at: `<profile_home>/properties/cimjm/CIMJMMetadata.xml`.

   Optional parameters:
   - Installation Manager agent data location: specifies the location of the Installation Manager agent data.

Note: The data location cannot be a subdirectory of the installation location.

- Installation Manager installation directory: specifies the location of the Installation Manager installation directory.

If you select the **Skip prerequisite checking** check box, you specify that no prerequisite checking is performed when installing Installation Manager and that Installation Manager disk space checking is disabled.For the job to run successfully, you must select **I accept the terms in the license agreements**. Click **Next**

To review the Installation Manager license, perform the following steps:

Note: Run the install command from the Installation Manager install kit.

- If you are using the graphical user interface (GUI), run the following command and follow the instructions:
- If you are using the command line, run the following command and follow the instructions:

Note: You can install Installation Manager so that it can be used by a group of users. Specify the following optional parameter: **installType**

   Values:
   - single: perform a single user installation in non administrative mode. This option is available for all CIM supported platforms.
   - Auto: the command initiates a single user installation in non administrative mode if your are a non administrative user. If you are an administrator, this action performs an administrative installation.

5. Schedule the job, and click **Next**.
6. Review the summary, and click **Finish** to submit the job.

Results

The job runs and installs Installation Manager on the selected targets.

What to do next

On the Job status page, click the job ID and view the job status. Click the status refresh icon to refresh the displayed job status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

Submitting jobs to update Installation Manager on remote hosts for Version 8.0:

In a flexible management environment, you can submit the **Update IBM Installation Manager** job to update the Installation Manager on registered hosts of the job manager.

Before you begin

Start the job manager and the targets. Ensure that the targets for which you want to update Installation Manager are registered with the job manager.

To submit jobs, your ID at the job manager must be authorized for the administrator role or the operator role. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply all of the job targets.

To run the job against a large number of targets, optionally create a group of targets and submit the job against the group.

Note: CIM jobs to install, uninstall, and update Installation Manager are not supported on z/OS targets. You must first install Installation Manager on z/OS targets before using CIM manage offerings jobs.

To review the Installation Manager license, perform the following steps:

- If you are using the graphical user interface (GUI), run the following command and follow the instructions:
- If you are using the command line, run the following command and follow the instructions:

About this task

You can use the administrative console of the job manager or the deployment manager to submit the job. From the console, choose the **Update IBM Installation Manager** job, specify the targets, schedule the job, review the summary, and submit the job.

Instead of using a console, you can run the updateIM job script in the AdministrativeJobs command group. See the Administrative job types topic.

Note: IBM Installation Manager 1.4.3 or above is required.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose the **Update IBM Installation Manager** job and click **Next**.
3. Choose job targets.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. On the Specify the job parameters page, specify the location of the Installation Manager instance that you want to update and the location of the repository that contains the update. For the job to run successfully, you must select **I accept the terms in the license agreements**. Click **Next**. You can also update Installation Manager using an installation kit. Specify the existing installation location. Select the **Update existing installation** check box. If updating with an Installation Manager installation kit, specify the fully qualified local path and file name of the installation kit. If the field is left blank, the update IBM Installation Manager job will locate and use the most recent IBM Installation Manager installation kit available in the default location for installation kits: $JOB_MANAGER_HOME/IMKit.
5. Schedule the job and click **Next**.
6. Review the summary, and click **Finish** to submit the job.

Results

The job runs and updates Installation Manager on the selected targets.

What to do next

On the Job status page, click the job ID and view the job status. Click the status refresh icon to refresh the displayed job status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

Submitting jobs to uninstall Installation Manager on remote hosts:

In a flexible management environment, you can submit the **Uninstall IBM Installation Manager** job to remove the installation manager from registered hosts of the job manager.

Before you begin

Start the job manager and the targets. Ensure that the targets for which you want to remove Installation Manager are registered with the job manager.

To submit jobs, your ID at the job manager must be authorized for the administrator role or the operator role . When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply all of the job targets.

To run the job against a large number of targets, optionally create a group of targets and submit the job against the group.

Note: CIM jobs to install, uninstall, and update Installation Manager are not supported on z/OS targets. You must first install Installation Manager on z/OS targets before using CIM manage offerings jobs.

About this task

You can use the administrative console of the job manager or the deployment manager to submit the job. From the console, choose the **Uninstall IBM Installation Manager** job, specify the targets, schedule the job, review the summary, and submit the job.

Instead of using a console, you can run the manageOfferings job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure
1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose the **Uninstall IBM Installation Manager** job and click **Next**.
3. Choose job targets.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. On the Specify the job parameters page, specify the location of the Installation Manager instance that you want to uninstall. Click **Next**.
5. Schedule the job and click **Next**.
6. Review the summary, and click **Finish** to submit the job.

Results

The job runs and uninstalls Installation Manager on the selected targets.

What to do next

On the Job status page, click the job ID and view the job status. Click the status refresh icon to refresh the displayed job status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

Submitting jobs to install SSH public keys on remote hosts:

In a flexible management environment, you can submit the **Install SSH Public Key** job to install SSH public keys on registered hosts of the job manager.

Before you begin

Start the job manager and the targets. Ensure that the targets for which you want to install an SSH public key are registered with the job manager.

To submit jobs, your ID at the job manager must be authorized for the administrator role or the operator role. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply all of the job targets.

To run the job against a large number of targets, optionally create a group of targets and submit the job against the group.

Note: IBM Installation Manager 1.4.3 or above is required.

About this task

You can use the administrative console of the job manager or the deployment manager to submit the job. From the job manager console, choose the **Install SSH Public Key** job, specify the targets, schedule the job, review the summary, and submit the job.

Instead of using a console, you can run the Install SSH Public Key job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure
1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose the **Install SSH Public Key** job and click **Next**.
3. Choose job targets.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. On the Specify the job parameters page, specify the location of the public key file that you want to install on the selected target. Click **Next**.
5. Schedule the job and click **Next**.
6. Review the summary, and click **Finish** to submit the job.

Results

The job runs and installs a public key file on the selected targets.

What to do next

On the Job status page, click the job ID and view the job status. Click the status refresh icon to refresh the displayed job status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

Installing the Version 8.0 product using the job manager and administrative console:

In a flexible management environment, you can use the job manager to install, update, and uninstall IBM WebSphere Application Server using the graphical user interface.

Before you begin

Note: This topic applies to WebSphere Application Server Version 8.0 only. For information about using centralized installation manager (CIM) for Version 6.1.x and 7.x, see Getting started with the centralized installation manager (CIM) for previous versions.

Ensure that you have the administrative console installed on your primary machine.

Note: CIM jobs to install, uninstall, and update Installation Manager are not supported on z/OS targets. You must first install Installation Manager on z/OS targets before using CIM manage offerings jobs.

About this task

To install WebSphere Application Server, use the administrative console to register your target machine, install IBM Installation manager, and install WebSphere Application Server or other product offerings that are compatible with Installation Manager. Using the administrative console, you can set parameters for the directory in which to install the product on the target machine, specify where to store product data on the target machine, and specify the URL of the repository to download the product from. Depending on your security setup, you can also specify keyring credentials to log in to the product repository.

After the centralized installation manager successfully completes the installation process on a remote node, it then deletes the installation image files that are located in the temporary location that you specify during the installation process. If the installation is unsuccessful, the files remain in the temporary location for you to use to determine what caused the installation error. However, you can safely delete the files.

Note: IBM Installation Manager 1.4.3 or above is required.

Procedure

1. Start the job manager. See Starting the job manager.
2. Register a host with the job manager. Before you can install the product on a target machine, you must register it with the job manager. For more information, see Register or unregister with job manager settings.
3. Launch the administrative console. For more information, read about the Administrative console.
4. Test the connection to the targets on which you want to install the product. This step is optional. Before you install the product on a target machine, you can test the connection.
   a. In the administrative console, select **Job** > **Submit**.
   b. In the Job type menu list, select **Test connection**. Click **Next**.

c. Specify the target names and target authentication.

- If you test the connection without specifying credentials, the test will use default to existing credentials.
- You can submit the **Test connection** job with a user name and password.
- You can submit the **Test connection** job with a user name and private key file.

5. Optionally run an inventory job. To see what is installed on your target machine, you can run an inventory job.

a. In the administrative console, select **Job** > **Submit**.

b. In the job type menu list, select **Inventory**. Click **Next**.

c. Specify the target names and target authentication.

- You can submit an inventory job with a user name and password.
- You can submit an inventory job without a user name and password.

6. Install or update Installation Manager on your target machine. This step is optional. If you already have the correct version of Installation Manager on your target machine, you can proceed to the next step. For more information, see Managing Installation Manager using the job manager. This step does not apply to zOS targets.

7. If you use secure shell (SSH) security, install your public key file. You can install the public key file using the same credentials as the job manager. This step does not apply to IBM i targets.

a. In the administrative console, select **Job** > **Submit**.

b. In the job type drop down menu, select **Install SSH Public Key**. Click **Next**.

c. Specify the job parameters.

8. Install the product. Use the manageOfferings job to install the product.

a. In the administrative console, select **Job** > **Submit**.

b. In the job type drop down menu, select **Manage offerings**. Click **Next**.

c. Specify the following optional or required job parameters.

   Required parameter:

   - Response file path name: The full path name to the response file on the job manager machine.

   Optional parameters:

   - IBM Installation Manager Path: Specify the path to install Installation Manager on the remote machine. If this parameter is blank, then Installation Manager is installed to the default location.
   - IBM Installation Manager key ring file: If the package repository requires a key ring file for authentication, specify the full path name of the key ring file on the job manager machine.
   - Key ring file password: If the key ring file is password protected, specify the key ring password.
   - IBM Installation Manager agent data location: Specify an IBM Installation Manager data location that is not the default location for the manageOfferings job.

     Note: Do not use a non-default data location unless you are familiar with IBM Installation Manager.

d. Select **I accept the terms in the license agreements**.

9. Optionally transfer files to or from the target machine. For example, if the installation fails, you might want to transfer the log files from the target machine to understand why the job failed.

- To collect a file from remote hosts:

a. In the administrative console, select **Job** > **Submit**.

b. In the job type menu list, select **Collect file**. Click **Next**.

c. Specify the job parameters.

   – The destination location is `<profile home>/config/temp/JobManager/<task id>/<host name>`.

- To distribute a file to remote hosts:
  a. In the administrative console, select **Job** > **Submit**.
  b. In the job type menu list, select **Distribute file**. Click **Next**.
  c. Specify the job parameters.
     – The source location must be `<profile home>/config/temp/JobManager`.
- To delete a file on remote hosts:
  a. In the administrative console, select **Job** > **Submit**.
  b. In the job type menu list, select **Remove file**. Click **Next**.
  c. Specify the job parameters.
10. Create a profile for the newly installed product on the target machine.
    a. In the administrative console, select **Job** > **Submit**.
    b. In the job type menu list, optionally select **Manage Profiles**. Click **Next**.
    c. Choose the job targets.
    d. Specify the job parameters.
       - wasHome: The directory where you installed the product on the target machine
       - responseFile: The response file used to create an IBM WebSphere Application Server profile

Results

You have installed WebSphere Application Server on a target machine and created a profile using the job manager.

What to do next

Using the job manager, you can run any command or script on your target machine.
1. In the administrative console, select **Job** > **Submit**.
2. In the job type drop down menu, select **runCommand**. Click **Next**.
3. Specify the job parameters.

You can uninstall Installation Manager using the administrative console. For more information, see Managing Installation Manager using the job manager.

Installing the Version 8.0 product using the job manager and command line:

In a flexible management environment, you can use the job manager to install, update, and uninstall IBM WebSphere Application Server using the command line with a response file.

Before you begin

Note: This topic applies to WebSphere Application Server Version 8.0 only. For information about using centralized installation manager (CIM) for Version 6.1.x and 7.x, see Getting started with the centralized installation manager (CIM) for previous versions.

Before you install WebSphere Application Server using the job manager, ensure that you have WebSphere Application Server Version 8.0 installed on your primary machine.

Note: CIM jobs to install, uninstall, and update Installation Manager are not supported on z/OS targets. You must first install Installation Manager on z/OS targets before using CIM manage offerings jobs.

About this task

To install WebSphere Application Server, use wsadmin to run the manageOfferings command. The manageOfferings command uses a response file and a security keyring. In the response file, you can set parameters for the directory in which to install the product on the target machine, specify where to store product data on the target machine, and specify the URL of the repository to download the product from. Depending on your security setup, you can also specify keyring credentials to log in to the product repository.

After the centralized installation manager successfully completes the installation process on a remote node, it then deletes the installation image files that are located in the temporary location that you specify during the installation process. If the installation is unsuccessful, the files remain in the temporary location for you to use to determine what caused the installation error. However, you can safely delete the files.

Note: IBM Installation Manager 1.4.3 or above is required.

Procedure

1. Start the job manager. For detailed instructions, see starting the job manager.
2. Register a host with the job manager. Before you can install the product on a target machine, you must register it with the job manager. Use the wsadmin tool to run the registerHost command.
   - You can register the host with a private key; for example:
     - Using Jacl:
       ```
       $AdminTask registerHost  {-host hostname -hostProps
       {{privateKeyFile filename} {username root }{saveSecurity true}}}
       ```
     - Using Jython:
       ```
       AdminTask.registerHost('[-host hostname -hostProps
       [[username user][privateKeyFile filename][saveSecurity true]]]')
       ```
   - You can register the host with a user name and password; for example:
     - Using Jacl:
       ```
       $AdminTask registerHost {-host hostname -hostProps { {password xxxxx}
       { username root } {saveSecurity true}}}
       ```
     - Using Jython:
       ```
       AdminTask.registerHost('[-host hostname -hostProps [[password xxxxx][username user]
       [saveSecurity true]]]')
       ```
3. Optional: Test the connection to the targets on which you want to install the product. Before you install the product on a target machine, you can test the connection.
   - If you test the connection without specifying credentials, the test will use default to existing credentials; for example:
     - Using Jacl:
       ```
       $AdminTask submitJob {-jobType testConnection -targetList {hostname}}
       ```
     - Using Jython:
       ```
       AdminTask.submitJob('-jobType testConnection -targetList [hostname]')
       ```
   - You can submit the Test connection job with a username and password; for example:
     - Using Jacl:
       ```
       $AdminTask submitJob {-jobType testConnection -targetList
       {hostname} -username username -password password}
       ```
     - Using Jython:
       ```
       AdminTask.submitJob('-jobType testConnection -targetList
       [hostname]  -username username -password password')
       ```
   - You can submit the Test connection job with a user name and private key file; for example:
     - Using Jacl:

```
$AdminTask submitJob {-jobType testConnection -targetList
{hostname} -username username -privateKeyFile private_key_filename}
```

– Using Jython:

```
AdminTask.submitJob('-jobType testConnection -targetList
[hostname] -username username -privateKeyFile C:\temp\private_key_filename')
```

4. Optionally run an Inventory job to see what is installed on your target machine.

   a. Submit an Inventory job with a user name and password.

      • Using Jacl:

      ```
      $AdminTask submitJob {-jobType inventory -targetList {hostname}
      -username username -password password}
      ```

      • Using Jython:

      ```
      AdminTask.submitJob('-jobType inventory -targetList [hostname]
      -username username -password password')
      ```

   b. Submit an Inventory job without a user name and password.

      • Using Jacl:

      ```
      $AdminTask submitJob {-jobType inventory -targetList {hostname}}
      ```

      • Using Jython:

      ```
      AdminTask.submitJob('-jobType inventory -targetList [hostname]')
      ```

5. Optional: Install or update Installation Manager on your target machine.

   If you already have the correct version of Installation Manager on your target machine, you can proceed to the next step. For more information, see managing Installation Manager using the job manager.

6. If you use SSH security, install your public key file.

   You can install the public key file using the same credentials as the job manager. This step does not apply to IBM i targets.

   a. Run the installSSHPublicKey admin task; for example:

      • Using Jacl:

      ```
      $AdminTask submitJob {-jobType installSSHPublicKey -targetList {target}
      -jobParams { {publicKeyFile keyfilepath} } -description "test installSSHPublicKey"}
      ```

      • Using Jython:

      ```
      AdminTask.submitJob ('-jobType installSSHPublicKey -targetList [target]
      -jobParams [[publicKeyFile keyfilepath]] -description "test installSSHPublicKey"')
      ```

7. Set up a response file for the manageOfferings command.

   a. Create a response file. You can create a response file using the Installation Manager. For more information, see creating a response file with Installation Manager.

   b. You can edit the response file to include information about your target machine.

   c. You can use the response file to install any offering that is compatible with Installation Manager. For more information, see the Installation Manager information center.

   a. Save the response file as `filename.txt`.

8. Run the manageOfferings command. For the job to run successfully, you must specify `acceptLicense` TRUE.

   a. Open wsadmin from the job manager profile bin directory.

   b. Enter the manageOfferings command in wsadmin. For example:

      • Using Jacl:

      ```
      $AdminTask submitJob {-jobType manageOfferings -targetList hostname -username user -password *******
      -jobParams
      {{responseFile <RESPONSE FILE LOCATION>} {acceptLicense TRUE} {IMPath <IM install location>}
      {keyringFile <key ring file location>} {keyringPassword pwd} }}
      ```

      • Using Jython:

      ```
      AdminTask.submitJob ('-jobType manageOfferings -targetList hostname -username user -password *******
      -jobParams
      [[responseFile <RESPONSE FILE LOCATION>] [acceptLicense TRUE][IMPath <IM install location>]
      [keyringFile <key ring file location>] [keyringPassword pwd]]')
      ```

The manageOfferings command pulls the response file that you created in this task and begins the product installation.

The following parameter for this job is required:

- responseFile: (Response file path name) This parameter contains the full path name to the offering response file on the job manager machine.

The following parameters for this job are optional:

a. IMPath: (IBM Installation Manager Path) This parameter contains the full path of the IBM installation manager on the remote machine. Use this parameter if you have more than one instance of Installation Manager on your remote machine. If you have only one instance of Installation Manager installed, you can leave this parameter empty because the job can find it. Specify whether the target machine has more than one instance of Installation Manager installed.

b. keyringFile: (IBM Installation Manager key ring file): If the package repository requires a key ring file for authentication, specify the full path name of the key ring file on the job manager machine.

c. keyringPassword: (Key ring file password If the key ring file is password protected, specify the key ring password.

9. Optional: Run the collectFile and distributeFile administrative tasks.

   Optionally transfer files to or from the target machine and delete files on the target machine. For example, if the installation fails, you might want to transfer the log files from the target machine to understand why the job failed. When using these administrative tasks, you can specify wildcards in the filename.

   Note: The destination must be a directory, it cannot be a file.

   - To collect a file from remote hosts:
     - Using Jacl:
       ```
       $AdminTask submitJob {-jobType collectFile -targetList hostname -jobParams
       {{source D:\\WAS80\\logs\\manageprofiles\\response.log} {destination log}}}
       ```
     - Using Jython:
       ```
       AdminTask.submitJob('-jobType collectFile -targetList hostname -jobParams
       [[source D:\\WAS80\\logs\\manageprofiles\\response.log] [destination log]')
       ```
   - To distribute a file to remote hosts:
     - Using Jacl:
       ```
       $AdminTask submitJob{-jobType distributeFile -targetList hostname
       -jobParams {{source test.txt}{destination D:\\temp\\test.txt} }}
       ```
     - Using Jython:
       ```
       AdminTask.submitJob('-jobType distributeFile -targetList hostname
       -jobParams [[source test.txt][destination D:\\temp\\test.txt] ]')
       ```
   - To delete a file on remote hosts:
     - Using Jacl:
       ```
       $AdminTask submitJob{-jobType removeFile -targetList hostname
       -jobParams {{location D:\\temp\\test.txt}}}
       ```
     - Using Jython:
       ```
       AdminTask.submitJob('-jobType removeFile -targetList hostname
       -jobParams [[location D:\\temp\\test.txt] ]')
       ```

10. Create a profile for the newly installed product on the target machine.

    Restriction: This step does not apply to z/OS targets.

    Specify the following parameters:

    - targetList: The machine where you want to create a new profile
    - wasHome: The directory where you installed the product on the machine that is running job manager

- responsefile: Enter the directory where you saved your response file. This text file provides the parameters and information of the profile to create.

For example:

- Using Jacl:

```
$AdminTask submitJob {-jobType manageprofiles -targetList hostname
-jobParams {{wasHome D:\\WAS70GA} {responseFile D:\\temp\\mp1.txt}}}
```

- Using Jython:

```
$AdminTask submitJob {-jobType manageprofiles -targetList hostname
-jobParams {{wasHome D:\\WAS70GA} {responseFile D:\\temp\\mp1.txt}}}
```

Results

You have installed the product on a target machine and created a profile using the job manager.

What to do next

Using the job manager, you can run any command or script on your target computer.

- Using Jacl:

```
$AdminTask runCommand {-host hostname -jobParams
{{command command_to_run}{workingDir working_directory_on_remote_host}}}
```

- Using Jython:

```
$AdminTask.runCommand ('-host hostname -jobParams
[[command command_to_run][workingDir working_directory_on_remote_host]]')
```

Managing Installation Manager using the job manager:

You can store and manage all of your installation manager installation kits from a central location.

Before you begin

Before you can work with IBM Installation Manager, you must register at least one host with the job manager. You must also have acquired one or more Installation Manager installation kits.

Note: CIM jobs to install, uninstall, and update Installation Manager are not supported on z/OS targets. You must first install Installation Manager on z/OS targets before using CIM manage offerings jobs.

Note: IBM Installation Manager 1.4.3 or above is required.

About this task

If you have multiple Installation Manager offerings or need to manage Installation Manager on multiple remote machines, the job manager can automate this process. Job manager can also store your Installation Manager installation kits in a single repository. This allows you to manage your installation kits from a single location and send your installation kits to multiple machines.

Procedure

- You can submit an inventory job to see what is installed on a host.
  - You can submit and inventory job with a username and password; for example:
    - Using Jacl:

```
$AdminTask submitJob {-jobType inventory -targetList {hostname} -username username -password password}
```

    - Using Jython:

```
AdminTask.submitJob('-jobType inventory -targetList [hostname] -username user -password xxxxxx')
```

  - If you saved user credentials while registering host, you can submit an inventory job without credentials; for example:

- Using Jacl:

```
$AdminTask submitJob {-jobType inventory -targetList {hostname} }
```

- Using Jython:

```
AdminTask.submitJob('-jobType inventory -targetList [hostname] ')
```

- You can browse the Installation Manager installation kit directory and change the directory location. Perform this task using the administrative console graphical user interface. Open the administrative console and select **Submit Jobs** > **Installation Manager installation kits**. For more information, see Installation Manager installation kits.

- You can submit a job to install Installation Manager on a host using the administrative console.

  1. In the administrative console, select **Job** > **Submit**.

  2. In the job type menu list, select **Install IBM Installation Manager**. Click **Next**.

  3. Specify the job parameters. The **Install Action** menu has the following options:

     – Install based on login credentials

     – Install for single user only

     – Install for a group of users

- You can submit a job to install Installation Manager on a host by sending the installation kit from the command line.

  The installIM job has the following required parameters:

  – kitPath: Specify the full path name to the IBM Installation Manager kit on the job manager machine.

  – acceptLicense: Must be set to true, if you do not specify this parameter, the job will fail.

  The installIM job has the following optional parameters:

  – **installPath**: Specify the path to install Installation Manager on the remote machine. If this parameter is not specified, then Installation Manager is installed to the default location.

  – **dataPath**: Specify the Installation Manager data path on the remote machine. If this parameter is not specified, the default Installation Manager data path is used.

  – Submit the install Installation Manager job without credentials; for example:

    - Using Jacl:

```
$AdminTask submitJob {-jobType installIM -targetList {hostname} -jobParams { {installPath <path>}
{dataPath <path>} {kitPath <path>} {acceptLicense true} } -description "IM install without username"}
```

    - Using Jython:

```
AdminTask.submitJob ('-jobType installIM -targetList [hostname] -jobParams [[installPath <path>]
[dataPath <path>] [kitPath <path>] [acceptLicense true]] -description "IM install without username"')
```

  – Submit the install Installation Manager job using a private key; for example:

    - Using Jacl:

```
$AdminTask submitJob {-jobType installIM -targetList {hostname} -jobParams {
{installPath <path>} {dataPath <path>} {kitPath <path>} {acceptLicense true} }
-privateKeyFile "<key file path>" -description "IM install with private key"}
```

    - Using Jython:

```
AdminTask.submitJob ('-jobType installIM -targetList [hostname] -jobParams [
[installPath <path>] [dataPath <path>] [kitPath <path>] [acceptLicense true] ]
-privateKeyFile '<key file path>' -description "IM install with private key"')
```

  – Submit the install Installation Manager job using a user name and password; for example:

    - Using Jacl:

```
$AdminTask submitJob {-jobType installIM -targetList {hostname} -jobParams { {installPath <path>}
{dataPath <path>} {kitPath <path>} {acceptLicense true} } -username root -password abcd
-description "IM install with username and pwd"}
```

    - Using Jython:

```
AdminTask.submitJob ('-jobType installIM -targetList [hostname] -jobParams [[installPath <path>]
[dataPath <path>] [kitPath <path>] [acceptLicense true] ] -username root -password abcd
-description "IM install with username and pwd"')
```

- You can review the Installation Manager license.

  – If you are using the graphical user interface (GUI), run the following command and follow the instructions:

- If you are using the command line, run the following command and follow the instructions:
- You can submit a job to update Installation Manager on a host by providing an Installation Manager repository URL from the command line. This job has the following required parameter:
  - acceptLicense: Must be set to true, if you do not specify this parameter, the job will fail.

    For example:
  - Using Jacl:

```
$AdminTask submitJob {-jobType updateIM -targetList {hostname} -jobParams { {installPath <path>}
{repository <repository URL>} {keyringFile <file path>} {keyringPassword <keyringpwd>} {acceptLicense true} }
-username root -password <password> -description "update IM with username and pwd"}
```

  - Using Jython:

```
AdminTask.submitJob('-jobType updateIM -targetList [hostname] -jobParams [ [installPath <path>]
[repository <repository URL>] [keyringFile <file path>] [keyringPassword]  [acceptLicense true] ]
-username <username> -password <password>')
```

- You can submit a job to update Installation Manager on a host using the administrative console.
  1. In the administrative console, select**Job** > **Submit**.
  2. In the job type menu list, select **Update IBM Installation Manager**. Click **Next**.
  3. Specify target names and target authentication.
  4. Specify the job parameters and accept the license agreement:
     - installPath: IBM Installation Manager installation location.
     - repository: IBM Installation Manager repository.
     - keyringFile: IBM Installation Manager key ring file, the credentials for the protected repository are retrieved from the key ring file.
     - keyringPassword: Password for accessing key ring file.
- You can delete Installation Manager installation kits from the repository. Perform this task using the administrative console graphical user interface. Open the administrative console and select **Jobs** > **Installation Manager installation kits**. For more information, see Installation Manager installation kits.
- You can submit a job to uninstall IBM Installation Manager. For example:
  - Using Jacl:

```
$AdminTask submitJob {-jobType uninstallIM -targetList {hostname} -jobParams { {installPath <IM install path>}}}
```

  - Using Jython:

```
AdminTask.submitJob('-jobType uninstallIM -targetList [hostname] -jobParams [ [installPath <IM install path>] ]]')
```

- You can submit a job to uninstall Installation Manager using the administrative console.
  1. In the administrative console, select **Jobs** > **Submit**.
  2. In the job type menu list, select **Uninstall IBM Installation Manager**. Click **Next**.
  3. Specify target names and target authentication.
  4. Specify the job parameters.
     - The following parameter is required: installPath, IBM Installation Manager installation location.
- You can submit a job to find Installation Manager data locations. You can add specific data locations, or search the system for Installation Manager data locations.
  1. In the administrative console, select **Jobs** > **Submit**.
  2. In the job type menu list, select **Add or search for Installation Manager data locations**. Click **Next**.
  3. Specify target names and target authentication.
  4. Specify the job parameters.
     - You can specify Installation Manager data locations.
     - You can search the system for Installation Manager data locations.

Results

You have installed, updated, or deleted Installation Manager and Installation Manager installation kits on a target machine.

What to do next

You can continue to view node resources and do other job management tasks such as submit jobs, create node groups for job submission, and view nodes.

Submitting jobs to manage applications

In a flexible management environment, you can submit jobs to install and administer enterprise applications on managed targets of the job manager and to uninstall files from targets. An enterprise application is an enterprise archive (EAR) file that conforms to Java Platform, Enterprise Edition (Java EE) specifications.

Before you begin

Before submitting a job, start the job manager and the target targets. If a target is a stand-alone application server, also start the administrative agent.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must be applicable to all of the job targets.

Before you can install an application, you must run a job that copies the enterprise application file to managed targets. Remember any destination value that is specified when distributing the file. See the topic on the distribute file job.

About this task

The topics in this section describe how to install and administer enterprise application files by running jobs in the job manager console or the deployment manager console.

Instead of using a console, you can run wsadmin commands in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

- Run the Install application job to deploy an enterprise application on managed targets.
- Run the Start application job.
- Run the Stop application job.
- Run the Update application job.
- Run the Uninstall application job to remove an enterprise application from managed targets.

What to do next

On the Job status page, click the ID of the job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

Installing applications using the job manager:

In a flexible management environment, you can submit the **Install application** job to deploy an enterprise application on managed application server targets of the job manager. An enterprise application is an enterprise archive (EAR) file that conforms to Java Platform, Enterprise Edition (Java EE) specifications.

Before you begin

Before installing an enterprise application on an application server, do the following:

- Develop and assemble the EAR file. The **Install application** job does not change existing Java Naming and Directory (JNDI) and other application bindings. If you must set binding values during deployment, do not use the job manager to deploy the application. Instead, use a deployment tool such as the administrative console application installation wizard or a wsadmin script and specify managed targets of the job manager as deployment targets.
- Start the job manager and the targets. Each server on which you are installing an application must be running. If a target is a stand-alone application server, also start the administrative agent.
- Ensure that the targets are compatible with the application. Version 7.0 or later targets support EAR files that have Java 2 Platform, Enterprise Edition (J2EE) 1.3, J2EE 1.4, or Java EE 5 modules.
- Copy the EAR file to the `/config/temp/JobManager` directory of the job manager profile.

  If the `JobManager` directory does not exist, create the `JobManager` directory in the job manager profile `/config/temp` directory. To create and access the directory, you must have the appropriate authority.

  If the EAR file exists on a managed target, you can run the **Collect file** job to copy the EAR file from the managed target to the *job_manager_profile*/config/temp/JobManager/*jobToken*/*targetName* directory. See the topic on the collect file job.
- Run the **Distribute file** job to copy the EAR file to managed targets. Remember any destination value that is specified when distributing the file.

  Note: You must distribute an EAR file to the targets before you can run the **Install application** job. The distribute file job copies the EAR file in the `/config/temp/JobManager` directory of the job manager profile to the targets. The name of the EAR file on the targets becomes whatever value that you specify for the destination when distributing the file. See the topic on the distribute file job.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that deploys an enterprise application file on selected targets. From the console, choose the **Install application** job, specify the file to install, specify job options, schedule the job, review the summary, and submit the job.

Instead of using a console, you can run the installApplication job script in the AdministrativeJobs command group to deploy the application file to targets. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to install an application file.
   a. Select the **Install application** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets on which you want to deploy the application file.
   a. Select a group of targets from the list, or select **Target names**.

      Only groups of targets that are valid for the job type that you selected are displayed in the list of target groups.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.

d. Click **Next**.

4. Specify parameters for the install application job.

   a. For **Application name**, specify the base file name of the application file.

      By default, the job appends the `.ear` file extension to the application name. Suppose that you distributed the `DynaCacheEsi.ear` file as described in the topic on distributing files from the job manager to targets. To continue with the `DynaCacheEsi.ear` example in that topic, specify the name of the application file without the `.ear` extension:

      `DynaCacheEsi`

   b. If you specified a **Destination** value other than the EAR file name when distributing the application, specify that destination value for **Application location**.

      By default, the job searches for the application in the default destination location. If you do not specify an **Application location** value, then the location defaults to `downloadedContent/`*`application_name`*`.ear` of the managed target. Thus, if you specified `DynaCacheEsi` for **Application name**, the application location defaults to `downloadedContent/DynaCacheEsi.ear`.

      When distributing the file, if you specified a value for **Destination** other than the EAR file name, specify the **Destination** value for **Application location**.

      For the `DynaCacheEsi.ear` example, you did not specify the EAR file name, `DynaCacheEsi.ear`, for the destination value when distributing the EAR file. You specified `dynacache_esi_sample` for the destination value. Thus, the application location is `downloadedContent/dynacache_esi_sample`. To enable the targets to find the application, specify the destination value:

      `dynacache_esi_sample`

   c. If you are installing the application on a federated node of a deployment manager, specify the target server, node, or cluster names.

      For **Server name**, click **Find** and specify the target server. Based on your selection, the product fills in values for the server and node names.

      If the target is a cluster, for **Cluster name**, click **Find** and specify the target cluster.

   d. Click **Next**.

5. Schedule the job.

   The times and dates that you specify are relative to the job manager.

   a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

      If you specify multiple email addresses, separate them with commas.

   b. Select when the job is available for submission.

   c. Select the job expiration.

   d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

   e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to install the file.

What to do next

On the Job status page, click the ID of the install application job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the installation is successful, the application name is shown in the list of target resources. Click **Jobs** > **Target resources** to see the list of resources.

After application installation, you can run jobs that administer the application or file:
- Start application
- Stop application
- Update application
- Uninstall application
- Remove file

Starting applications using the job manager:

In a flexible management environment, you can submit the **Start application** job to start enterprise applications that are deployed on managed targets of the job manager.

Before you begin

Start the job manager and the targets. If a target is a stand-alone application server, also start the administrative agent.

The application that you want to start must be installed on a managed target and not be running.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that starts the running of a deployed application file. From the console, choose the **Start application** job, specify the file to start, specify job options, review the summary, and submit the job.

Starting the application changes the application status to active, loads the application in the run time, and opens the application to receive client requests.

Instead of using a console, you can run the startApplication job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure
1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to start applications on managed targets.
   a. Select the **Start application** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets on which to start the application.
   a. Select a group of targets from the list, or select **Target names**.

      Only groups of targets that are valid for the job type that you selected are displayed in the list of target groups.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the start application job.
   a. For **Application name**, specify the base file name of the application file.

Click **Find**. On the Find target resources page, select the application resource to start. See the topic on installing applications using the job manager. To continue with the `DynaCacheEsi.ear` example in that topic, find the application resource and specify the following application name:

`DynaCacheEsi`

b. Click **Next**.

5. Schedule the job.

The times and dates that you specify are relative to the job manager.

a. Optionally specify one or more email addresses where notifications are sent when the job finishes. If you specify multiple email addresses, separate them with commas.

b. Select when the job is available for submission.

c. Select the job expiration.

d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to start the application.

What to do next

On the Job status page, click the ID of the start application job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, the status on the Target resources page of the application is `Started`. Click **Jobs** > **Target resources** > *resource_name* to see the resource status.

Stopping applications using the job manager:

In a flexible management environment, you can submit the **Stop application** job to stop enterprise applications that are deployed and running on managed targets of the job manager.

Before you begin

The application that you want to stop must be running on managed target of the job manager.

Start the job manager if it is not already running. If a target is a stand-alone application server, also start the administrative agent.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that stops the running of a deployed application. From the console, choose the **Stop application** job, specify the application to stop, specify job options, review the summary, and submit the job.

Stopping the application changes the application status to stopped. The application can no longer receive client requests.

Instead of using a console, you can run the stopApplication job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to stop applications on managed targets.
   a. Select the **Stop application** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets on which to stop the application.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify job parameters.
   a. For **Application name**, specify the base file name of the application file.

      Click **Find**. On the Find target resources page, select the application resource to stop. See the topics on installing and starting applications using the job manager. To continue with the `DynaCacheEsi.ear` example in those topics, find the application resource and specify the following application name:

      `DynaCacheEsi`
   b. Click **Next**.
5. Schedule the job.
6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to stop the application.

What to do next

On the Job status page, click the ID of the stop application job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, the status on the Target resources page of the application is `Stopped`. Click **Jobs** > **Target resources** > *resource_name* to see the resource status.

After stopping the application, you can run the following jobs:
- Start application
- Update application
- Status
- Uninstall application
- Remove file

Updating applications using the job manager:

In a flexible management environment, you can submit the **Update application** job to upgrade all or part of an enterprise application that is deployed on managed targets of the job manager.

Before you begin

Before running the job, do the following:

1. Start the job manager and the targets. Each server on which you are updating an application must be running. If a target is a stand-alone application server, also start the administrative agent.
2. Run the **Distribute file** job for the application file that replaces or upgrades the currently installed application file. You might specify a different destination value for this job than the value for the currently installed file. Remember any destination value that is specified when distributing the file.
3. Run the **Stop application** job to stop the currently installed application file that you want to update.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that updates a deployed application file on selected targets. From the console, choose the **Update application** job, specify the file to upgrade, specify job options, schedule the job, review the summary, and submit the job.

Instead of using a console, you can run the updateApplication job script in the AdministrativeJobs command group to deploy the application file to targets. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to update an application file.
   a. Select the **Update application** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets on which you want to deploy the application file.
   a. Select a group of targets from the list, or select **Target names**.

      Only groups of targets that are valid for the job type that you selected are displayed in the list of target groups.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify parameters for the update application job.
   a. For **Application name**, specify the base file name of the application file.

      By default, the job appends the `.ear` file extension to the application name.

      Click **Find**. On the Find target resources page, select the application resource that upgrades the currently installed application file.

      Suppose that you want to upgrade the `DynaCacheEsi.ear` file used as an example in the topics on distributing files and installing applications. To continue with the `DynaCacheEsi.ear` example, specify the name of the application file without the `.ear` extension:

      `DynaCacheEsi`
   b. If you specified a **Destination** value other than the EAR file name when distributing the application, specify that destination value for **Application location**.

      By default, the job searches for the application in the default destination location. If you do not specify a **Application location** value, then the location defaults to `downloadedContent/`

application_name.ear of the managed target. Thus, if you specified `DynaCacheEsi` for **Application name**, the application location defaults to `downloadedContent/DynaCacheEsi.ear`.

To continue with the `DynaCacheEsi.ear` example, suppose that the destination value for the upgraded file is `dynacache_esi_updated_sample`. Specify the new destination value:

`dynacache_esi_updated_sample`

c. If you are updating the application on a federated node of a deployment manager, specify the target server, node, or cluster names.

For **Server name**, click **Find** and specify the target server. Based on your selection, the product fills in values for the server and node names.

If the target is a cluster, for **Cluster name**, click **Find** and specify the target cluster.

d. Click **Next**.

5. Schedule the job.

The times and dates that you specify are relative to the job manager.

a. Optionally specify one or more email addresses where notifications are sent when the job finishes.

If you specify multiple email addresses, separate them with commas.

b. Select when the job is available for submission.

c. Select the job expiration.

d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to install the application.

What to do next

On the Job status page, click the job ID and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, the status on the Target resources page of the application is `UPDATED`. Click **Jobs** > **Target resources** > *resource_name* to see the resource status.

After updating an application, you can run other jobs that administer the application, such as the start application job.

Uninstalling applications using the job manager:

In a flexible management environment, you can submit jobs to remove deployed applications from managed targets of the job manager.

Before you begin

Start the job manager and the targets. If a target is a stand-alone application server, also start the administrative agent.

Submit the **Stop application** job and stop the running of the application that you want to uninstall. The application must be installed on one or more application servers that are on managed targets of the job manager.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that removes a deployed application from a managed target. From the console, choose the **Uninstall application** job, specify the application to remove, specify job options, review the summary, and submit the job.

Instead of using a console, you can run the uninstallApplication job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to uninstall an application from selected targets.
   a. Select the **Uninstall application** job type from the list.
   b. Optionally specify a description of the job.
   c. Click **Next**.
3. Choose the job targets on which the application resides.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify job parameters.
   a. For **Application name**, specify the base file name of the application file.

      Click **Find**. On the Find target resources page, select the application resource to uninstall. For example, the topic on installing applications using the job manager describes how to install the `DynaCacheEsi.ear` example application. To uninstall that application, find the application resource and specify the following name:

      `DynaCacheEsi`
   b. Click **Next**.
5. Schedule the job.
6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job and attempt to uninstall the application.

What to do next

On the Job status page, click the job ID of the uninstall application job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, click **Jobs** > **Target resources**. The names of deleted applications are not in the list of resources.

After uninstalling the application, if you previously ran the **Distribute file** job to distribute the application to targets, you can run the **Remove file** job to remove the application file from the flexible management environment.

Submitting jobs to collect status on managed servers and applications

In a flexible management environment, you can submit the **Status** job to refresh data on managed resources of the job manager. Resources include applications and servers of each target. Run the **Status** job to refresh data on target resources in the job manager database. You can view the refreshed data in the job manager console or by wsadmin scripts.

Before you begin

Start the job manager and the targets. Ensure that the targets for which you want status are registered with the job manager.

Your ID at the job manager must be authorized for the administrator role or the operator role to submit jobs. When you submit a job, you can specify a user name and password for authentication and authorization at the target or targets. When you submit a job to multiple targets, the user name and password or the credentials for the submitter must apply all of the job targets.

To run the job against a large number of targets, optionally create a group of targets and submit the job against the group.

About this task

You can use the administrative console of the job manager or the deployment manager to submit the job. From the console, choose the **Status** job, specify the targets, schedule the job, review the summary, and submit the job.

Instead of using a console, you can run the status job script in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose the **Status** job and click **Next**.
3. Choose job targets.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. On the Specify the job parameters page, click **Next**. There are no job parameters for the Status job.
5. Schedule the job and click **Next**.
6. Review the summary, and click **Finish** to submit the job.

Results

The job runs and refreshes data on applications and servers of the targets.

What to do next

On the Job status page, click the job ID and view the job status. Click the status refresh icon ![icon] to refresh the displayed job status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, the data on the Target resources page is updated. Click **Jobs** > **Target resources** to see the resource status.

Submitting jobs to collect inventory data

In a flexible management environment, you can submit the **Inventory** job to refresh data on job types and on managed resources of the job manager. Resources include applications and servers of each target. If you installed a product that adds job types on a managed target, run the **Inventory** job to refresh data on job types and target resources. You can view the refreshed data in the job manager console or by wsadmin scripts.

Before you begin

Ensure that the targets for which you want data are registered with the job manager:

- To register a stand-alone application server with the job manager, first use the administrative agent registerNode command to register the stand-alone application server with the administrative agent. Then, use the administrative agent console or registerWithJobManager command to register the stand-alone application server target with the job manager.
- To register a deployment manager with the job manager, use the deployment manager console or the registerWithJobManager command.

Start the job manager and the targets. If a target is a stand-alone application server, also start the administrative agent.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that refreshes data on job types and on applications and servers of managed targets. From the console, choose the **Inventory** job, specify the targets, schedule the job, review the summary, and submit the job.

You do not need to run the **Inventory** job unless you installed a product that adds job types on a managed target. You can run the **Status** job to obtain the same data for managed resources without regathering the job type data.

Instead of using a console, you can run the `inventory` administrative job of the submitJob command in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose the **Inventory** job and click **Next**.
3. Choose job targets.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.

d. Click **Next**.

4. On the Specify the job parameters page, specify job parameters if a target is a host. Only hosts have job parameters for the Inventory job.

   If a target is a host, specify job parameters that provide information on the host such as operating system name, version, Installation Manager path, package group, packages, and profiles.

   Click **Next**.

5. Schedule the job and click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The job manager runs the job and refreshes data on job types and on applications and servers of the targets.

What to do next

On the Job status page, click the ID of the job and view the job status. If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job is successful, the data on the Target resources page is updated. Click **Jobs** > **Target resources** to see the resource status.

Submitting jobs to run wsadmin scripts

In a flexible management environment, you can submit the **Run wsadmin script** job to run commands in a wsadmin script file on managed targets of the job manager. You must distribute the wsadmin script file before you can submit jobs that run the script file.

Before you begin

Before running the **Run wsadmin script** job, do the following:

- Start the job manager and the targets. If a target is a stand-alone application server, also start the administrative agent.
- Develop the wsadmin script file.

   For example, create a script file named `extract_server_props.py` that has the following script in the Jython language:

```
AdminTask.extractConfigProperties('[-propertiesFileName server.props -configData Server=server1 ]')
```

   The script runs the extractConfigProperties command to extract the server configuration properties file of an application server named `server1`. The server configuration properties are written to a file named `server.props`.

   The return code of the script determines the success or failure of a job that runs the script. If the script has a return code of zero (0), the job is successful. If the return code is a nonzero value, the job fails.

   For more information, see topics on wsadmin scripting.

- Copy the script file to the `/config/temp/JobManager` directory of the job manager profile.

   If the `JobManager` directory does not exist, create the `JobManager` directory in the job manager profile `/config/temp` directory. To create and access the directory, you must have the appropriate authority.

   If the script file exists on a managed target, you can run the **Collect file** job to copy the script file from the managed target to the *job_manager_profile*`/config/temp/JobManager/`*jobToken*`/`*targetName* directory. See the topic on the collect file job.

- Run the **Distribute file** job to copy the script file to managed targets. Remember any destination value that is specified when distributing the file. See the topic on the distribute file job.

   For the `extract_server_props.py` script example, you might assign a **Destination** value of `extract_server_props.py`.

About this task

You can use the administrative console of the job manager or the deployment manager to submit a job that runs a wsadmin script file on selected targets. From the console, choose the **Run wsadmin script** job, specify the file, specify job options, schedule the job, review the summary, and submit the job.

Instead of using a console, you can run the runWsadminScript job in the AdministrativeJobs command group. See the Administrative job types topic.

Procedure

1. Click **Jobs** > **Submit** from the navigation tree of the administrative console.
2. Choose a job to run a script.
   a. Select the **Run wsadmin script** job type from the list.
   b. Optionally describe the job.
   c. Click **Next**.
3. Choose the targets on which you want to run the script.
   a. Select a group of targets from the list, or select **Target names**.
   b. If you selected **Target names**, then specify a target name and click **Add**, or click **Find** and specify the chosen targets on the Find targets page.

      For example, suppose that you submitted a job that distributed the `extract_server_props.py` Jython script file. Specify targets that have a server named `server1`. If an administrative agent or deployment manager manages multiple application server targets that have a server named `server1`, specify only one target. When you specify multiple targets, only the last created `server.props` file persists. Previously created `server.props` files are overwritten.
   c. If user authentication is required, specify a user name, password, or any other authentication values as needed.
   d. Click **Next**.
4. Specify job parameters.
   a. For **Script file location**, specify the **Destination** value given when distributing the script file.

      For example, suppose that you submitted a job that distributed the Jython script file `extract_server_props.py` and assigned a **Destination** value of `extract_server_props.py`. For **Script file location**, use the destination value:

      `extract_server_props.py`

      For the `extract_server_props.py` script to run successfully, the targets specified for **Target names** must have a server named `server1`.
   b. For **Profile location**, optionally specify the profile destination value that was given when the file was distributed to the targets.
   c. For **Script parameters**, specify parameters that are needed to run the wsadmin script.

      If a parameter attribute contains any spaces, place double quotation marks (") around the parameter. If a quoted parameter attribute contains imbedded quotes, put a backslash before the imbedded quotes. For the `extract_server_props.py` example, optionally specify that wsadmin use the Jython language:

      `-lang jython`
   d. Click **Next**.
5. Schedule the job.

   The times and dates that you specify are relative to the job manager.
   a. Optionally specify one or more email addresses where notifications are sent when the job finishes.
   b. Select when the job is available for submission.

      You can submit the job to be available now, or specify a time and date that the job is retrieved from the job manager.

c. Select the job expiration.

d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

e. Click **Next**.

6. Review the summary, and click **Finish** to submit the job.

Results

The targets run the job.

What to do next

On the Job status page, click the ID of the run wsadmin script job and view the job status. Click the status refresh icon to refresh the displayed status.

If the job is not successful, view any error messages that result from running the job, correct the error condition, and submit the job again.

If the job status is `Succeeded`, verify that the script ran successfully. For the `extract_server_props.py` example, when the script runs successfully the targets extract the server1 properties to a file named `server.props`. A deployment manager places the file in its main directory, for example, `Dmgr01`. A stand-alone target places the file in the main directory of its administrative agent, for example, `AdminAgent01`.

Find target resources

Use this page to find target resources for resource job parameters when you submit a job through the job submission wizard. This page is used for job types such as start server, stop server, and start cluster.

To access this page, first click **Jobs** > **Submit**. The job type that you select in step 1 of the Job submission wizard affects whether you can access the Find target resources page in step 3. For example, if you select the job type, start server, in step 1, then in step 3 you can select **Find** to access the Find target resources page.

Find:

You can use the Find option to determine the target resources to display. After you click **Find**, the Find results are displayed in **Available resources common to all selected targets**. Click **Reset** to assign the parameters the default values.

Table 16. Find parameters. Specify Find parameters to limit the search for target resources.

Parameter names	Operators	Search strings
Type	The type is preselected to the type of resource option that you need for job submission. You cannot change the type. The list contains one or more values of `Server`, `Application`, or `Cluster`.	Not applicable

Table 16. Find parameters (continued).   Specify Find parameters to limit the search for target resources.

Parameter names	Operators	Search strings
The remaining parameters define target resources and vary depending on the **Type** parameter selected. The target name, job type, and unique identifier parameters are always available.	Valid operators include = (equal to), != (not equal to), `is null`, and `is not null`.	Specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the target resource parameter to `Server*` finds all server type resources that start with `Server`. You can search for an exact match for multiple items by including comma-separated items. For example, you can search on two resource names by entering `server1`, `server2`. When you search for more than one item, you cannot use the asterisk. Target names are included based on your operator choices.
Maximum results	Not applicable	Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration.

Example: If the resources are server1, server2, and server3, you can specify the = operator and the `server` or `server*` search string for the resource name parameter.

Available resources common to all selected targets: Specifies the results of the find operation. If no choices are listed or you do not see the resource you want, you can search again. If your job is targeted to multiple targets, only those resources that are present on all targets will be listed. If at least one resource is listed, you can select one resource and click **OK** to return it to the parameter on the previous console page.

Checking job status

In a flexible management environment, you can check the overall status of jobs, the status of specific job targets, and the job history of targets. You can suspend, resume, or delete jobs on the Job status collection page.

Before you begin

Before you can check the job status, you must have registered at least one target with the job manager and submitted a job for the target.

To suspend a started job, resume a suspended job, or delete selected jobs, your ID must be authorized for the operator role.

About this task

After you submit a job using the job manager console or the deployment manager console, the Job status collection page is displayed. The page contains information about only the job that you submitted. The page shows the unique job ID; for example, 122763380912576341. You can use the job ID to query, suspend, resume, or delete the job. The page also shows the job description, state, time of activation, time of expiration, and status.

If you access the Job status collection page by selecting **Jobs** > **Status** in an administrative console navigation, you can use the Find option to limit the number of jobs that are displayed based on the criteria you specify. The first time you access the Job status collection page, no jobs are listed. You must enter parameters for the Find option to obtain a list of jobs based on the parameter information that you provide. The next time you select **Jobs** > **Status**, a list of jobs is displayed based on the parameters you last specified on the Find option for this job manager administrative console page. You can then optionally

modify the Find option criteria to display a different set of jobs. After at least one job displays, you can check the status of the displayed jobs, the status of specific job targets, and the job history for targets of a particular job.

This topic describes how to use the job manager console to check job status. Instead of using the console, you can run a getJobTargetStatus wsadmin command; for example:

```
AdminTask.getJobTargetStatus('[-jobToken 1227633380912576341]')
```

The job token is the job ID for the submitted job. Run the Jython script command from the `bin` directory of the job manager profile. For more information, see the AdministrativeJobs command group for the AdminTask object topic.

Procedure

- Optionally use the Find option to display a set of jobs.

    If no jobs are displayed, you must use the Find option to display jobs based on the parameter information that you enter.

    1. Click **Jobs** > **Status** in the job manager console.
    2. For the parameters on which you want to do a Find operation, specify an operator and a text string.
    3. Click **Find**.

    The list of jobs along with their status information are in the collection table.

- Check the status of a job at its targets.

    1. Select **Jobs** > **Status** in the console to access the Job status collection page if you did not get to the page as a result of a job submission.
    2. Select either a job from the Job ID column or a number on the graph in the Status Summary column for a particular job. The graph is divided in up to four sections, indicating success, partial success, failure, or other, in that order, of the targets in the job.
    3. Click the status refresh icon [icon] to refresh the displayed status.
    4. Optionally use the Find option to display the status of specific job targets based on the parameter information that you enter.

        a. To run the Find operation on specific parameters, specify an operator and a text string as appropriate.
        b. Click **Find**.

    A list of targets for the job, along with the status for each target, are displayed on the Job status settings page.

- Check the job history of targets.

    1. Select **Jobs** > **Status** in the job manager console to access the Job status collection page if you did not get to the page as a result of a job submission.
    2. Select either a job from the Job ID column or a number on the graph in the Status Summary column for a particular job. The graph is divided in up to four sections, indicating success, partial success, failure, or other, in that order, of the targets in the job.
    3. On the Job status settings page, click a target name link in the Status column.

        The Job status history page is displayed, showing a history of the job processing on a managed target. A typical job history is for the status to progress from Distributed to In progress to Succeeded. Table 1 describes the job status values.

Table 17. Job status descriptions.   The status indicates whether the job completed successfully.

Job status	Description
Not attempted	The target has not received the job. The status is NOT_ATTEMPTED.
Distributed	The target has received the job. The status is DISTRIBUTED.

Table 17. Job status descriptions (continued). The status indicates whether the job completed successfully.

Job status	Description
In progress	The target is running the job concurrently with other jobs. The status is ASYNC_IN_PROGRESS.
Failed	The job failed and is no longer running. The status is FAILED.
Rejected	The target rejected the job because, for example, the target does not support the job type. The status is REJECTED.
Succeeded	Job completed successfully. The status is SUCCEEDED.
Partially succeeded	Applies only to startCluster and stopCluster jobs where the cluster has multiple cluster members and to startApplication and stopApplication jobs where the application is installed on multiple targets. If only some cluster members are started or stopped or the application does not start on all application targets, the status of the job is PARTIALLY_SUCCEEDED.

4. Click the status refresh icon ![icon] to refresh the displayed status.

    Refresh the job status until the status is Failed, Rejected, or Succeeded.

5. Click an output file name to view the contents of a file that provides information about the job processing.

    Not all jobs produce output files.

6. Optionally use the Find option to display job history based on the parameter information that you enter.

    a. To run the Find operation on specific parameters, specify an operator and a text string as appropriate.

    b. Click **Find**.

    The status of the job for the target is displayed on the Job status history page.

- Suspend a job.

    1. Select **Jobs** > **Status** in the job manager console to access the Job status collection page if you did not get to the page as a result of a job submission.

    2. Select the check box next to a job with an active or pending state.

    3. Click **Suspend**.

- Resume a job.

    1. Select **Jobs** > **Status** in the job manager console to access the Job status collection page if you did not get to the page as a result of a job submission.

    2. Select the check box next to a job whose state is Suspended.

    3. Click **Resume**.

- Delete a job.

    By default, submitted jobs remain active for one day (24 hours). An active job is a running Java process that consumes machine resources. Delete jobs that you no longer need. You can use the job manager console Job status page.

    1. Select **Jobs** > **Status** in the job manager console to access the Job status collection page if you did not get to the page as a result of a job submission.

    2. Select the check box next to the job that you want to delete.

    3. Click **Delete**.

Results

You might have run a Find operation to display job status based on criteria that you specify, checked the status of jobs at their targets, checked the jobs history of targets, suspended a job, resumed a job, or deleted a job.

What to do next

You can continue to check job status and do other job management tasks such as submit other jobs, create target groups for job submission, view target resources, or view targets.

Job status collection

Use this page to display information about submitted jobs, including the job ID, description, state, activation time, expiration time, and status summary. Jobs are submitted to administer targets that have been registered with the job manager.

The targets can be host computers, deployment managers, or unfederated application servers that have been registered with an administrative agent. To run jobs on a target, the target must be registered with the job manager.

To view this administrative console page, do either of the following:
- When the job submission wizard completes, this console page is displayed only with the job that you just submitted.
- Click **Jobs** > **Status** to view the status of any of your submitted jobs.

To suspend a started job, resume a suspended job, or delete selected jobs, your ID must be authorized for the operator role.

Use one of the following buttons to suspend, resume, or delete a job.

Table 18. Button descriptions.   Select a job and click a button.

Button	Description
Suspend	Specifies that a target can no longer retrieve the job.
Resume	Specifies that a target can retrieve the job.
Delete	Specifies that a job and all its history are removed and no longer available. When you click **Delete**, you are given an opportunity to confirm or cancel the delete operation.

Find:

Specifies parameters to limit the submitted jobs to display. After you click **Find**, the Find results are displayed in the table that follows the Find and Preference options. Click **Reset** to assign the parameters the default values.

The following table lists Find parameters. Except for the maximum results parameter, the parameters define a job. The maximum results parameter specifies the number of records to display.

Table 19. Find parameters.   Specify Find parameters to limit the search for submitted jobs.

Parameter name	Operators	Search strings
State	Valid operators include `Active`, `Suspended`, `Pending`, and `Expired`. By default, jobs in active state are searched.	Not applicable

Table 19. Find parameters (continued). Specify Find parameters to limit the search for submitted jobs.

Parameter name	Operators	Search strings
Description	Valid operators are = (equal to), != (not equal to), is null, and is not null. The default is =.	Specifies the string or partial string of a parameter.
Activation time	Valid operators are >= and <=. The default is >=.	A partial string is designated using an asterisk (*). For example, to find all jobs with a description that starts with inventory, set the description to inventory*. You cannot use the asterisk in the job ID field or the fields related to time.
Expiration time		
Target name	Valid operators are = (equal to) and != (not equal to). The default is =.	To search for an exact match for multiple items, include the items to match separated by commas. For example, to search on two job IDs, specify 119489625729609463, 119489651801509472.
Target groups	Valid operators are = (equal to), != (not equal to), is null, and is not null. The default is =.	
Job ID	Valid operators are = (equal to) and != (not equal to). The default is =.	When you search on more than one item, you cannot use the asterisk.
Maximum results	Not applicable	Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration. The default is 50.

Example: If the targets are EastCoast1, EastCoast2, WestCoast1, and WestCoast2, you can specify the = operator and the East* search string for the target name parameter. Job status is displayed for the jobs that include the EastCoast1 and EastCoast2 targets.

Job ID: Specifies the job number of the job that you submitted.

Description: Specifies the description that you entered when you submitted the job.

State:

Specifies whether the state of the job is Active, Pending, Expired, or Suspended.

Table 20. Job state descriptions. The state indicates whether the job is active.

State	Description
Active	The job is activated and not expired. The job does not have to be running to be active.
Pending	The job has been submitted, but has not been activated.
Expired	The expiration time that you specified during job submission was reached before the job was started. A job does not start on a target after the expiration time is reached. If the expiration time is reached while the job is running on any of its targets, the job continues on those targets.
Suspended	The job has been suspended. If the suspension occurs while the job is running on a target, the job continues on that target. If the suspended job has not started when the suspension occurs, the job will not start on any of its targets unless someone resumes the job. You can suspend a job from this console page by clicking **Suspend**. You can resume a job from this console page by clicking **Resume**.

Activation time:

Specifies the activation time that you entered when you submitted the job or the actual time when the job was submitted if you did not specify an activation time.

The activation time is the time the job is available to run on the targeted targets or groups of targets.

Expiration time:

Specifies the time the job is to expire. If the job has not started by the expiration time, the job will not start and the state will change to `Expired`. If the expiration time occurs while the job is running on a target, the job continues on that target.

When you submit the job, you can set the expiration date and time, choose the default expiration time option, or specify the amount of time until the job expires.

Status summary:

Graphically provides an overview of how the job is running at its targets. The graph is divided in up to four sections, indicating success, partial success, failure, or other, in that order, of the targets in the job.

Table 21. Job status descriptions. The status indicates whether the job completes processing.

Status	Description
Success	Indicates the number of targets on which the job completed successfully.
Partial success	Indicates the number of targets on which the job partially completed. Partial success can occur, for example, when a target represents multiple servers, and only some of the servers on the target complete successfully.
Failed	Indicates the number of targets on which the job failed to complete.
Other	Indicates the number of targets on which the job has some other status than success, partial success, or failure. A status of other can include targets on which the job is currently running, or targets on which the job has not started.

Job status settings:

Use this page to display the job status such as success, partial success, or failed, at each target of the job. Job information, including the job ID, description, activation time, and expiration time, is also displayed.

To view the status of all the targets for a particular job, click **Jobs** > **Status** > ***job_ID***.

To view only the successful targets, only the failed targets or only targets with a status of other for a particular job, click **Jobs** > **Status** > ***number for the successful, failed or other targets in the status summary***.

Find:

Specifies parameters to limit the targets to display. After you click **Find**, the Find results are displayed in the table that follows the Find option. Click **Reset** to assign the parameters the default values.

Table 22. Find parameters.   Specify Find parameters to limit the targets to display.

Parameter name	Operators	Search strings
Target name	Default operator is = (equal to).	Specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the target name to `AppSrv*` finds all jobs with a target name that starts with `AppSrv`. You can search on an exact match for multiple items by including the items to match separated by commas. For example, you can search on two target names by entering `AppSrv01, AppSrv02`.
Status	Valid operators are `Succeeded`, `Partially succeeded`, `Failed`, `Rejected`, `In progress`, `Distributed`, and `Not attempted`. `In progress` is the same as `ASYNC_IN_PROGRESS` in the Status column. `Not attempted` is the same as `NOT_ATTEMPTED` in the Status column.	Not applicable
Maximum results	Not applicable	The search string specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration. The default is `50`.

Example: If the targets are EastCoast1, EastCoast2, WestCoast1, and WestCoast2, you can specify the = operator and the `East*` search string for the target parameter. Job status is displayed for the jobs that include the EastCoast1 and EastCoast2 targets.

Job ID: Specifies the job number of the job that you submitted.

Description: Specifies the description that you entered when you submitted the job.

Activation time:

The activation time is the time that the job is available to start, but not necessarily the time that the job actually starts.

You entered the activation time when you submitted the job.

Expiration time:

Specifies the expiration time that you entered when you submitted the job.

If the job has not started by the expiration time, the job will not start.

Target Names: Specifies the name of each target that is included in the job.

Status:

Specifies the status of the job at its targets.

Table 23. Job status descriptions.   The status indicates whether the job completes processing.

Status	Description
Succeeded	The job completed successfully on the target.

Table 23. Job status descriptions (continued). The status indicates whether the job completes processing.

Status	Description
Partially succeeded	The job partially completed on the target. Partial success can occur, for example, when a target represents multiple servers, and only some of the servers on the target complete successfully.
Failed	The running of the job on the target failed.
Not attempted	The agent for the target has not received the job.
Distributed	The agent for the target has received the job, but the job has not completed.
In progress	The agent for the target has received the job and is running the job concurrent with other jobs for other targets that belong to the agent.
Rejected	The agent rejected the job because the agent does not support the job type. The rejection can happen if a new target is added to the job group after the job is submitted.

Output Files:

Specifies the names of any output files that contain log messages or other information about the job processing.

For output files that are not binary files, click on a file name to view the contents of the file.

If the output file has one of the following extensions typical of binary files, you cannot link to the file from this page:

.bin	.boot	.car	.cer	.dll	.ear	.exe	.gif	.gz	.iso	.img	.jar
.jceks	.jks	.jpg	.ks	.p12	.rar	.so	.tar	.tif	.war	.zip

Job status history collection:

Use this page to display the job status of the target at various stages of the job for the target.

To view this administrative console page, click **Jobs** > **Status** > *job_ID* > *job_status*.

Click **Previous Records** and **Next Records** to scroll backwards and forwards through the list of records if you are not viewing all records in the job history for the target. More records can be generated as the job progresses. These records are retrieved and included in the job history records as you click **Previous Records** and **Next Records**.

Find:

You can use the Find option to limit the submitted jobs to display. After you click **Find**, the Find results are displayed in the table that follows the Find option. Click **Reset** to assign the parameters the default values.

Table 24. Find parameters. Specify Find parameters to limit the search for submitted jobs.

Parameter names	Operators	Search strings
Job ID	Valid operators for the find operation are = (equal to), != (not equal to), is null, and is not null.	You cannot use the asterisk in the job ID field. You can search on an exact match for multiple items by including the items to match separated by commas. For example, you can search on two job IDs by entering `119489625729609463, 119489651801509472`.
Target name		For the target name parameter, specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the target name to `AppSrv*` finds all jobs with a target name that starts with `AppSrv`.
Time stamp	Valid operators are >=, and <=.	You cannot use the asterisk in the time stamp field. You can search on an exact match for multiple items by including the items to match separated by commas.
Maximum results	Not applicable	The search string specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration. The default is `50`.

Time stamp: Specifies the date and time at which the status is logged for the job running at the target.

Status:

Specifies the status of the job running at the target for various stages of the job. The following table lists valid statuses with a description of each status.

Table 25. Job status descriptions. The status indicates whether the job completes processing.

Status	Description
Succeeded	The job completed successfully on the target.
Partially succeeded	The job partially completed on the target. Partial success can occur, for example, when a target represents multiple servers, and only some of the servers on the target complete successfully.
Failed	The running of the job on the target failed.
Not attempted	The agent for the target has not received the job.
Distributed	The agent for the target has received the job, but the job has not completed.
In progress	The agent for the target has received the job and is running the job concurrent with other jobs for other targets that belong to the agent.
Rejected	The agent rejected the job because the agent does not support the job type. The rejection can happen if a new target is added to the job group after the job is submitted.

Message: Specifies error messages associated with the job when the status is `Failed`.

Administering groups of nodes for the job manager

In a flexible management environment, you can create, modify, delete, and view groups of nodes. Groups of nodes make job submission simpler because you can submit a job for a group of nodes instead of a entering multiple node names for a job submission.

Before you begin

Before you can add a node to a group of nodes, you must have registered at least one node with the job manager.

About this task

Groups of nodes are particularly useful if you submit multiple jobs to the same set of nodes.

The first time you access the Groups of nodes collection panel, no groups of nodes are listed. You must create at least one group. You must then enter parameters for the Find option to obtain a list of groups of nodes based on the parameter information that you provide. The next time you select **Jobs > Groups of nodes**, a list of groups of nodes are displayed based on the parameters you last specified on the Find option for this job manager administrative console panel. You can then optionally modify the Find option criteria to display a different set of groups of nodes. After at least one group of nodes is displayed, you can administer the groups of nodes by doing such tasks as adding and removing members for node groups, or deleting node groups.

Procedure

- Create a group of nodes.
    1. Click **Jobs > Groups of nodes** in the job manager administrative console navigation.
    2. Click **New**.
    3. Enter the name of the group of nodes.
    4. Optionally enter a description.
    5. Optionally add members to or remove members from the group of nodes.

        Members are nodes. You can add members to the group or delete members from the group now or later. You can use the Add option, the Find option, or both options to add the members. Follow the steps on adding to or removing members from a group of nodes.
    6. Click **Apply** to save the changes, and then click **OK** to return to the collection page.
- Optionally use the Find option to display groups of nodes.

    If no groups of nodes are displayed, you must use the Find option to display groups of nodes based on the parameter information that you enter.
    1. Click **Jobs > Groups of nodes** in the job manager administrative console navigation.
    2. Specify a valid operator and a text string.
    3. Click **Find**.
- Optionally add or remove the members in a group of node.

    You can add and remove members from the group of nodes. Members are nodes.
    1. Click **Jobs > Groups of nodes** in the job manager administrative console navigation.
    2. Click one of the names of a group of nodes.
    3. To add a node, use the Add option, the Find option, or both.
        a. To use the Add option, type the name of the node in the **Member** list box, and then click **Add**.
        b. Continue to type the name of a node, and then click **Add** until you have added all the members.
        c. To use the Find option, click **Find**.
            1) Enter criteria for the Find option by adding text for one or more options. For example, specify the node name as `test*` or `test*a`.
            2) After getting the list of nodes in the **Chosen nodes** list, click **OK** to return the list to the Groups of nodes panel.

            You can change the find criteria, and select the Find option multiple times to create the list you want.
    4. To remove a node, select the node, and click **Remove**.
    5. Click **Apply**, and then click **OK**.
- Optionally delete one or more groups of nodes.
    1. Click **Jobs > Groups of nodes** in the job manager administrative console navigation.

2. Select one or more groups of nodes.

3. Click **Delete**.

Results

Depending on the tasks that you completed, you might have created a group of nodes, used the Find option to display groups of nodes, added or deleted members in the group of node, or deleted groups of nodes.

What to do next

You can continue to administer groups of nodes and do other job management tasks such as view nodes, submit jobs, and view node resources.

Target group collection

Use this page to create and view groups of targets. Target groups make job submission easier because you can submit a job for a group of targets instead of a separate job for each target.

To view this administrative console page, click **Jobs** > **Target groups**.

Click **Find** to list target groups.

To create or delete a group, you must be authorized for the configurator role.

Find:

Specifies parameters to limit the groups of targets to display. After you click **Find**, the Find results are displayed in the table that follows the Find and preferences options. Click **Reset** to assign the parameters the default values.

Table 26. Find results. Specify Find parameters to limit the search for target groups.

Parameter name	Operators	Search strings
Group name	Valid operators are = (equal to) and != (not equal to). The default operator is =.	Specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the group name parameter to `Region*` finds all jobs with a group name that starts with `Region`. You can search on an exact match for multiple items by including the items to match separated by commas. For example, you can search on two group names by entering `Region1`, `Region2`. When you search on more than one item, you cannot use the asterisk.

The default search string for group name is `*`. |
Job type	Valid operators are = (equal to), != (not equal to), `is null`, and `is not null`. The default operator is =.	
Description		
Maximum results	Not applicable	The search string specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration. The default is `50`.

Example: If you have already defined groups Region1, Region2, Center1, and Center2, you can specify the = operator and the `Region*` search string for the group name parameter. When you click **Find**, only groups Region1 and Region2 are displayed in the collection.

Group name: Specifies the name of the group.

Members: Specifies the number of members in the group. A member is a target.

Description: Specifies a description of the group.

Target group settings:

Use this page to view and set the description and members of a group of targets. Target groups make job submission easier because you can submit a job for a group of targets instead of a separate job for each target.

To view this administrative console page, click **Jobs** > **Target groups** > *group_name*.

Group name: Specifies the name of the group. When you create a new target group, the name is verified to be unique.

Data type	String
Maximum length	64 characters
Default	None

Description: Specifies an optional string that describes the group.

Data type	String
Maximum length	256 characters
Default	None

Member list: Specifies targets to add to the group. You can either type the target name or use the **Find** option to add targets to the list. The target must already exist. There are three types of targets:
* Deployment manager
* Unfederated application servers that have been registered with an administrative agent
* Host computer

The target must be registered with the job manager.

Find targets:

Use this page to build a list of targets that you can use to choose the targets on which you want the job to run. You can also find targets to add to a group of targets.

To view this administrative console page, do either of the following:
* On step 2 of the Job submission wizard, select **Find**.
* Click **Jobs** > **Target groups** > *group_name* > **Find**.

Click **Find** to list targets that are registered with the job manager.

To specify targets on which you want the job to run, select one or more targets in the Excluded targets list and click **>**. The selected targets are moved to the Chosen targets list.

To move targets from the Chosen targets list, select the targets and click **<**. The selected targets are moved to the Excluded targets list.

Find:

Specifies parameters to limit the search for targets. By default, all targets are searched. After you click **Find**, the Find results are displayed in Excluded targets. Click **Reset** to assign the parameters the default values.

The following table lists Find parameters. The target type, target name, job type, and unique identifier parameters define targets and are always available. The advanced find options are for the target properties, dynamic and built at run time. The dynamic parameters are displayed when you click **Advanced find options**. The maximum results parameter specifies the number of records to display.

Table 27. Find parameters.   Specify Find parameters to limit the search for targets.

Parameter name	Operators	Search strings
Target type	Valid operators include `All`, `Host`, and `Node`. By default, all target types are searched.	Not applicable
Target name	Valid operators are = (equal to) and != (not equal to). The default operator is =.	The search string specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, to find all jobs with a target name that starts with `Node`, set the target name parameter to `Node*`. To search for an exact match for multiple items, include comma-separated items. For example, to search on two target names, specify `Node1`, `Node2`. When you search for more than one item, you cannot use the asterisk. **Example:** If the targets are AppSvr01, AppSvr02, AppSvr03, and Test01, you can specify the = operator and the `App*` search string for the **Target name** parameter, then click **Find**. The target names displayed are AppSvr01, AppSvr02, and AppSvr03.
Job type	Valid operators are = (equal to), != (not equal to), `is null`, and `is not null`. The default operator is =.	
Unique identifier		
Advanced find options These options are dynamic parameters for target properties that are built at run time. The list of options depends upon your runtime environment.		

Table 27. Find parameters (continued).   Specify Find parameters to limit the search for targets.

Parameter name	Operators	Search strings
Resources Enables you to further narrow the list of targets to find targets that do or do not have specific resources. For example, to find targets that have a server named `server1`.	Specify query conditions to find resources. 1. Specify **With** (button shows **With** | Without), or click the button to toggle to **Without** (button shows With | **Without**). The button specifies whether to search for resources that meet the query conditions (**With**), or do not meet the query conditions (**Without**). 2. Select a resource type, such as `application`, `server`, `cluster`, `Installation Manager`, `Package`, or `Profile`. The list of resource types depends upon whether the registered targets are stand-alone application servers, deployment managers, or hosts. 3. Select a property for the specified resource type. 4. Select an operator. Valid operators are = (equal to), != (not equal to), < (less than), > (greater than), <= (less than or equal to), >= (greater than or equal to), `is null`, and `is not null`. The default operator is =. 5. Specify a value for the resource property; for example, an asterisk (*). To add more conditions to the first query, click _, the Add icon. If you click _, you cannot change the **With | Without** toggle button. Click _, the Delete icon, to remove an added condition. To narrow the search and add a query that changes the **With | Without** toggle button, click ⊞ , the Add Search Clause icon below the **With | Without** toggle button. In the dialog that opens, select **Perform subset search** to perform a subset search that refines the results of the first query and click **OK**. If you want the query to simply use a different **With | Without** selection and not perform a subset search, then do not select **Perform subset search**. Then, specify another query condition.	The value that you specify for the resource property is like a search string. You can use an asterisk (*) to designate a partial string. You can also specify comma-separated items to search on multiple values. When you search for more than one item, do not use the asterisk. **Example:** To find targets that have stopped applications, specify the following query conditions: 1. **With** 2. `application` resource type 3. `status` resource property 4. = (equal to) operator 5. `STOPPED` property value Then click **Find**. The table lists targets that have stopped applications. To narrow the search to targets that have stopped applications and that do not have a server named `server1`, specify the previous query. Then, click ⊞ , the Add Search Clause icon below the **With** | Without button. In the dialog that opens, select **Perform subset search** and click **OK**. Another query row is displayed, with parentheses around the **With** | Without button. The parentheses indicate the query is a subset search; only the targets resulting from the previous query are searched. In the new query row, specify the following query conditions: 1. **Without** 2. `Server` resource type 3. `serverName` resource property 4. = (equal to) operator 5. `server1` property value Click **Find**.

Table 27. Find parameters (continued). Specify Find parameters to limit the search for targets.

Parameter name	Operators	Search strings
Maximum results	Not applicable	The search string specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration. The default value is 50.

Excluded targets: Specifies the initial results of the Find option. The targets are not included in the list you are building until you click **>** to move targets into the chosen targets list. You can select multiple items in the list, and move them simultaneously.

Chosen targets: Specifies the list of targets to be included in the group of targets that you are creating. Click **<** to move targets into the excluded targets list. You can select multiple items in the list, and move them simultaneously.

Tuning the job polling interval

You can tune the polling interval that each deployment manager or administrative agent uses to poll the job manager for jobs. The default polling interval is 30 seconds. With a larger polling interval, the rate at which the deployment managers or administrative agent contact the job manager decreases, which alleviates high processor usage. With a lower polling interval, jobs start running sooner.

Before you begin

To configure the polling interval of a deployment manager, the deployment manager must be running.

To configure the polling interval of an administrative agent, the administrative agent must be running.

About this task

The job manager processor usage increases as more nodes or deployment managers are registered with the job manager. This increase occurs even if a job is not submitted to the job manager. The job manager eventually runs at 100 percent processor usage as more targets are registered. The threshold for the number of targets that you can register with the job manager without high processor usage depends on the hardware capability of the machine that runs it.

This topic describes how to change the polling interval for the deployment manager or administrative agent using the administrative console.

Procedure

1. Determine the polling interval to set to lower processor usage or to improve job processing.

   The default polling interval is 30 seconds. As the number of nodes or deployment managers that are registered with the job manager increases, the polling interval must increase to avoid high processor usage on the job manager. The actual value depends on the hardware that the job manager is running. However, you must determine the value on a case-by-case basis. With up to 100 nodes or deployment managers registered with the job manager, you might need a polling interval that is as long as 20 minutes.

2. Change the polling interval for the deployment manager as needed.

   a. Click **System administration** > **Deployment manager** > **Job managers** > **UUID**.

   b. Change the value for the polling interval setting.

   c. Save the changes and restart the deployment manager.

3. Change the polling interval, as needed, for each node that an administrative agent manages.

a. Click **System administration** > **Administrative agent** > **Nodes** > *node_name* > **Job managers** > **Job manager UUID**.

b. Change the value for the polling interval setting.

c. After changing the setting for all the nodes that an administrative agent manages, save the changes and restart the administrative agent.

Results

You configured the polling interval.

What to do next

Monitor the processor usage on the job manager. If the processor usage is still high, increase the polling interval. If jobs take too long to start, lower the polling interval.

Configuring administration services

You can configure administration services such as remote file services, repository services, and Java Management Extensions (JMX) connectors.

Remote files services for file transfer and file synchronization

Configuration documents describe the available application servers, their configurations, and their contents. Two file services manage configuration documents: the file transfer service and the file synchronization service.

The following information describes what the file services do:

File transfer service

The file transfer service enables the moving of files between the deployment manager and the nodes as well as between the wsadmin scripting process and either the deployment manager or the application server. It uses the HTTP protocol to transfer files. When you enable security in the WebSphere Application Server product, the file transfer service uses certificate-based mutual authentication. You can use the default key files in a test environment. Ensure that you change the default key file to secure your system.

The ports used for file transfer are the HTTP_Transport port, the HTTPS transport port, the administrative console port, and the administrative console secure port. For more information, see the Planning for TCP/IP port convention topic in the *Installing your application serving environment* PDF.

File synchronization service

The file synchronization service ensures that a file set on each node matches that on the deployment manager node. This service promotes consistent configuration data across a cell. You can adjust several configuration settings to control file synchronization on individual nodes and throughout a system.

This service runs in the deployment manager and node agents, and ensures that configuration changes made to the cell repository are propagated to the appropriate node repositories. The cell repository is the master repository, and configuration changes made to node repositories are not propagated up to the cell. During a synchronization operation a node agent checks with the deployment manager to see if any configuration documents that apply to the node have been updated. New or updated documents are copied to the node repository, and deleted documents are removed from the node repository.

The default behavior, which is enabled, is for each node agent to periodically run a synchronization operation. You can configure the interval between operations or disable the periodic behavior. You can also configure the synchronization service to synchronize a node repository before starting a server on the node.

Repository service settings

Use this page to view and change the configuration for an administrative service repository.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Repository service**.

Audit Enabled

Specifies whether to audit repository updates in the log file. The default is to audit repository updates.

Data type	Boolean
Default	true

Remote files services for file transfer and file synchronization

Configuration documents describe the available application servers, their configurations, and their contents. Two file services manage configuration documents: the file transfer service and the file synchronization service.

The following information describes what the file services do:

File transfer service

The file transfer service enables the moving of files between the deployment manager and the nodes as well as between the wsadmin scripting process and either the deployment manager or the application server. It uses the HTTP protocol to transfer files. When you enable security in the WebSphere Application Server product, the file transfer service uses certificate-based mutual authentication. You can use the default key files in a test environment. Ensure that you change the default key file to secure your system.

The ports used for file transfer are the HTTP_Transport port, the HTTPS transport port, the administrative console port, and the administrative console secure port. For more information, see the Planning for TCP/IP port convention topic in the *Installing your application serving environment* PDF.

File synchronization service

The file synchronization service ensures that a file set on each node matches that on the deployment manager node. This service promotes consistent configuration data across a cell. You can adjust several configuration settings to control file synchronization on individual nodes and throughout a system.

This service runs in the deployment manager and node agents, and ensures that configuration changes made to the cell repository are propagated to the appropriate node repositories. The cell repository is the master repository, and configuration changes made to node repositories are not propagated up to the cell. During a synchronization operation a node agent checks with the deployment manager to see if any configuration documents that apply to the node have been updated. New or updated documents are copied to the node repository, and deleted documents are removed from the node repository.

The default behavior, which is enabled, is for each node agent to periodically run a synchronization operation. You can configure the interval between operations or disable the periodic behavior. You can also configure the synchronization service to synchronize a node repository before starting a server on the node.

Configuring remote file services

Configuration data for the WebSphere Application Server product resides in files. Two services help you reconfigure and otherwise manage these files: the file transfer service and file synchronization service.

About this task

By default, the file transfer service is always configured and enabled at a node agent, so you do not need to take additional steps to configure this service. However, you might need to configure the file synchronization service.

Procedure

1. Go to the File Synchronization Service page. Click **System Administration** > **Node agents** in the console navigation tree. Then, click the node agent for which you want to configure a synchronization server and click **File synchronization service**.

2. On the File Synchronization Service page, customize the service that helps make configuration data consistent across a cell by moving updated configuration files from the deployment manager to the node. Change the values for properties on the File Synchronization Service page. The file synchronization service is always started, but you can control how it runs by changing the values.

3. By default if node synchronization fails five times consecutively, automatic synchronization is disabled. To keep automatic synchronization enabled, specify a custom property on the Java virtual machine (JVM) of node agent using the administrative console.

   a. Click **System Administration** > **Node Agents** > *nodeagent*. Click on the node agent for which you want to configure the optional custom property.

   b. Under Server Infrastructure, expand **Java and Process Management** then click **Process Definition** > **Java Virtual Machine** > **Custom Properties**.

   c. Click **New**.

   d. Specify a name and value for the custom property.

      The com.ibm.websphere.management.sync.allowfailure custom property:

      To keep automatic synchronization enabled, specify this custom property with a value of `true`.

Property	com.ibm.websphere.management.sync.allowfailure
Data type	Boolean
Default	False

4. Optionally add a custom property for file synchronization on the file synchronization service page.

   a. Click **System Administration** > **Node Agents** > *nodeagent*. Click on the node agent for which you want to configure the optional custom property.

   b. Under Additional Properties, click **File synchronization service**

   c. Under Additional Properties, click **Custom Properties**

   d. Click **New**.

   e. Specify a name and value for the custom property.

      The com.ibm.websphere.management.application.expand.wto custom property:

      Specify this custom property with a value of `true` if you want to display Enterprise Archive (EAR) file expansion errors on thez/OS operating system console. When the errors display, the operator can take appropriate corrective action for the failure.

Property	com.ibm.websphere.management.application.expand.wto
Data type	Boolean
Default	False

File transfer service settings

Use this page to configure the service that transfers files from the deployment manager to individual remote nodes.

To view this administrative console page, click **System Administration > Node agents >** *node_agent_name* **> File transfer service**.

Enable service at server startup:

Specifies whether the server attempts to start the specified service. Some services are always enabled and disregard this property if set. This setting is enabled by default.

Data type	Boolean
Default	true

Retries count:

Specifies the number of times you want the file transfer service to retry sending or receiving a file after a communication failure occurs.

Data type	Integer
Default	3
	If the retries count setting is blank, the file transfer service sets the default to 3. If the retries count setting is 0, the file transfer service does not retry. The default is the recommended value.

Retry wait time:

Specifies the number of seconds that the file transfer service waits before it retries a failed file transfer.

Data type	Integer
Default	10
	If the retry wait time setting is blank, the code sets the default to 10. If the retry wait time setting is 0, the file transfer service does not wait between retries. The default is the recommended value.

File synchronization service settings

Use this page to specify that a file set on one node matches that on the central deployment manager node and to ensure consistent configuration data across a cell.

You can synchronize files on individual nodes or throughout your system.

Note: If your installation includes mixed release cells, a large numbers of nodes, and runs a large number of applications, you might want to use the **Generic JVM Arguments** field, on the Java Virtual Machine Settings page of the administrative console, to enable the hot restart sync feature of the synchronization service. This feature indicates to the synchronization service that the installation is running in an environment where configuration updates are not made when the deployment manager is not active. Therefore, the service does not have to perform a complete repository comparison when the deployment manager or node agent servers restart. For more information, see the Generic JVM arguments in the Java virtual machine settings documentation.

Important: Do not routinely disable the synchronization process as various portions of the application server run time depend on synchronization. For example, the security run time depends on node synchronization to propagate updated certificates during automated replacement processes. Also, the security runtime also depends on it for Lightweight Third Party Authentication (LTPA) key changes. Other portions of the run time are dependent on synchronization. However, a complete list is not available. If you *permanently* disable synchronization, nodes might not be synchronized and an outage will result. The only exception is the configuration save operation. To avoid synchronizing the configuration when

the configuration repository is being updated by a save operation, it might be beneficial to *temporarily* disable the synchronization process, save the configuration, and then re-enable the synchronization process. This process ensures that changes are fully committed to the configuration repository before the node synchronization.

To view this administrative console page, click **System administration** > **Node agents** > *node_agent_name* > **File synchronization service**.

Enable service at server startup:

Specifies whether the server attempts to start the file synchronization service. This setting does not cause a file synchronization operation to start. This setting is enabled by default.

Important: Disabling synchronization is not recommended. Various portions of the run time have dependencies on synchronization being enabled. For example, in a multiple-server product, the security environment depends on node synchronization to propagate updated certificates during automated replacement and for LTPA key changes. If synchronization is disabled, the nodes might get out of synchronization, resulting in an outage.

Data type	Boolean
Default	true

Synchronization interval:

Specifies the number of minutes that elapse between synchronizations. Increase the time interval to synchronize files less often. Decrease the time interval to synchronize files more often.

Data type	Integer
Units	Minutes
Default	10
	The minimum value that the application server uses is 1. If you specify a value of 0, the application server ignores the value and uses the default of 1.
	Important: You can change the node synchronization interval to a greater number of minutes. However, before you make the change, carefully consider the impact because some functions critically depend on node synchronization. Before changing the value to any value other than the default value, it is advisable that you complete careful, large-scale testing in your environment.

Automatic synchronization:

Specifies whether to synchronize files automatically after a designated interval. When this setting is enabled, the node agent automatically contacts the deployment manager every synchronization interval to attempt to synchronize the node's configuration repository with the master repository owned by the deployment manager.

If the Automatic synchronization setting is enabled, the node agent attempts file synchronization when it establishes contact with the deployment manager. The node agent waits the synchronization interval before it attempts the next synchronization.

Remove the check mark from the check box if you want to control when files are sent to the node.

Data type	Boolean
Default	true

Startup synchronization:

Specifies whether the node agent attempts to synchronize the node configuration with the latest configurations in the master repository prior to starting an application server.

The default is to not synchronize files prior to starting an application server. Enabling the setting ensures that the node agent has the latest configuration but increases the amount of time it takes to start the application server.

Note that this setting has no effect on the startServer command. The startServer command launches a server directly and does not use the node agent.

Data type   Boolean
Default   false

Exclusions:

Specifies files or patterns that should not be part of the synchronization of configuration data. Files in this list are not copied from the master configuration repository to the node, and are not deleted from the repository at the node.

The default is to have no files specified.

To specify a file, use a complete name or a name with a leading or trailing asterisk (*) for a wildcard. For example:

`cells/`*`cell name`*`/nodes/`*`node name`*`/file name`	Excludes this specific file
`*/`*`file name`*	Excludes files named *file name* in any context
`dirname/*`	Excludes the subtree under `dirname`

Press **Enter** at the end of each entry. Each file name appears on a separate line.

Since these strings represent logical document locations and not actual file paths, only forward slashes are needed no matter the platform.

Changes to the exclusion list are picked up when the node agent is restarted.

Data type   String
Units   File names or patterns

Repository service settings

Use this page to view and change the configuration for an administrative service repository.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Repository service**.

Audit Enabled

Specifies whether to audit repository updates in the log file. The default is to audit repository updates.

Data type   Boolean
Default   true

Java Management Extensions connector properties

You can specify or set a property in the administrative console, the wsadmin tool, Application Server commands, the scripts that run from a command-line interface, or a custom Java administrative client program that you write. You can also set SOAP connector properties in the soap.client.props file and IPC connector properties in the ipc.client.props file.

A Java Management Extensions (JMX) connector can be a Remote Method Invocation (RMI) connector, a Simple Object Access Protocol (SOAP) connector, a JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI) connector, or an Inter-Process Communications (IPC) connector.

Note: You should eventually convert all of your RMI connectors to JSR160RMI connectors because support for the RMI connector is deprecated.

For specific information on how to code the JMX connector properties for the wsadmin tool, the Application Server commands, or scripts, see the particular tool or command. Read the application programming interfaces documentation to learn how to code the JMX connector properties for a custom Java administrative client program.

The JMX connectors that servers create use JMX connector properties that are accessible in the administrative console. The wsadmin tool and the Java administrative client use JMX connector properties in the soap.client.props, ipc.client.props, and sas.client.prop files.

For the administrative console, this topic specifies the coding of the particular setting or property. Coding of properties in the `soap.client.props` file and the `ipc.client.props` file that are specific to JMX connectors is specified. These SOAP properties begin with com.ibm.SOAP and the IPC properties begin with com.ibm.IPC. Other properties in the `soap.client.props` file and the `ipc.client.props` file that contain information that can be set elsewhere in the application server are not documented here. The coding for the com.ibm.ssl.contextProvider property, which can be set only in the `soap.client.props` file and the `ipc.client.props` file, is specified.

Each profile has property files at the following locations:
- For the SOAP connector:
  - *profile_root*/properties/soap.client.props
- For the IPC connector:
  - *profile_root*/properties/ipc.client.props

These property files allow you to set different properties, including security and timeout properties. These properties are the default for all the administrative connections that use either the SOAP JMX connector or the IPC JMX connector between processes that run in a particular profile. For instance, the wsadmin program running under a particular profile uses the property values from these files for the SOAP connector behavior and the IPC connector behavior unless the properties are overridden by some other programmatic means.

To view the JMX connector custom properties administrative console page that goes with this topic, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **JMX connectors** > *connector_type* > **Custom properties**.

SOAP connector properties

This section discusses the following JMX connector properties that pertain to SOAP connectors:
- Configuration URL
- Secure Sockets Layer (SSL) security
- Security context provider

- SOAP request timeout
- SSL alias

Configuration URL

Specify the configuration Universal Resource Locator (URL) property if you want a program to read SOAP properties from this file. You can set the property by using one of the following options:

- Scripts run from a command-line interface. Scripts can pass the Configuration URL property to the application server on the com.ibm.SOAP.ConfigURL system property.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	ConfigURL
Data type	String
Valid Value	http://*Path*/soap.client.props
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_SOAP_CONFIG property.

Secure Sockets Layer (SSL) security

Use this property to enable SSL security between the application server and the SOAP client. Set the property by using one of the following options:

- Scripts that run from a command-line interface.
- The `soap.client.props` file.

Property	com.ibm.SOAP.securityEnabled
Data type	Boolean
Default	False

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	securityEnabled
Data type	Boolean
Default	False

- A Java administrative client. Use the AdminClient.CONNECTOR_SECURITY_ENABLED property.

Security context provider

This property indicates the Secure Sockets Layer (SSL) implementation to use between the application server and the SOAP client.

Set the property by using the `soap.client.props` file.

Property	com.ibm.ssl.contextProvider
Data type	String
Valid Values	IBMJSSE2
Default	IBMJSSE2

SOAP request timeout

The value that you choose depends on a number of factors, such as the size and the number of the applications that are installed on the server, the speed of your machine, and the usage of your machine.

The program default value for the request timeout is `600 seconds`. However, other components that connect to the SOAP client can override the default. Components that use the `soap.client.props` file have a default value of `180 seconds`.

Set the property by using one of the following options:
- Scripts that run from a command-line interface.
- The `soap.client.props` file.

Property	com.ibm.SOAP.requestTimeout
Data type	Integer
Range in seconds	0 to n
	If the property is zero (0), the request never times out.
Default	180

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	requestTimeout
Data type	Integer
Range in seconds	0 to n
	If the property is zero (0), the request never times out.
Default	600

- A Java administrative client. The property is AdminClient.CONNECTOR_SOAP_REQUEST_TIMEOUT.

SSL alias

This property specifies the alias to use for an SSL configuration for client connections. The value of the alias is what you want it to be.

Set the property in the `soap.client.props` file.

Property	com.ibm.ssl.alias
Data type	String
Default	DefaultSSLSettings

IPC connector properties

This section discusses the following JMX connector properties that pertain to IPC connectors:
- Configuration URL
- IPC request timeout
- Secure Sockets Layer (SSL) security
- Security context provider
- SSL alias

Configuration URL

Specify the configuration URL property if you want a program to read IPC properties from this file. You can set the property by using one of the following options:
- Scripts run from a command-line interface. Scripts can pass the Configuration URL property to the Application Server on the com.ibm.IPC.ConfigURL system property.

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	ConfigURL
Data type	String
Valid Value	http://*Path*/ipc.client.props
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_IPC_CONFIG property.

IPC request timeout

The value that you choose depends on a number of factors, such as the size and the number of the applications that are installed on the server, the speed of your machine, and the usage of your machine.

The program default value for the request timeout is `600 seconds`. However, other components that connect to the IPC client can override the default. Components that use the `ipc.client.props` file have a default value of `180 seconds`.

Set the property by using one of the following options:
- Scripts that run from a command-line interface.
- The `ipc.client.props` file.

Property	com.ibm.IPC.requestTimeout
Data type	Integer
Range in seconds	0 to n
	If the property is zero (0), the request never times out.
Default	180

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	requestTimeout
Data type	Integer
Range in seconds	0 to n
	If the property is zero (0), the request never times out.
Default	600

- A Java administrative client. The property is AdminClient.CONNECTOR_IPC_REQUEST_TIMEOUT.

Secure Sockets Layer (SSL) security

Use this property to enable SSL security between Application Server and the IPC client. Set the property by using one of the following options:
- Scripts that run from a command-line interface.
- The `ipc.client.props` file.

Property	com.ibm.IPC.securityEnabled
Data type	Boolean
Default	False

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	securityEnabled
Data type	Boolean
Default	False

- A Java administrative client. Use the AdminClient.CONNECTOR_SECURITY_ENABLED property.

Security context provider

This property indicates the SSL implementation to use between the application server and the IPC client.

Set the property by using the `ipc.client.props` file.

Property	com.ibm.ssl.contextProvider
Data type	String
Valid Values	`IBMJSSE2`
Default	IBMJSSE2

SSL alias

This property specifies the alias to use for an SSL configuration for client connections. The value of the alias is what you want it to be.

Set the property in the `ipc.client.props` file.

Property	com.ibm.ssl.alias
Data type	String
Default	DefaultSSLSettings

SOAP, RMI, JSR160RMI, and IPC connector properties

This section discusses JMX connector properties that pertain to the following SOAP connectors, RMI connectors, JSR160RMI connectors, and IPC connectors:
- Connector type
- Disabling a connector
- Host
- Password
- Port
- User name

Connector type

A connector type of SOAP, RMI, JSR160RMI, or IPC depends on whether the application server connects to a SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:
- The wsadmin tool.
- Scripts that run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	Type
Data type	String

Valid values	SOAPConnector		
	RMIConnector		
	JSR160RMIConnector		
	IPCConnector		
Default	SOAPConnector	JSR160RMI	IPC

- A Java administrative client. Use the AdminClient.CONNECTOR_TYPE property. Specify the connector type by using the AdminClient.CONNECTOR_TYPE_RMI, the AdminClient.CONNECTOR_TYPE_SOAP, the AdminClient.CONNECTOR_TYPE_JSR160RMI, or the AdminClient.CONNECTOR_TYPE_IPC constants.

Disabling a connector

You can enable or disable any of the JMX connectors from the administrative console.
- The wsadmin tool.
- The administrative console. Select the box next to the connector to enable the connector. Clear the box next to the connector to disable the connector.

Property	enabled
Data type	Boolean
Value	truelfalse

Host

The host name or the IP address of the server to which the application server connects. The server can be a SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:
- The wsadmin tool.
- Scripts that run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	host
Data type	String
Valid values	Host name or IP address
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_HOST property.

Password

The password that the application server uses to access the SOAP server, the RMI server, the JSR160RMI server, or the IPC server. You can set the property by using one of the following options:
- The wsadmin tool.
- Scripts run from a command-line interface.
- The `soap.client.props` file for the SOAP server, the RMI server, or the JSR160RMI server.

Property	com.ibm.SOAP.loginPassword
Data type	String
Valid values	The value must match the global SSL settings for SOAP, RMI, or JSR160RMI.
Default	None

- The `ipc.client.props` file for the IPC server.

Property	com.ibm.IPC.loginPassword
Data type	String
Valid values	The value must match the global SSL settings for IPC.
Default	None

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	password
Data type	String
Valid values	The value must match the global SSL settings for SOAP, RMI, JSR160RMI, or IPC.
Default	None

- A Java administrative client. Use the AdminClient.PASSWORD property.

Port

The port number of the server to which the application server connects. The server can be a SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	port
Data type	Integer
Valid value	Port number
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_PORT property.

User name

The user name that the application server uses to access the SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts run from a command-line interface.
- The `soap.client.props` file for the SOAP server, an RMI server, a JSR160RMI server.

Property	com.ibm.SOAP.loginUserid
Data type	String
Valid value	The value must match the global SSL settings for SOAP, RMI, or JSR160RMI.
Default	None

- The `ipc.client.props` file for the IPC server.

Property	com.ibm.IPC.loginUserid
Data type	String
Valid value	The value must match the global SSL settings for IPC.
Default	None

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	username
Data type	String
Valid value	The value must match the global SSL settings for SOAP, RMI, JSR160RMI, or IPC.
Default	None

- A Java administrative client. Use the AdminClient.USERNAME property.

RMI connector properties

This section discusses the following JMX connector properties that pertain to RMI connectors:
- Disabling the JSR 160 RMI connector

Disabling the JSR 160 RMI connector

Support for JMX Remote application programming interface (JSR 160) is enabled by default so that you automatically receive specification-compliant JMX function. To disable the function for a particular server, set the property by using one of the following options:
- The wsadmin tool.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	disableJDKJMXConnector
Data type	string
Value	true

Java Management Extensions (JMX) connectors

Use this page to view and change the configuration for Java Management Extensions (JMX) connectors, which make connections between server processes. The types of JMX connectors are Simple Object Access Protocol (SOAP), Remote Method Invocation (RMI), JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI), and Inter-Process Communications (IPC).

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **JMX Connectors**.

Java Management Extensions (JMX) connectors communicate with WebSphere Application Server when you invoke a scripting process. There is no default for the type and parameters of a connector. The `wsadmin.properties` file specifies the SOAP connector and an appropriate port number. You can also use the RMI connector, the JSR160RMI connector, or the IPC connector.

Use one of the following methods to select the connector type and attributes:
- Specify properties in a properties file.
- Indicate options on the command line.

On the z/OS platform, the SOAP connector, the RMI connector, the JSR160RMI connector, and the IPC connector connect to a controller (server) . WebSphere Application Server for z/OS internally routes MBean requests from a controller to its servant regions as appropriate.

Type

Specifies the type of the JMX connector.

Data type	Enumeration
Default	SOAPConnector
Range	**SOAPConnector**
	For JMX connections using Simple Object Access Protocol (SOAP).
	RMIConnector
	For JMX connections using Remote Method Invocation (RMI).
	JSR160RMIConnector
	For JMX connections using JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI).
	IPCConnector
	For JMX connections using Inter-Process Communications (IPC).

Enabled

Specifies whether a JMX connector is enabled. If `Yes` is specified, the connector is enabled. All JMX connectors are enabled by default.

To disable a JMX connector, select the connector and click **Disable**. The **Enabled** value changes to `No`. To enable a JMX connector, select the connector and click **Enable**. The Enabled value changes to `Yes`.

Data type	Boolean

JMX connector settings

Use this page to view the configuration for a Java Management Extensions (JMX) connector, which makes connections between server processes.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **JMX Connectors** > *connector_type*.

The Simple Object Access Protocol (SOAP) connector, the Remote Method Invocation (RMI) connector, the JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI) connector, or the Inter-Process Communications (IPC) connector connect to a controller (server). WebSphere Application Server for z/OS internally routes MBean requests from a controller to its servant regions as appropriate.

Type:

Specifies the type of the JMX connector.

Data type	Enumeration
Default	SOAPConnector

Range

SOAPConnector
For JMX connections using Simple Object Access Protocol (SOAP).

RMIConnector
For JMX connections using Remote Method Invocation (RMI).

JSR160RMIConnector
For JMX connections using JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI).

IPCConnector
For JMX connections using Inter-Process Communications (IPC).
gotcha: IPC_CONNECTOR_ADDRESS is a special end point. The end point must be defined as `'localhost'`. Editing a hostname definition from `'localhost'` to any other <**hostname**> is not permitted if you plan to use this same server as a template to create another server in another node and which is to have an end point definition different than <**hostname**>.

SOAP connector and Inter-Process Communications connector properties files

Use the `soap.client.props` file to set properties for the SOAP connector and the `ipc.client.props` file to set properties for the Inter-Process Communications (IPC) connector. Most of the properties in the `ipc.client.props` file have corresponding properties in the `soap.client.props` file.

The SOAP connector properties file for a particular profile is at the following location:

- *profile_root*/properties/soap.client.props

The IPC connector properties file for a particular profile is at the following location:

- *profile_root*/properties/ipc.client.props

The following table provides basic information on the various properties. Read the properties files to obtain more detailed information.

Table 28. SOAP connector and IPC connector property descriptions. The properties configure the SOAP and IPC connectors.

SOAP connector properties	IPC connector properties	Description
com.ibm.SOAP.securityEnabled	com.ibm.IPC.securityEnabled	Specifies enablement of security for the connector. Set the property to `true` to enable security.
com.ibm.SOAP.authenticationTarget	com.ibm.IPC.authenticationTarget	Specifies the type of authentication for the connector if security is enabled. You can specify `BasicAuth` for basic authentication. If no value is specified, basic authentication is used.
com.ibm.SOAP.loginUserid	com.ibm.IPC.loginUserid	Specifies the user ID for the connector if security is enabled, and you do not enter a user ID through a command prompt or standard in.
com.ibm.SOAP.loginPassword	com.ibm.IPC.loginPassword	Specifies the password for the connector if security is enabled, and you do not enter a password through a command prompt or standard in.
com.ibm.SOAP.loginSource	com.ibm.IPC.loginSource	Specifies automatic prompting for the user ID and password when you specify `prompt`. Prerequisites for using this property are discussed in the properties file for the particular connector.

Table 28. SOAP connector and IPC connector property descriptions (continued). The properties configure the SOAP and IPC connectors.

SOAP connector properties	IPC connector properties	Description
com.ibm.SOAP.requestTimeout	com.ibm.IPC.requestTimeout	Specifies how long in seconds the connector waits for a server response. The property for the SOAP connector and the property for the IPC connector are each initially set to 180 in their respective properties files.
com.ibm.ssl.alias	com.ibm.ssl.alias	This property specifies the alias to use for a Secure Sockets Layer (SSL) configuration for client connections. The value of the alias is what you want it to be.
	timeToExpiration	Specifies the time in seconds that connections can be idle in the connection pool. Beyond this time the connections are purged. The initial setting for the property is 360.

Note: Many of the system management commands contain implicit stop server operations. Most of these commands use the IPC connector properties configuration. However, some commands, such as the stopServer, stopNode, and stopManager commands, continue to use the SOAP connector properties configuration for compatibility reasons. Thus, to avoid additional user ID and password prompts, specify the user ID and password information in both the `soap.client.props` and `ipc.client.props` files.

Extension MBean Providers collection

Use this page to view and change the configuration for JMX extension MBean providers.

You can configure JMX extension MBean providers to be used to extend the existing WebSphere managed resources in the core administrative system. Each MBean provider is a library containing an implementation of a JMX MBean and its MBean XML Descriptor file.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Extension MBean Providers**.

Name

The name used to identify the Extension MBean provider library.

Description

An arbitrary descriptive text for the Extension MBean Provider configuration.

Classpath

The path to the Java archive (JAR) file that contains the Extension MBean provider library. This class path is automatically added to the Application Server class path.

Extension MBean Provider settings

Use this page to view and change the configuration for a JMX extension MBean provider.

You can configure a library containing an implementation of a JMX MBean, and its MBean XML Descriptor file, to be used to extend the existing WebSphere managed resources in the core administrative system

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Extension MBean Providers** > *provider_library_name*.

Name:

The name used to identify the Extension MBean provider library.

Data type   String

Classpath:

The path to the Java archive (JAR) file that contains the Extension MBean provider library. This class path is automatically added to the Application Server class path. The class loader needs this information to load and parse the Extension MBean XML Descriptor file.

Data type   String

Description:

An arbitrary descriptive text for the Extension MBean Provider configuration. Use this field for any text that helps identify or differentiate the provider configuration.

Data type   String

Extension MBean collection

You can configure Java Management Extension (JMX) MBeans to extend the existing WebSphere Application Server managed resources in the administrative console. Use this page to register JMX MBeans. Any MBeans that are listed have already been registered.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Extension MBean Providers** > *provider_library_name* > **extensionMBeans**.

descriptorURI

Specifies the location, relative to the provider class path, where the MBean XML descriptor file is located.

type

Specifies the type to use for registering this MBean. The type must match the type that is declared in the MBean descriptor file.

Extension MBean settings

Use this page to view and configure Java Management Extension (JMX) MBeans.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Extension MBean Providers** > *provider_library_name* > **extensionMBeans** > *descriptorURI*.

descriptorURI:

Specifies the location, relative to the provider class path, where the MBean XML descriptor file is located.

Data type   String

type:

Specifies the type to use for registering this MBean. The type must match the type that is declared in the MBean descriptor file.

Data type   String

Administrative audit messages in system logs

The product provides administrative audit messages in system logs that contain some audit information. The audit messages described in this topic are part of the standard product audit stream and do not provide administrative event auditing information such as who changed files.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

Important: The functionality described in this topic uses system logs and is not a part of the security auditing subsystem. The audit information captured by this functionality does not correspond with the audit information captured by the security auditing subsystem. For information about the security auditing subsystem, see the topic on auditing the security infrastructure.

Administrative audits use the same trace logging facility as the rest of the product, and do not use the logging facility that is a part of the security auditing subsystem. The audits are available in both the `activity.log` file and the `SystemOut.log` of the server that performs the action. You do not need to enable trace to produce the audits. However, through the Repository service console page, you can control whether configuration change auditing is done. This type of audit is done by default. Operational command auditing is always enabled. Information about which user performed the change is available only when security is enabled.

You can do administrative audits with or without the security audit facility.

The following administrative actions are audited:
- All configuration changes, in terms of the configuration documents that are created, modified, or deleted.
- Certain operational changes like starting and stopping nodes, clusters, servers, and applications. These managed bean (MBean) operations provide administrative auditing:

Table 29. Administrative auditing MBean operations. The MBean types provide administrative auditing MBean operations.

MBean type	MBean operations
CellSync	syncNode
Cluster	start, stop, stopImmediate, rippleStart
NodeAgent	launchProcess, stopNode, restart
Server	stop, stopImmediate
AppManagement	startApplication, stopApplication

Configuration change audits have ADMRxxxxI message IDs, where xxxx is the message number. Operational audits have ADMN10xxI message IDs, where 10xx is the message number.

Here are some audit examples from a WebSphere Application Server, Network Deployment environment.

The following audit example is from the deployment manager `SystemOut.log` file:

```
[7/23/03 17:04:49:089 CDT] 39c26dad FileRepositor A ADMR0015I: Document
cells/ellingtonNetwork/security.xml was modified by user u1.
   [7/23/03 17:04:49:269 CDT] 3ea0edb5 FileRepositor A ADMR0016I: Document
cells/ellingtonNetwork/nodes/ellington/app.policy was created by user u1.
```

```
...
     [7/23/03 17:13:54:081 CDT] 39a572a1 AdminHelper   A ADMN1008I: Attempt
made to start the SamplesGallery application. (User ID = u1)
     ...
```

The following audit example is from the node agent `SystemOut.log` file:

```
[7/23/03 17:38:43:461 CDT]  23d1326 AdminHelper   A ADMN1000I: Attempt
made to launch server1 on node ellington. (User ID = u1)
```

The following audit example is from the application server`SystemOut.log` file:

```
[7/23/03 17:39:59:360 CDT] 24865373 AdminHelper   A ADMN1020I: Attempt
made to stop the server1 server. (User ID = u1)
```

The message text is split for printing purposes.

Java Management Extensions connector properties

You can specify or set a property in the administrative console, the wsadmin tool, Application Server commands, the scripts that run from a command-line interface, or a custom Java administrative client program that you write. You can also set SOAP connector properties in the soap.client.props file and IPC connector properties in the ipc.client.props file.

A Java Management Extensions (JMX) connector can be a Remote Method Invocation (RMI) connector, a Simple Object Access Protocol (SOAP) connector, a JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI) connector, or an Inter-Process Communications (IPC) connector.

Note: You should eventually convert all of your RMI connectors to JSR160RMI connectors because support for the RMI connector is deprecated.

For specific information on how to code the JMX connector properties for the wsadmin tool, the Application Server commands, or scripts, see the particular tool or command. Read the application programming interfaces documentation to learn how to code the JMX connector properties for a custom Java administrative client program.

The JMX connectors that servers create use JMX connector properties that are accessible in the administrative console. The wsadmin tool and the Java administrative client use JMX connector properties in the soap.client.props, ipc.client.props, and sas.client.prop files.

For the administrative console, this topic specifies the coding of the particular setting or property. Coding of properties in the `soap.client.props` file and the `ipc.client.props` file that are specific to JMX connectors is specified. These SOAP properties begin with com.ibm.SOAP and the IPC properties begin with com.ibm.IPC. Other properties in the `soap.client.props` file and the `ipc.client.props` file that contain information that can be set elsewhere in the application server are not documented here. The coding for the com.ibm.ssl.contextProvider property, which can be set only in the `soap.client.props` file and the `ipc.client.props` file, is specified.

Each profile has property files at the following locations:
*   For the SOAP connector:
    *   *profile_root*/properties/soap.client.props
*   For the IPC connector:
    *   *profile_root*/properties/ipc.client.props

These property files allow you to set different properties, including security and timeout properties. These properties are the default for all the administrative connections that use either the SOAP JMX connector or the IPC JMX connector between processes that run in a particular profile. For instance, the wsadmin

program running under a particular profile uses the property values from these files for the SOAP connector behavior and the IPC connector behavior unless the properties are overridden by some other programmatic means.

To view the JMX connector custom properties administrative console page that goes with this topic, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **JMX connectors** > *connector_type* > **Custom properties**.

SOAP connector properties

This section discusses the following JMX connector properties that pertain to SOAP connectors:
- Configuration URL
- Secure Sockets Layer (SSL) security
- Security context provider
- SOAP request timeout
- SSL alias

Configuration URL

Specify the configuration Universal Resource Locator (URL) property if you want a program to read SOAP properties from this file. You can set the property by using one of the following options:
- Scripts run from a command-line interface. Scripts can pass the Configuration URL property to the application server on the com.ibm.SOAP.ConfigURL system property.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	ConfigURL
Data type	String
Valid Value	http://*Path*/soap.client.props
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_SOAP_CONFIG property.

Secure Sockets Layer (SSL) security

Use this property to enable SSL security between the application server and the SOAP client. Set the property by using one of the following options:
- Scripts that run from a command-line interface.
- The `soap.client.props` file.

Property	com.ibm.SOAP.securityEnabled
Data type	Boolean
Default	False

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	securityEnabled
Data type	Boolean
Default	False

- A Java administrative client. Use the AdminClient.CONNECTOR_SECURITY_ENABLED property.

Security context provider

This property indicates the Secure Sockets Layer (SSL) implementation to use between the application server and the SOAP client.

Set the property by using the `soap.client.props` file.

Property	com.ibm.ssl.contextProvider
Data type	String
Valid Values	`IBMJSSE2`
Default	IBMJSSE2

SOAP request timeout

The value that you choose depends on a number of factors, such as the size and the number of the applications that are installed on the server, the speed of your machine, and the usage of your machine.

The program default value for the request timeout is `600 seconds`. However, other components that connect to the SOAP client can override the default. Components that use the `soap.client.props` file have a default value of `180 seconds`.

Set the property by using one of the following options:
- Scripts that run from a command-line interface.
- The `soap.client.props` file.

Property	com.ibm.SOAP.requestTimeout
Data type	Integer
Range in seconds	0 to n
	If the property is zero (0), the request never times out.
Default	180

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	requestTimeout
Data type	Integer
Range in seconds	0 to n
	If the property is zero (0), the request never times out.
Default	600

- A Java administrative client. The property is AdminClient.CONNECTOR_SOAP_REQUEST_TIMEOUT.

SSL alias

This property specifies the alias to use for an SSL configuration for client connections. The value of the alias is what you want it to be.

Set the property in the `soap.client.props` file.

Property	com.ibm.ssl.alias
Data type	String
Default	DefaultSSLSettings

IPC connector properties

This section discusses the following JMX connector properties that pertain to IPC connectors:

- Configuration URL
- IPC request timeout
- Secure Sockets Layer (SSL) security
- Security context provider
- SSL alias

Configuration URL

Specify the configuration URL property if you want a program to read IPC properties from this file. You can set the property by using one of the following options:

- Scripts run from a command-line interface. Scripts can pass the Configuration URL property to the Application Server on the com.ibm.IPC.ConfigURL system property.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	ConfigURL
Data type	String
Valid Value	http://*Path*/ipc.client.props
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_IPC_CONFIG property.

IPC request timeout

The value that you choose depends on a number of factors, such as the size and the number of the applications that are installed on the server, the speed of your machine, and the usage of your machine.

The program default value for the request timeout is `600 seconds`. However, other components that connect to the IPC client can override the default. Components that use the `ipc.client.props` file have a default value of `180 seconds`.

Set the property by using one of the following options:

- Scripts that run from a command-line interface.
- The `ipc.client.props` file.

Property	com.ibm.IPC.requestTimeout
Data type	Integer
Range in seconds	0 to n
	If the property is zero (0), the request never times out.
Default	180

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	requestTimeout
Data type	Integer
Range in seconds	0 to n
	If the property is zero (0), the request never times out.
Default	600

- A Java administrative client. The property is AdminClient.CONNECTOR_IPC_REQUEST_TIMEOUT.

Secure Sockets Layer (SSL) security

Use this property to enable SSL security between Application Server and the IPC client. Set the property by using one of the following options:
- Scripts that run from a command-line interface.
- The `ipc.client.props` file.

Property	com.ibm.IPC.securityEnabled
Data type	Boolean
Default	False

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	securityEnabled
Data type	Boolean
Default	False

- A Java administrative client. Use the AdminClient.CONNECTOR_SECURITY_ENABLED property.

Security context provider

This property indicates the SSL implementation to use between the application server and the IPC client.

Set the property by using the `ipc.client.props` file.

Property	com.ibm.ssl.contextProvider
Data type	String
Valid Values	`IBMJSSE2`
Default	IBMJSSE2

SSL alias

This property specifies the alias to use for an SSL configuration for client connections. The value of the alias is what you want it to be.

Set the property in the `ipc.client.props` file.

Property	com.ibm.ssl.alias
Data type	String
Default	DefaultSSLSettings

SOAP, RMI, JSR160RMI, and IPC connector properties

This section discusses JMX connector properties that pertain to the following SOAP connectors, RMI connectors, JSR160RMI connectors, and IPC connectors:
- Connector type
- Disabling a connector
- Host
- Password
- Port

- User name

Connector type

A connector type of SOAP, RMI, JSR160RMI, or IPC depends on whether the application server connects to a SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts that run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	Type		
Data type	String		

Valid values	`SOAPConnector`
	`RMIConnector`
	`JSR160RMIConnector`
	`IPCConnector`		

Default	SOAPConnector	JSR160RMI	IPC

- A Java administrative client. Use the AdminClient.CONNECTOR_TYPE property. Specify the connector type by using the AdminClient.CONNECTOR_TYPE_RMI, the AdminClient.CONNECTOR_TYPE_SOAP, the AdminClient.CONNECTOR_TYPE_JSR160RMI, or the AdminClient.CONNECTOR_TYPE_IPC constants.

Disabling a connector

You can enable or disable any of the JMX connectors from the administrative console.

- The wsadmin tool.
- The administrative console. Select the box next to the connector to enable the connector. Clear the box next to the connector to disable the connector.

Property	enabled
Data type	Boolean
Value	true|false

Host

The host name or the IP address of the server to which the application server connects. The server can be a SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts that run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	host
Data type	String
Valid values	Host name or IP address
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_HOST property.

Password

The password that the application server uses to access the SOAP server, the RMI server, the JSR160RMI server, or the IPC server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts run from a command-line interface.
- The `soap.client.props` file for the SOAP server, the RMI server, or the JSR160RMI server.

Property	com.ibm.SOAP.loginPassword
Data type	String
Valid values	The value must match the global SSL settings for SOAP, RMI, or JSR160RMI.
Default	None

- The `ipc.client.props` file for the IPC server.

Property	com.ibm.IPC.loginPassword
Data type	String
Valid values	The value must match the global SSL settings for IPC.
Default	None

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	password
Data type	String
Valid values	The value must match the global SSL settings for SOAP, RMI, JSR160RMI, or IPC.
Default	None

- A Java administrative client. Use the AdminClient.PASSWORD property.

Port

The port number of the server to which the application server connects. The server can be a SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	port
Data type	Integer
Valid value	Port number
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_PORT property.

User name

The user name that the application server uses to access the SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts run from a command-line interface.
- The `soap.client.props` file for the SOAP server, an RMI server, a JSR160RMI server.

Property	com.ibm.SOAP.loginUserid
Data type	String
Valid value	The value must match the global SSL settings for SOAP, RMI, or JSR160RMI.
Default	None

- The `ipc.client.props` file for the IPC server.

Property	com.ibm.IPC.loginUserid
Data type	String
Valid value	The value must match the global SSL settings for IPC.
Default	None

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	username
Data type	String
Valid value	The value must match the global SSL settings for SOAP, RMI, JSR160RMI, or IPC.
Default	None

- A Java administrative client. Use the AdminClient.USERNAME property.

RMI connector properties

This section discusses the following JMX connector properties that pertain to RMI connectors:
- Disabling the JSR 160 RMI connector

Disabling the JSR 160 RMI connector

Support for JMX Remote application programming interface (JSR 160) is enabled by default so that you automatically receive specification-compliant JMX function. To disable the function for a particular server, set the property by using one of the following options:
- The wsadmin tool.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	disableJDKJMXConnector
Data type	string
Value	true

Java Management Extensions (JMX) connectors

Use this page to view and change the configuration for Java Management Extensions (JMX) connectors, which make connections between server processes. The types of JMX connectors are Simple Object Access Protocol (SOAP), Remote Method Invocation (RMI), JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI), and Inter-Process Communications (IPC).

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **JMX Connectors**.

Java Management Extensions (JMX) connectors communicate with WebSphere Application Server when you invoke a scripting process. There is no default for the type and parameters of a connector. The

`wsadmin.properties` file specifies the SOAP connector and an appropriate port number. You can also use the RMI connector, the JSR160RMI connector, or the IPC connector.

Use one of the following methods to select the connector type and attributes:
- Specify properties in a properties file.
- Indicate options on the command line.

On the z/OS platform, the SOAP connector, the RMI connector, the JSR160RMI connector, and the IPC connector connect to a controller (server) . WebSphere Application Server for z/OS internally routes MBean requests from a controller to its servant regions as appropriate.

Type
Specifies the type of the JMX connector.

Data type	Enumeration
Default	SOAPConnector
Range	**SOAPConnector**
	For JMX connections using Simple Object Access Protocol (SOAP).
	RMIConnector
	For JMX connections using Remote Method Invocation (RMI).
	JSR160RMIConnector
	For JMX connections using JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI).
	IPCConnector
	For JMX connections using Inter-Process Communications (IPC).

Enabled
Specifies whether a JMX connector is enabled. If `Yes` is specified, the connector is enabled. All JMX connectors are enabled by default.

To disable a JMX connector, select the connector and click **Disable**. The **Enabled** value changes to `No`. To enable a JMX connector, select the connector and click **Enable**. The Enabled value changes to `Yes`.

Data type	Boolean

JMX connector settings
Use this page to view the configuration for a Java Management Extensions (JMX) connector, which makes connections between server processes.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **JMX Connectors** > *connector_type*.

The Simple Object Access Protocol (SOAP) connector, the Remote Method Invocation (RMI) connector, the JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI) connector, or the Inter-Process Communications (IPC) connector connect to a controller (server). WebSphere Application Server for z/OS internally routes MBean requests from a controller to its servant regions as appropriate.

Type:

Specifies the type of the JMX connector.

Data type	Enumeration
Default	SOAPConnector

Range

SOAPConnector
    For JMX connections using Simple Object Access
    Protocol (SOAP).
RMIConnector
    For JMX connections using Remote Method
    Invocation (RMI).
JSR160RMIConnector
    For JMX connections using JMX Remote
    application programming interface (JSR 160)
    Remote Method Invocation (JSR160RMI).
IPCConnector
    For JMX connections using Inter-Process
    Communications (IPC).
    gotcha: IPC_CONNECTOR_ADDRESS is a
    special end point. The end point must be defined
    as 'localhost'. Editing a hostname definition from
    'localhost' to any other <**hostname**> is not
    permitted if you plan to use this same server as a
    template to create another server in another node
    and which is to have an end point definition
    different than <**hostname**>.

SOAP connector and Inter-Process Communications connector properties files

Use the `soap.client.props` file to set properties for the SOAP connector and the `ipc.client.props` file to
set properties for the Inter-Process Communications (IPC) connector. Most of the properties in the
`ipc.client.props` file have corresponding properties in the `soap.client.props` file.

The SOAP connector properties file for a particular profile is at the following location:

* *profile_root*/`properties/soap.client.props`

The IPC connector properties file for a particular profile is at the following location:

* *profile_root*/`properties/ipc.client.props`

The following table provides basic information on the various properties. Read the properties files to obtain
more detailed information.

*Table 30. SOAP connector and IPC connector property descriptions.   The properties configure the SOAP and IPC
connectors.*

SOAP connector properties	IPC connector properties	Description
com.ibm.SOAP.securityEnabled	com.ibm.IPC.securityEnabled	Specifies enablement of security for the connector. Set the property to `true` to enable security.
com.ibm.SOAP.authenticationTarget	com.ibm.IPC.authenticationTarget	Specifies the type of authentication for the connector if security is enabled. You can specify `BasicAuth` for basic authentication. If no value is specified, basic authentication is used.
com.ibm.SOAP.loginUserid	com.ibm.IPC.loginUserid	Specifies the user ID for the connector if security is enabled, and you do not enter a user ID through a command prompt or standard in.
com.ibm.SOAP.loginPassword	com.ibm.IPC.loginPassword	Specifies the password for the connector if security is enabled, and you do not enter a password through a command prompt or standard in.
com.ibm.SOAP.loginSource	com.ibm.IPC.loginSource	Specifies automatic prompting for the user ID and password when you specify `prompt`. Prerequisites for using this property are discussed in the properties file for the particular connector.

Table 30. SOAP connector and IPC connector property descriptions (continued). The properties configure the SOAP and IPC connectors.

SOAP connector properties	IPC connector properties	Description
com.ibm.SOAP.requestTimeout	com.ibm.IPC.requestTimeout	Specifies how long in seconds the connector waits for a server response. The property for the SOAP connector and the property for the IPC connector are each initially set to 180 in their respective properties files.
com.ibm.ssl.alias	com.ibm.ssl.alias	This property specifies the alias to use for a Secure Sockets Layer (SSL) configuration for client connections. The value of the alias is what you want it to be.
	timeToExpiration	Specifies the time in seconds that connections can be idle in the connection pool. Beyond this time the connections are purged. The initial setting for the property is 360.

Note: Many of the system management commands contain implicit stop server operations. Most of these commands use the IPC connector properties configuration. However, some commands, such as the stopServer, stopNode, and stopManager commands, continue to use the SOAP connector properties configuration for compatibility reasons. Thus, to avoid additional user ID and password prompts, specify the user ID and password information in both the `soap.client.props` and `ipc.client.props` files.

Extension MBean Providers collection

Use this page to view and change the configuration for JMX extension MBean providers.

You can configure JMX extension MBean providers to be used to extend the existing WebSphere managed resources in the core administrative system. Each MBean provider is a library containing an implementation of a JMX MBean and its MBean XML Descriptor file.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Extension MBean Providers**.

Name

The name used to identify the Extension MBean provider library.

Description

An arbitrary descriptive text for the Extension MBean Provider configuration.

Classpath

The path to the Java archive (JAR) file that contains the Extension MBean provider library. This class path is automatically added to the Application Server class path.

Extension MBean Provider settings

Use this page to view and change the configuration for a JMX extension MBean provider.

You can configure a library containing an implementation of a JMX MBean, and its MBean XML Descriptor file, to be used to extend the existing WebSphere managed resources in the core administrative system

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Extension MBean Providers** > *provider_library_name*.

Name:

The name used to identify the Extension MBean provider library.

| **Data type** | String |

Classspath:

The path to the Java archive (JAR) file that contains the Extension MBean provider library. This class path is automatically added to the Application Server class path. The class loader needs this information to load and parse the Extension MBean XML Descriptor file.

| **Data type** | String |

Description:

An arbitrary descriptive text for the Extension MBean Provider configuration. Use this field for any text that helps identify or differentiate the provider configuration.

| **Data type** | String |

Extension MBean collection

You can configure Java Management Extension (JMX) MBeans to extend the existing WebSphere Application Server managed resources in the administrative console. Use this page to register JMX MBeans. Any MBeans that are listed have already been registered.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Extension MBean Providers** > *provider_library_name* > **extensionMBeans**.

descriptorURI

Specifies the location, relative to the provider class path, where the MBean XML descriptor file is located.

type

Specifies the type to use for registering this MBean. The type must match the type that is declared in the MBean descriptor file.

Extension MBean settings

Use this page to view and configure Java Management Extension (JMX) MBeans.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Extension MBean Providers** > *provider_library_name* > **extensionMBeans** > *descriptorURI*.

descriptorURI:

Specifies the location, relative to the provider class path, where the MBean XML descriptor file is located.

| **Data type** | String |

type:

Specifies the type to use for registering this MBean. The type must match the type that is declared in the MBean descriptor file.

| **Data type** | String |

Administrative audit messages in system logs

The product provides administrative audit messages in system logs that contain some audit information. The audit messages described in this topic are part of the standard product audit stream and do not provide administrative event auditing information such as who changed files.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log`, `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

Important: The functionality described in this topic uses system logs and is not a part of the security auditing subsystem. The audit information captured by this functionality does not correspond with the audit information captured by the security auditing subsystem. For information about the security auditing subsystem, see the topic on auditing the security infrastructure.

Administrative audits use the same trace logging facility as the rest of the product, and do not use the logging facility that is a part of the security auditing subsystem. The audits are available in both the `activity.log` file and the `SystemOut.log` of the server that performs the action. You do not need to enable trace to produce the audits. However, through the Repository service console page, you can control whether configuration change auditing is done. This type of audit is done by default. Operational command auditing is always enabled. Information about which user performed the change is available only when security is enabled.

You can do administrative audits with or without the security audit facility.

The following administrative actions are audited:
- All configuration changes, in terms of the configuration documents that are created, modified, or deleted.
- Certain operational changes like starting and stopping nodes, clusters, servers, and applications. These managed bean (MBean) operations provide administrative auditing:

Table 31. Administrative auditing MBean operations. The MBean types provide administrative auditing MBean operations.

MBean type	MBean operations
CellSync	syncNode
Cluster	start, stop, stopImmediate, rippleStart
NodeAgent	launchProcess, stopNode, restart
Server	stop, stopImmediate
AppManagement	startApplication, stopApplication

Configuration change audits have ADMRxxxxI message IDs, where xxxx is the message number. Operational audits have ADMN10xxI message IDs, where 10xx is the message number.

Here are some audit examples from a WebSphere Application Server, Network Deployment environment.

The following audit example is from the deployment manager `SystemOut.log` file:

```
[7/23/03 17:04:49:089 CDT] 39c26dad FileRepositor A ADMR0015I: Document
cells/ellingtonNetwork/security.xml was modified by user u1.
   [7/23/03 17:04:49:269 CDT] 3ea0edb5 FileRepositor A ADMR0016I: Document
cells/ellingtonNetwork/nodes/ellington/app.policy was created by user u1.
```

```
  ...
    [7/23/03 17:13:54:081 CDT] 39a572a1 AdminHelper   A ADMN1008I: Attempt
made to start the SamplesGallery application. (User ID = u1)
  ...
```

The following audit example is from the node agent `SystemOut.log` file:

```
[7/23/03 17:38:43:461 CDT]  23d1326 AdminHelper   A ADMN1000I: Attempt
made to launch server1 on node ellington. (User ID = u1)
```

The following audit example is from the application server`SystemOut.log` file:

```
[7/23/03 17:39:59:360 CDT] 24865373 AdminHelper   A ADMN1020I: Attempt
made to stop the server1 server. (User ID = u1)
```

The message text is split for printing purposes.

Administration service settings

Use this page to view and change the configuration for an administration service.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services**.

Remote connector

Specifies the remote JMX Connector type. The remote JMX connector is the connector that is used between server processes that reside on different physical machines, for example, between the deployment manager and the node agent. Available options of SOAPConnector, RMIConnector, and JSR160RMI Connector are defined using the JMX Connectors page.

Data type	String
Default	SOAPConnector

Local connector

Specifies the local JMX Connector type. The local JMX connector is the connector used between server processes that reside on the same physical machine, for example, between the node agent and its application servers. Available options of SOAPConnector, RMIConnector, JSR160RMI Connector, and IPC Connector are defined using the JMX Connectors page.

Data type	String
Default	IPC Connector

Administration services custom properties

This topic discusses the administration services custom properties that you can set on the administrative console.

To view the administration services custom properties administrative console page that goes with this topic, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Administration** > **Administration services** > **Custom properties**.

Specify a property and its value as a name-value pair on the Administration services custom properties page. You can use the custom properties page to define the following administration services custom properties:

- "com.ibm.websphere.mbeans.disableRouting" on page 213

com.ibm.websphere.mbeans.disableRouting

When a custom managed bean (MBean) is registered directly with the MBean server that runs in a WebSphere Application Server process, the MBean object name is enhanced by default to include the cell, node, and process names as key properties. To turn off the default behavior, set the following custom property on the application server.

With this enhancement, in a WebSphere Application Server, Network Deployment environment, the MBean that is registered on an application server is addressable through a client that is connected to the deployment manager.

If this custom property is set, an administrative client needs to connect directly to the application server on which the MBean is registered to invoke methods. The MBean cannot participate in all the distributed functions of the administrative system.

One or more MBean object names tagged with `<on>...</on>`. You can specify the object name of your MBean or a pattern that matches the names of several MBeans.

Example:

If you register a custom MBean with the `WebSphere:type=custom,name=custommbean1` object name and another custom MBean with the `WebSphere:type=custom,name=custommbean2` object name, each of the following values is valid:

- `<on>WebSphere:type=custom,name=custommbean1</on>`

  The value disables the MBean object name modification for this MBean.

- `<on>WebSphere:type=custom,*</on>`

  The value disables the MBean object name modification for both MBeans.

- `<on>WebSphere:type=custom,name=custommbean1</on><on>WebSphere:type=custom,name=custommbean2</on>`

  The value disables the object name modification for both MBeans.

Managing nodes and resources on z/OS

The z/OS location service daemon starts automatically if it is not already running when you start an application server or a deployment manager server. You can manage the location service daemon in a console. Location service daemons provide the CORBA location service in support of Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP). The daemon works with z/OS workload management to distribute RMI requests (for example, enterprise bean requests) among application servers in a cell. A location service daemon process runs on each system that has a node in a sysplex node group. One location service daemon exists for each sysplex node group in a cell.

Stopping or canceling the z/OS location service daemon from the MVS console

Location service daemons provide the CORBA location service in support of Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP). This topic discusses how to issue MVS™ console commands to stop or cancel the z/OS location service daemon.

Before you begin

You must first install WebSphere Application Server.

About this task

If you cancel or stop the location service daemon, it cancels or stops all WebSphere Application Servers on the system. If you installed WebSphere Application Server, Network Deployment, the location service daemon also cancels or stops the deployment manager and the node agents.

Procedure

Issue one of the following commands to stop the location service daemon:

```
STOP DAEMON01
CANCEL DAEMON01
CANCEL DAEMON01,ARMRESTART
```

If you issue the **stop** command, the server finishes all remaining work before shutting itself down. If you issue the **cancel** command, the server stops quickly. Inflight data and transactions might be lost.
Use the **cancel** command that has the ARMRESTART option if automatic restart management (ARM) is active and you want the ARM to restart the location service daemon.

Results

You have stopped or cancelled the location service daemon.

Determining if the z/OS location service daemon is running

This topic describes what steps you must follow to determine if the location service daemon is running on a z/OS system.

Before you begin

The location service daemon starts automatically if it is not already running when you start an Application Server or a deployment manager server.

About this task

Perform the following step any time you want to know whether the location service daemon is running.

Procedure

To determine if the location service daemon is running, issue one of the display commands , such as d a,l, from the MVS console.

Results

You know the system is running if you see the BBODMN*x* address space running, where *x* is a letter (A, B, C, and so on).

Modifying z/OS location service daemon settings

This article describes what steps you must follow to modify the location service daemon settings in an administrative console. Location service daemons provide the CORBA location service in support of Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP).

Before you begin

Complete the following items before starting this task:
• Configure the location service daemon

You must first configure the location service daemon in the WebSphere Application Server customization dialogs. After you configure the daemon, you can modify or view the location service daemon settings from the administrative console.

- Optionally enable security

  There is no specific setting to enable Secure Sockets Layer (SSL) for the location service daemon. If you want to use the SSL protocol to encrypt communications to the location service daemon, complete the following items:

  – Enable global security for the cell.

  – Define a valid system SSL repertoire for the location service daemon.

  – Set the z/OS user ID that is assigned to the location service daemon-started task to the same z/OS user ID that was used to create the key ring.

About this task

The location service daemon is an integral component of the Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP) communication function for the WebSphere Application Server for z/OS. The daemon works with z/OS workload management to distribute RMI requests (for example enterprise bean requests) among application servers in a cell.

When a client makes a remote call to an enterprise bean, a location service daemon determines which servers are eligible to process the request. The location service daemon makes the decision with the z/OS workload management function (WLM). The daemon then routes the request to the selected server, which establishes a CORBA session with the client. Subsequent calls to the same enterprise bean flow directly over the established session.

In a cell, one location service daemon exists for each sysplex node group. A location service daemon process runs on each system that has a node in a sysplex node group. An example of a system is the z/OS operating system on a logical partition (LPAR).

You can now modify the location service daemon settings in an administrative console.

Procedure

1. Go to the Settings page for the location service daemon to view the help page.

2. For the administrative console, click **System Administration > Node groups > _sysplex node group_ > z/OS location service daemon** in the console navigation tree. For the job manager administrative console, click **System Administration -> Job manager > z/OS Location Service** in the console navigation tree. For the administrative agent administrative console, click **System Administration -> Administrative agent > z/OS Location Service** in the console navigation tree.

3. Specify the following values for the location service daemon:

   - The job name
   - The ports on which it listens
   - The IP names on which it listens
   - The start command
   - The start command arguments
   - The SSL repertoire that it uses

Results

The location service daemon settings are modified.

z/OS location service daemon settings

In a cell, one location service daemon definition exists for each sysplex node group. A location service daemon process runs on each system that has a node in a sysplex node group in that cell. When a client

makes a remote call to an enterprise bean, a location service daemon determines which server or servers are eligible to process the request, and routes the request to the selected server. An example of a system is the z/OS operating system on a logical partition (LPAR).

Changes made to these settings apply to the location service daemon in a sysplex node group.

This administrative console page is only visible if the node group contains z/OS nodes. To view this administrative console page, click **System Administration > Node groups > *sysplex node group* > z/OS location service daemon**.

Job name:

Specifies the job name of the location service daemon. This name can be from one to eight characters. You can use alphanumeric or the national language characters of @#$.

Data type	String
Default	None

Start command:

Specifies the command string that servers use to automatically start the location service daemon.

Data type	String
Format	START *location service daemon JCL procedure name*
	The procedure name is defined at the node level during customization. You can change the procedure name on the WebSphere Variables administrative console panels by going to **Environment > WebSphere variables**.
Example	START BBO6DMN
Default	${NODE_DAEMON_START_COMMAND}
	NODE_DAEMON_START_COMMAND is the configuration variable name whose value you can change on the WebSphere Variables administrative console panels.

Listen IP name:

Specifies the IP address on which the location service daemon listens. The Listen IP name must be either a dotted decimal IP address or an asterisk (*). The asterisk means that the location service daemon listens on all available IP addresses.

Data type	String
Default	asterisk (*)

Daemon IP name:

Specifies the IP name that clients use to access enterprise beans and Common Object Request Broker Architecture (CORBA) components on servers that belong to the sysplex node group that this location service daemon serves.

The daemon IP name must be a DNS name and not a dotted-decimal IP address. The value must contain more than one qualifier such as "foo.com" and not a single qualifier, such as "foo."

Data type	String
Default	None

Port:

Specifies the port on which the location service daemon listens for Remote Method Invocation and Internet Inter-ORB (RMI/IIOP) requests.

Because the system reserves port numbers less than 1024 for Transmission Control Protocol/Internet Protocol (TCP/IP) applications, the recommendation is to use port numbers greater than 1023. However, the port range starts at 1.

Data type	Integer
Default	None
Range	1 to 65535

SSL port:

Specifies the port on which the location service daemon listens for encrypted Remote Method Invocation and Internet Inter-ORB (RMI/IIOP) requests.

Because the system reserves port numbers less than 1024 for Transmission Control Protocol/Internet Protocol (TCP/IP) applications, the recommendation is to use port numbers greater than 1023. However, the port range starts at 0. A value of 0 disables the SSL port.

Data type	Integer
Default	None
Range	0 to 65535

SSL settings:

Specifies a predefined list of Secure Sockets Layer settings for connections.

These settings are System SSL repertoires that you configured on the SSL repertoire panel. Select one repertoire from the list.

Administrative topology: Resources for learning

Use the following links to find relevant supplemental information about WebSphere Application Server administrative topologies and distributed administration. The information resides on IBM and non-IBM Internet sites, whose sponsors control the technical accuracy of the information.

These links are provided for convenience. Often, the information is not specific to the product, but is useful all or in part for understanding the product. When possible, links are provided to technical papers and IBM Redbooks® that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

View links to additional information:

Administration
- IBM WebSphere Application Server Redbooks

  The site contains a listing of all WebSphere Application Server Redbooks.
- IBM WebSphere developerWorks

  The site is the home of technical information for developers working with WebSphere products. You can download WebSphere software, take a fast path to developerWorks® zones, learn about WebSphere products through a newcomers page, tutorials, technology previews, training, and Redbooks, get

answers to questions about WebSphere products, and join the WebSphere community, where you can keep up with the latest developments and technical papers.

- WebSphere Application Server Support page

  Take advantage of the web-based Support and Service resources from IBM to quickly find answers to your technical questions. You can easily access this extensive web-based support through the IBM Software Support portal at URL `http://www.ibm.com/software/support/` and search by product category or by product name. For example, if you are experiencing problems specific to the product, click **WebSphere Application Server** in the product list. The Support page is displayed.

Chapter 3. Working with server configuration files

This topic show how to manage application server configuration files.

About this task

Application server configuration files define the available application servers, their configurations, and their contents.

A configuration repository stores configuration data.

By default, configuration repositories reside in the *config* subdirectory of the profile root directory.

Note: Do not store any non-default XML or XML backup files under the *profile_root*/config directory. Limit the config directory and subdirectories to valid default WebSphere Application Server configuration files. A variety of symptoms and exceptions are possible if non-default files are stored in this directory.

A cell-level repository stores configuration data for the entire cell and is managed by a file repository service that runs in the deployment manager. The deployment manager and each node have their own repositories. A node-level repository stores configuration data that is needed by processes on that node and is accessed by the node agent and application servers on that node.

When you change a WebSphere Application Server configuration by creating an application server, installing an application, changing a variable definition or the like, and then save the changes, the cell-level repository is updated. The file synchronization service distributes the changes to the appropriate nodes.

You should periodically save changes to your administrative configuration. You can change the default locations of configuration files, as needed.

Procedure

- Edit configuration files.

  The master repository is comprised of .xml configuration files

  You can edit configuration files using

  - The administrative console. See the Using the administrative console topic in the *Using the administrative clients* PDF.
  - Scripting. See the Getting started with scripting topic in the *Using the administrative clients* PDF.
  - The wsadmin commands. See the Using command line tools topic in the *Using the administrative clients* PDF.
  - Programing. See the Using administrative programs (JMX) topic in the *Using the administrative clients* PDF.
  - By editing a configuration file directly.

  The configuration files are in ASCII format. You cannot edit them in the Hierarchical File System (HFS). Instead, transfer the files to your workstation, edit them, and then transfer them back to the HFS.
- Save changes made to configuration files. Using the console, you can save changes as follows:
  1. In the navigation select **System Administration > Save changes to master repository**.
  2. Put a check mark in the **Synchronize changes with Nodes** check box.
  3. Click **Save**.
- Handle temporary configuration files resulting from a session timing out.
- Change the location of temporary configuration files.
- Change the location of backed-up configuration files.

- Change the location of temporary workspace files.
- Back up and restore configurations.

Configuration documents

WebSphere Application Server stores configuration data in several documents in a cascading hierarchy of directories. Most configuration documents have XML content.

The configuration documents describe the available application servers, their configurations, and their contents.
- "Hierarchy of directories of documents"
- "Changing configuration documents" on page 221
- "Transformation of configuration files" on page 221

Hierarchy of directories of documents

The cascading hierarchy of directories and the documents' structure support multinode replication to synchronize the activities of all servers in a cell. In a WebSphere Application Server, Network Deployment environment, changes made to configuration documents in the cell repository, are automatically replicated to the same configuration documents that are stored on nodes throughout the cell.

At the top of the hierarchy is the **cells** directory. It holds a subdirectory for each cell. The names of the cell subdirectories match the names of the cells. For example, a cell named *cell1* has its configuration documents in the subdirectory *cell1*. The name of the cell must be different from the cluster name pair.

On the WebSphere Application Server, Network Deployment node, the subdirectories under the cell contain the entire set of documents for every node and server throughout the cell. On other nodes, the set of documents is limited to what applies to that specific node. If a configuration document only applies to *node1*, then that document exists in the configuration on *node1* and in the WebSphere Application Server, Network Deployment configuration, but not on any other node in the cell.

Each cell subdirectory has the following files and subdirectories:
- The `cell.xml` file, which provides configuration data for the cell
- Files such as `security.xml`, `virtualhosts.xml`, `resources.xml`, and `variables.xml`, which provide configuration data that applies across every node in the cell
- The **clusters** subdirectory, which holds a subdirectory for each cluster defined in the cell. The names of the subdirectories under clusters match the names of the clusters.

    Each cluster subdirectory holds a cluster.xml file, which provides configuration data specifically for that cluster.
- The **nodes** subdirectory, which holds a subdirectory for each node in the cell. The names of the nodes subdirectories match the names of the nodes.

    Each node subdirectory holds files such as `variables.xml` and `resources.xml`, which provide configuration data that applies across the node. Note that these files have the same name as those in the containing cell's directory. The configurations specified in these node documents override the configurations specified in cell documents having the same name. For example, if a particular variable is in both cell- and node-level `variables.xml` files, all servers on the node use the variable definition in the node document and ignore the definition in the cell document.

    Each node subdirectory holds a subdirectory for each server defined on the node. The names of the subdirectories match the names of the servers. Each server subdirectory holds a `server.xml` file, which provides configuration data specific to that server. Server subdirectories might hold files such as `security.xml`, `resources.xml` and `variables.xml`, which provide configuration data that applies only to the server. The configurations specified in these server documents override the configurations specified in containing cell and node documents having the same name.

- The **applications** subdirectory, which holds a subdirectory for each application deployed in the cell. The names of the applications subdirectories match the names of the deployed applications.

  Each deployed application subdirectory holds a `deployment.xml` file that contains configuration data on the application deployment. Each subdirectory also holds a **META-INF** subdirectory that holds a Java 2 Platform, Enterprise Edition (J2EE) application deployment descriptor file as well as IBM deployment extensions files and bindings files. Deployed application subdirectories also hold subdirectories for all .war and entity bean .jar files in the application. Binary files such as .jar files are also part of the configuration structure.

An example file structure is as follows:

```
cells
  cell1
    cell.xml resources.xml virtualhosts.xml variables.xml security.xml
    nodes
      nodeX
        node.xml variables.xml resources.xml serverindex.xml
        serverA
          server.xml variables.xml
        nodeAgent
          server.xml variables.xml
      nodeY
        node.xml variables.xml resources.xml serverindex.xml
    applications
      sampleApp1
        deployment.xml
        META-INF
          application.xml ibm-application-ext.xml ibm-application-bnd.xml
      sampleApp2
        deployment.xml
        META-INF
          application.xml ibm-application-ext.xml ibm-application-bnd.xml
```

Changing configuration documents

You can use one of the administrative tools (console, wsadmin, Java APIs) to modify configuration documents or edit them directly. It is preferable to use the administrative console because it validates changes made to configurations. ""Configuration document descriptions" on page 222" states whether you can edit a document using the administrative tools or must edit it directly.

Note: The following z/OS variable definitions no longer exist in Version 8.0 configuration documents:

- `private_Enable_zWAS_for_64bit` in server scope `variables.xml`
- `AMODE=64` in processDefinition for control, servant, or adjunct processes in `server.xml`
- `was.com.ibm.websphere.zos.jvmmode` in processDefinition for control processes in `server.xml`

  In Version 8.0, you do not see `AMODE=64` in Start command arguments for the server process. To see the current bit mode of the server:

- Using wsadmin, run AdminTask commands to get the bit mode used.
- Using the administrative console, see **Run in 64 bit JVM Mode** on the application server settings page. Click **Servers** > **Server Types** > **WebSphere application servers** > ***server_name***.

Transformation of configuration files

The WebSphere Application Server master configuration repository stores configuration files for all the nodes in the cell. When you upgrade the deployment manager from one release of WebSphere Application Server to another, the configuration files that are stored in the master repository for the nodes on the old release are converted into the format of the new release.

With this conversion, the deployment manager can process the configuration files uniformly. However, nodes on an old release cannot readily use configuration files that are in the format of the new release. WebSphere Application Server addresses the problem when it synchronizes the configuration files from the master repository to a node on an old release. The configuration files are first transformed into the old release format before they ship to the node. WebSphere Application Server performs the following transformations on configuration documents:

- Changes the XML name space from the format of the new release to the format of the old release
- Strips out attributes of cell-level documents that are applicable to the new release only
- Strips out new resource definitions that are not understood by old release nodes

Configuration document descriptions

Most configuration documents have XML content. The table describes the documents and states whether you can edit them using an administrative tool or must edit them directly.

If possible, edit a configuration document using the administrative console because it validates any changes that you make to configurations. You can also use one of the other administrative tools (wsadmin or Java APIs) to modify configuration documents. Using the administrative console or wsadmin scripting to update configurations is less error prone and likely quicker and easier than other methods.

However, you cannot edit some files using the administrative tools. Configuration files that you must edit manually have an X in the **Manual editing required** column in the table below.

Document descriptions

(The paths in the Locations column are split on multiple lines for publishing purposes.)

Configuration file	Locations	Purpose	Manual editing required
admin-authz.xml	config/cells/ *cell_name*/	Define a role for administrative operation authorization.	
app.policy	config/cells/ *cell_name*/ nodes/*node_name*/	Define security permissions for application code.	X
cell.xml	config/cells/ *cell_name*/	Identify a cell.	
deployment.xml	config/cells/ *cell_name*/ applications/ *application_name*/	Configure application deployment settings such as target servers and application-specific server configuration.	
filter.policy	config/cells/ *cell_name*/	Specify security permissions to be filtered out of other policy files.	X
integral-jms-authorizations.xml	config/cells/ *cell_name*/	Provide security configuration data for the integrated messaging system.	X
library.policy	config/cells/ *cell_name*/ nodes/*node_name*/	Define security permissions for shared library code.	X
namestore.xml	config/cells/ *cell_name*/	Provide persistent name binding data.	X

naming-authz.xml	config/cells/ *cell_name*/	Define roles for a naming operation authorization.	X
node.xml	config/cells/ *cell_name*/ nodes/*node_name*/	Identify a node.	
resources.xml	config/cells/ *cell_name*/ config/cells/ *cell_name*/ nodes/*node_name*/ config/cells/ *cell_name*/ nodes/*node_name*/ servers/ *server_name*/	Define operating environment resources, including JDBC, JMS, JavaMail, URL, JCA resource providers and factories.	
security.xml	config/cells/ *cell_name*/	Configure security, including all user ID and password data.	
server.xml	config/cells/ *cell_name*/ nodes/ *node_name*/ servers/ *server_name*/	Identify a server and its components.	
serverindex.xml	config/cells/ *cell_name*/ nodes/ *node_name*/	Specify communication ports used on a specific node.	
spi.policy	config/cells/ *cell_name*/ nodes/ *node_name*/	Define security permissions for service provider libraries such as resource providers.	X
variables.xml	config/cells/ *cell_name*/ config/cells/ *cell_name*/ nodes/ *node_name*/ config/cells/ *cell_name*/ nodes/*node_name*/ servers/ *server_name*/	Configure variables used to parameterize any part of the configuration settings.	
virtualhosts.xml	config/cells/ *cell_name*/	Configure a virtual host and its MIME types.	

Object names: What the name string cannot contain

When you create a new object using the administrative console or a wsadmin command, you often must specify a string for a name attribute.

Most characters are allowed in the name string. However, the name string cannot contain the following characters. The name string also cannot contain leading and trailing spaces.

/	forward slash
\	backslash
*	asterisk
,	comma
:	colon
;	semi-colon
=	equal sign
+	plus sign
?	question mark
l	vertical bar
<	left angle bracket
>	right angle bracket
&	ampersand (and sign)
%	percent sign
'	single quote mark
"	double quote mark
]]>	No specific name exists for this character combination.
.	period (not valid if first character; valid if a later character)
#	Hash mark
$	Dollar sign
~	Tilde
(	Left parenthesis
)	Right parenthesis

gotcha:

- Character restrictions are not enforced for DataSource, ServiceLog, GroupExt, UserExt, or SubjectExt object names.
- You can use one of the following methods to turn off character validation for custom property names, and for the name value of Property and J2EEResourceProperty configuration objects in wsadmin commands.
  - Set the com.ibm.websphere.management.configservice.validatePropNames Java system property to `false` in the Java virtual machine (JVM) for the deployment manager server.
  - Set the com.ibm.websphere.management.configservice.validatePropNames property using the **-javaoption** parameter when you use the wsadmin tooling in the local mode.

    ```
    wsadmin -conntype none -javaoption
    "-Dcom.ibm.websphere.management.configservice.validatePropNames=false"
    ```

Handling temporary configuration files resulting from session timeout

If the console is not used for 15 minutes or more, the session times out. The same thing happens if you close the browser window without saving the configuration file. Changes to the file are saved to a temporary file when the session times out, after 15 minutes. This topic discusses what happens depending on whether you load the saved file.

Before you begin

A configuration file must have been saved from a previous administrative console session for the user ID that you are currently using to access the administrative console.

About this task

When a session times out, the configuration file in use is saved under the `userid/timeout` directory under the ServletContext's temp area. This value is the value of the javax.servlet.context.tempdir attribute of the ServletContext context. By default, it is: *profile_root*/temp/hostname/Administration/admin/admin.war

You can change the temp area by specifying it as a value for the tempDir init-param of the action servlet in the deployment descriptor (web.xml) of the administrative application.

The configuration file is also saved automatically when the same user ID logs into the non-secured console again, effectively starting a different session. This process is equivalent to forcing the existing user ID out of session, similar to a session timing out.

The next time you log on to the administrative console, you are prompted to load the saved configuration file. Do one of the following actions:

Procedure

- Load the saved file.
    1. If a file with the same name exists in the *profile_root*/config directory, that file is moved to the userid/backup directory in the temp area.
    2. The saved file is moved to the *profile_root*/config directory.
    3. The file is then loaded.
- Do not load the saved file.

    The saved file is deleted from the userid/timeout directory in the temp area.

Results

You loaded the saved configuration file if you chose to do so.

What to do next

Once you have logged into the administrative console, do whatever administration of WebSphere Application Server that you need to do.

Changing the location of temporary configuration files

You can change the default directory where temporary configuration files are stored.

About this task

The configuration repository uses copies of configuration files and temporary files while processing repository requests. It also uses a backup directory while managing the configuration. You can change the default locations of these files from the configuration directory to a directory of your choice by using the administrative console.

The default location for the configuration temporary directory is *profile_root*/config/temp. Use the administrative console to change the location of the temporary repository file location for all types of server processes. For example, to change the setting for Application Server, do the following steps:

Procedure

1. Click **Servers > Application servers** in the navigation tree of the administrative console. Then, click ***server name* > Administration > Administration services > Repository service > Custom properties**.
2. On the Properties page, click **New**.
3. On the settings page for a property, define a property for the temporary file location. The key for this property is was.repository.temp. The value is the full path name to the desired location.
4. Click **OK**.

Changing the location of backed-up configuration files

You can change the default directory where backup files are stored.

About this task

During administrative processes like adding a node to a cell or updating a file, configuration files are temporarily backed up to a backup location.

The default location for the backup configuration directory is *profile_root*/config/backup. Use the administrative console to change the location of the repository backup directory for all types of server processes. For example, to change the setting for Application Server, do the following steps:

Procedure
1. Click **Servers > Application servers** in the navigation tree of the administrative console. Then, click *server name* > **Administration > Administration services > Repository service > Custom properties**.
2. On the Properties page, click **New**.
3. On the settings page for a property, define a property for the backup file location. The key for this property is `was.repository.backup`. The value is the full path name to the desired location.
4. Click **OK**.

Changing the location of the wstemp temporary workspace directory

Configuration changes are stored in the `wstemp` temporary workspace directory until the changes are merged with the master configuration repository. This topic discusses how to change the location of the `wstemp` temporary workspace directory.

Before you begin

You must first install WebSphere Application Server before you change the location of the `wstemp` directory, which is a temporary workspace directory.

About this task

Whenever a user logs into the administrative console, or uses wsadmin scripting to make a configuration change, the changes are stored in the workspace. When a user uses the ConfigService configuration service interface of the Java application programming interfaces (APIs), the user specifies a session object that is associated with the workspace in order to store the changes. Only when the user performs a save operation under the administrative console, wsadmin scripting, or the Java APIs are the changes propagated and merged with the master configuration repository. For each administrative console user or each invocation of wsadmin scripting, the application server creates a separate workspace directory to store the intermediate changes until the changes are merged with the master configuration repository. Users of the Java APIs use different session objects to decide where the workspace directory resides. Both the administrative console and wsadmin scripting generate user IDs randomly. The user IDs are different from the user IDs that you use to log into the administrative console or wsadmin scripting. The Java APIs can either randomly generate the user ID or specify the user ID as an option when creating the session object.

You might want to change the location of the `wstemp` directory if you want to keep it in a separate place from the product installation.

The product determines the location of the workspace in the following order by using the first Java virtual machine (JVM) property in the list that is set. If no JVM property is set, the product uses the default workspace location.

Table 32. Workspace locations of JVM system properties. The Location column states the `wstemp` directory location for specified JVM system properties.

JVM system property	Location	Comments
websphere.workspace.root	The `wstemp` directory location is the value of the websphere.workspace.root JVM system property plus • /wstemp For example, the websphere.workspace.root JVM system property and its value could be • `-Dwebsphere.workspace.root` `=/temp` The property and its value are split on multiple lines for printing purposes.	Set the JVM system property for the deployment manager to change the `wstemp` directory location. Use the full path rather than a relative path for this property.
If the websphere.workspace.root property is not set, the value of the user.install.root property is used.	The default `wstemp` location is the value of the user.install.root JVM system property plus • /wstemp	Do not change the user.install.root property as the profile creation process sets this property by pointing to the *profile_root* directory. In this case, the wstemp location is: • *profile_root*/wstemp

Procedure

• Change the workspace location for a particular JVM property by setting the -D option on the java command.

  This method of changing the workspace location is only needed when you run a stand-alone administrative program in local mode.

  For example, use the following option:

`-Dwebsphere.workspace.root=the location of the new workspace directory`

• Change the JVM custom property through the administrative console by setting the JVM property as a name-value pair on the Custom properties page.

  For example,

  1. Click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Java and Process Management** > **Process definition** > **Java Virtual Machine** > **Custom properties**.

  2. Click **New**.

  3. Specify `websphere.workspace.root` as the name.

  4. Specify the full path of the new workspace directory as the value. The `wstemp` directory is created under that path.

  5. Stop the server.

     This step is optional if you want to keep your existing workspace files.

  6. Copy files from the old location of the workspace directory to the new location of the workspace directory.

     This step is optional if you want to keep your existing workspace files.

  7. Start the server.

     This step is optional if you want to keep your existing workspace files.

Results

You have used either the administrative console or the -D option on the java command to change the location of the `wstemp` temporary workspace directory.

Backing up and restoring administrative configuration files

You can back up administrative configuration files using the backupConfig command. You can restore administrative configuration files using the restoreConfig command.

About this task

WebSphere Application Server represents its administrative configurations as XML files. You should back up configuration files on a regular basis.

Restore the configuration only if the configuration files that you backed up are at the same level of the release, including fixes, as the release to which you are restoring.

Procedure

1. Synchronize administrative configuration files.
   a. Click **System administration** > **Nodes** in the console navigation tree to access the Nodes page.
   b. Click **Full Resynchronize**. The resynchronize operation resolves conflicts among configuration files and can take several minutes to run.
2. Run the backupConfig command to back up configuration files. See the backupConfig command topic in the *Using the administrative clients* PDF for information.
3. Run the restoreConfig command to restore configuration files. See the restoreConfig command topic in the *Using the administrative clients* PDF for information. Specify backup files that do not contain invalid or inconsistent configurations.

Backing up the WebSphere Application Server for z/OS system

This topic discusses methods for backing up configuration and data for the WebSphere Application Server for z/OS system.

About this task

Use the following guidelines to back up parts of your WebSphere Application Server for z/OS system.

Procedure

1. **Back up the HFS that contains your WebSphere Application Server for z/OS configuration** (e.g. WebSphere/V5R0M0/AppServer).
2. Back up the RMDATA log for Resource Recovery Service (RRS). Otherwise, a failure could force you to do a cold start of RRS.
3. Set the ARCHIVE log retention period to one day.
4. Incorporate the following items in your normal backup procedures:
   - WebSphere Application Server for z/OS procedure libraries.
   - WebSphere Application Server for z/OS load libraries.
   - The directory where WebSphere Application Server for z/OS run-time information is written. The default is */WebSphere/V5R0M0*.
5. Back up your own application executable files, databases, and bindings.
6. If you wish to back up a single server, you can use the export/import function in the Administrative Console. For details on how to do this, see the assembling applications information.

Server configuration files: Resources for learning

Use the following links to find relevant supplemental information about administering WebSphere Application Server configuration files. The information resides on IBM and non-IBM Internet sites, whose sponsors control the technical accuracy of the information.

These links are provided for convenience. Often, the information is not specific to the IBM WebSphere Application Server product, but is useful all or in part for understanding the product. When possible, links are provided to technical papers and IBM Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

View links to additional information:

Administration

- IBM WebSphere Application Server Redbooks

  This site contains a listing of all WebSphere Application Server Redbooks.
- IBM WebSphere developerWorks

  This site is the home of technical information for developers working with WebSphere products. You can download WebSphere software, take a fast path to developerWorks zones, such as VisualAge® Java or WebSphere Application Server, learn about WebSphere products through a newcomers page, tutorials, technology previews, training, and Redbooks, get answers to questions about WebSphere products, and join the WebSphere community, where you can keep up with the latest developments and technical papers.
- WebSphere Application Server Support page

  Take advantage of the Web-based Support and Service resources from IBM to quickly find answers to your technical questions. You can easily access this extensive Web-based support through the IBM Software Support portal at URL `http://www.ibm.com/software/support/` and search by product category, or by product name. For example, if you are experiencing problems specific to WebSphere Application Server, click **WebSphere Application Server** in the product list. The WebSphere Application Server Support page appears.

Configuration problem settings

Use this page to identify and view problems that exist in the current configuration.

To view this administrative console page, click **Troubleshooting > Configuration Problems** in the console navigation tree.

To view a configuration problem, click **Configuration Validation** in the console navigation tree, then select the type of configuration you want to view.

Configuration document validation

Use these fields to specify the level of validation to perform on configuration documents.

Maximum
>
> Selecting **Maximum: Validate all documents** turns on validation for all documents, regardless of whether or not they are extracted, and regardless of the relationships between the documents.

High   Selecting **High: Validate extracted, parent, and local sibling documents** turns on validation for extracted documents and their parent documents, and turns on validation for the sibling documents of the documents which have been extracted. For example, if **High** validation is selected, and if the `server.xml` document is extracted, when performing validation, validation is performed on the three documents: `server.xml`, `node.xml`, and `cell.xml`. and on the two sibling documents `variables.xml` and `resources.xml` within the server1 directory.

Medium

Selecting **Medium: Validate extracted and parent documents** turns on validation for the documents which have been extracted by the user interface, and also turns on validation of the parent documents of the documents which have been extracted. For example, using the partial directory structure from above, if **Medium** validation is selected, and if the `server.xml` document is extracted, when performing validation, validation is performed on all three of the documents `server.xml`, `node.xml`, and `cell.xml`.

Low   Selecting **Low: Validate extracted documents** turns on validation for just those documents which have been extracted by the user interface.

None   Selecting **None: Do not validate documents** disables validation. No configuration documents are validated.

Enable Cross Validation

Enables cross validation of configuration documents. Enabling cross validation enables comparison of configuration documents for conflicting settings.

Configuration Problems

Displays current configuration problem error messages. Click a message for detailed information about the problem.

Scope

Sorts the configuration problem list by the configuration file where each error occurs. Click a message for detailed information about the problem.

Message

Displays the message returned from the validator.

Explanation

A brief explanation of the problem.

User action

Specifies the recommended action to correct the problem.

Target Object

Identifies the configuration object where the validation error occurred.

Severity

Indicates the severity of the configuration error. There are three possible values for severity.

Error   This means that there is a problem with the configuration that might cause partial or complete failure of server function. This is the most severe warning.

Warning

This means that there is a problem with the configuration that might cause a failure of server function, or that might cause the server to function in an unexpected manner.

Information

A setting of the configuration that is unexpected and noteworthy, which requires customer notification. Information is used when the configuration has a value which is probably okay, but should be double checked by the administrator. This is the least crucial level of severity.

Local URI

Specifies the local URI of the configuration file where the error occurred.

Full URI

Specifies the full URI of the configuration file where the error occurred.

Validator classname

The classname of the validator reporting the problem.

Runtime events

Use the Runtime event pages of the administrative console to view the events published by application server classes.

To view these administrative console pages, click **Troubleshooting**. Expand **Runtime Messages** and click either **Runtime Error**, **Runtime Warning**, or **Runtime Information**.

Separate pages show error events, warning events, and informational events. Each page displays events in the same format.

You can adjust the number of messages that appear on the page in the **Preferences** settings.

Click a message to view event details.

Timestamp
When the event occurred.

Message originator
Internal application server class that published the event.

Message
Identifier and short description of the event.

Message details

Use the Message Details panel of the administrative console to view detailed information about errors, warnings, and informational messages.

To view these administrative console pages, click **Troubleshooting**. Expand **Runtime Messages** and click either **Runtime Error**, **Runtime Warning**, or **Runtime Information**. Click a message to display this panel.

Each message has the following general property fields.

Message
The message ID and text.

Message type
Error, Warning, or Information.

Explanation
A description of the message.

User action
What you should do about the message.

Message originator

The name of the product class that originated the message.

Source object type

The name of the component that originated the message.

Timestamp

The date and time that the message originated.

Thread ID

The thread identifier.

Node name

The name of the node of the application server that originated the message.

Server name

The name of the application server process that originated the message.

Diagnostic Provider ID

The Diagnostic Provider ID of the component that originated the message. Click on Configuration Data, State Data, or Tests to run the corresponding diagnostic action against the originating component. A Diagnostic Provider ID will not be supplied with all messages.

Chapter 4. Administering application servers

An application server configuration provides settings that control how an application server provides services for running applications and their components.

About this task

After you install the product, you might have to perform one or more of the following tasks. Unless the task you want to perform is dependent on the existence of an application server, you can perform these tasks in any order.

Procedure

- Create an application server.
- Create clusters for workload balancing.
- Configure the server startup process such that only server components that are initially needed are started.

  When the server is configured such that only the components that are initially needed are started during the startup process, the remaining components are dynamically started as they are needed.

  gotcha: If you are running other WebSphere products on top of this product, make sure that those other products support this functionality before you select this property.
- Configure transport chains to handle client requests.
- Develop custom services.
- Define processes for the application server.

  As part of defining processes, you might want to:

  1. Define process execution statements for starting or initializing a UNIX process.
  2. Configure monitoring policies to track the performance of a process.
  3. Configure the process logs to which standard out and standard error streams write.
  4. Configure name-value pairs for properties.
- Configure the Java virtual machine.
- Optional: Run WebSphere Application Server for z/OS controllers and daemons in reusable address spaces whenever possible. For more ceptual information, see the documentation in this information center about the reusable address space.

Results

Any new application servers you create are displayed in the list of servers on the administrative console Application servers page.

What to do next

- Manage your application servers. Any newly created application servers are configured with many default settings that do not display when you run the Create New Application Server wizard. You might need to change some of these settings to better fit the needs of your environment.
- Deploy an application or component on the application server.
- View the status of the applications running on the application server.

Testing and production phases

Before explaining the test and production configurations for the product, you must understand which test phase should be done on the z/OS platform and which should be done on other platforms.

gotcha: Sharing resources between a production workload and a test workload can expose the production workload to a set of error conditions to which it is not exposed if the production and test workloads run in different cells. If possible, you should run production and test workloads in separate cells on your system.

Before setting up your test and production configurations for the product, you must understand which test phases should be done on the z/OS platform and which should be done on other platforms. The following sections explain the different phases:

- Unit test phase
- Component test phase
- Function test phase
- System test phase
- Production phase

Unit test phase

Applications that you plan on running in a z/OS environment should be developed on a distributed operating system, such as Windows or Linux Intel, on which the product is installed. These development environments contain assembly tools for web content delivery that are not available on z/OS. The IBM tooling solution assumes that you develop enterprise beans in one of these tools and perform basic testing of the business logic in the distributed environment before moving the application to the z/OS environment.

Component test phase

Component testing involves the joining together of several enterprise beans into logical components, providing them with access to data, and testing them together. While this can be done on a z/OS platform, it is recommended that you do this level of testing on a distributed platform. Performing this type of testing on a distributed platform enables a small team of developers to join the code pieces together and test the interactions. This type of testing focuses on the individual beans and their relationships to each other rather than z/OS platform functions and features.

Function test phase

Function testing involves joining the various components together, connecting them to test data in the target database, and validating the function that the application provides. Where this test is performed depends on the function and its data requirements. If the target deployment platform is z/OS, you might want to do this level of testing on z/OS. In this situation you should install the applications that you are testing on one or more servers that are only used for testing.

When you install the application on a test server, define where in the JNDI directory the references to the application are stored, and then configure the test clients such that they know the location of the test application. The test clients can then drive requests against the test server to perform the functional testing. You can use remote debugging tools to diagnose problems you encounter along the way.

System test phase

Before you put an application into production on z/OS, you should install the application into a system test environment on z/O and simulate a real workload on that application. When setting up your system test environment, you should define an additional test server on a cell that is dedicated to the test system, and install the application onto that server. When installed, enterprise beans that are part of the application should be registered in a different subtree of the JNDI directory. This normally happens by default but it is good to verify that this registration occurs. The test clients must be configured to the version of the application that is being tested before you run your tests.

Production phase

After you are satisfied with the functional and system testing, install the application in a cell that is used for production . The difference between a production cell and a test cell is whether the remote debugger is allowed to be attached. Normally, it is not acceptable for a production workload to stop because a remote debugging request is sent to the cell.

Test cells and production cells

If you require complete availability of your production system, this configuration eliminates the risk of including production and test in the same cell.

As the graphic below indicates, placing test and production servers into separate cells eliminates all local sharing between test and production and provides the highest risk reduction possible. .

Configuring virtual hosts

Virtual hosts let you manage a single application server on a single machine as if the application server were multiple application servers each on their own host machine. You can separate and control which resources are available for client requests by combining multiple host machines into a single virtual host, or by assigning host machines to different virtual hosts.

Before you begin

If your external HTTP server configuration uses the default port, 9080, you do not have to perform these steps.

About this task

Virtual hosts isolate and independently manage multiple sets of resources on the same physical machine. Resources associated with one virtual host cannot share data with resources associated with another virtual host. This is true even though the virtual hosts share the same application server on the same physical machine.

For example, suppose that:
- An Internet service provider (ISP) has two customers with Internet sites hosted on the same machine. The ISP keeps the two sites isolated from one another, despite their sharing a machine, by using virtual hosts. The ISP associates the resources of the first company with VirtualHost1 and the resources of the second company with VirtualHost2. Both virtual hosts map to the same application server.
- Both company sites offer the same servlet. Each site has its own instance of the servlet, and is unaware of the same servlet on the other site. If the company whose site is organized on VirtualHost2 is past due in paying its account with the ISP, the ISP can refuse all servlet requests that are routed to VirtualHost2. Even though the same servlet is available on VirtualHost1, the requests directed at VirtualHost2 do not go to the other virtual host.

Because the servlet is associated with a virtual host instead of the actual DNS address, The servlet on virtual host VirtualHost1 does not share its context with the servlet that has the same name on virtual host VirtualHost2. Requests for the servlet on VirtualHost1 can continue as usual, even though VirtualHost2 is refusing to fill requests for the servlet with the same name.

If any of the following conditions exist, you must update the HTTP port numbers associated with the default virtual host. or define a new virtual host and associate it with the ports your HTTP server configuration uses:
- Your external HTTP server configuration uses a port other than the default port of 9080, you must define the port that you are using.
- You are using the default HTTP port 9080, but the port is no longer defined. You must define port 9080.

- You have created multiple application servers as either stand-alone servers or cluster members, and these servers use the same virtual host. Because each server must be listening on a different port, you must define a virtual host alias for the HTTP port of each server.

If you define new virtual host aliases, identify the port values that the aliases use on the Host alias settings page in the administrative console.

Perform the following steps to create a new virtual host or change the configuration of an existing virtual host.

Procedure

1. In the administrative console, click **Environment > Virtual hosts**.
2. Optional: Create a new virtual host. If you create a new virtual host, a default set of 90 MIME entries are automatically created for that virtual host.
   a. In the administrative console, click **New**.
   b. Enter the name of the new virtual host and click **OK**. The new virtual host appears in the list of virtual hosts you can configure.
3. Select the virtual host whose configuration you want to change.
4. Under Additional Properties, click **Host aliases**.
5. Create new host aliases or update existing host aliases to associate each of your HTTP port numbers with this virtual host.

   There must be a virtual host alias corresponding to each port your HTTP server configuration uses. There is one HTTP port associated with each web container, and it is usually assigned to the virtual host named `default_host`. You can change the default assignment to any valid virtual host.

   The host aliases associated with the `default_host` virtual host are set to * when you install the product. The * (an asterisk) indicates that the alias name does not have to be specified or that any name can be specified.

   When the URL for the application is entered into a web browser, the port number is included. For example, if 9082 is the port number, the specified URL might look like the following:

   `http://localhost:9082/wlm/SimpleServlet`

   To create a new host alias:
   a. Click **New**.
   b. Specify a host alias name in the Host Name field and one of your HTTP ports in the Port field.

      You can specify * (an asterisk) for the alias name if you do not want to require the specification of the alias name or if you want to allow any name to be specified.
   c. Click **OK** and **Save** to save your configuration change.

   To update an existing host alias:
   a. Select an existing host alias name.
   b. Change the value specified in the Port field to one of your HTTP ports.
   c. Click **OK** and **Save** to save your configuration change.

6. Optional: Define a MIME object type and its file name extension if you require a MIME type other than the pre-defined types.
   a. For each needed MIME entry on the MIME type collection page, click **New**.
   b. On the MIME type settings page, specify a MIME type and extension.
   c. Click **OK** and **Save** to save your configuration change.

7. Regenerate the web server plug-in configuration.
   a. **Servers > Server Types > Web servers**, then select the appropriate web server.
   b. Click **Generate plug-in**, then click **Propagate plug-in**.

8. Restart the application server.

Virtual hosts

A virtual host is a configuration entity that enables a single host machine to resemble multiple host machines. It maintains a list of Multipurpose Internet Mail Extensions (MIME) types that it processes. You can associate a virtual host to one or more Web modules, but you can associate each web module with one and only one virtual host. Resources associated with one virtual host cannot share data with resources associated with another virtual host, even if the virtual hosts share the same physical machine.

Each virtual host has a logical name and a list of one or more DNS aliases by which it is known. A DNS alias is the TCP/IP hostname and port number that is used to request the servlet, for example `yourHostName:80`. When no port number is specified, 80 is assumed.

The virtual host configuration uses wildcard entries with the ports for its virtual host entries.
- The default alias is *:80, using an external port that is not secure.
- Aliases of the form *:9080 use the internal port that is not secure.
- Aliases of the form *:9443 use the secure internal port.
- Aliases of the form *:443 use the secure external port.

A client request for a servlet, JavaServer Pages file, or related resource contains a DNS alias and a Uniform Resource Indicator (URI) that is unique to that resource. When a client request for a servlet, JavaServer Pages file, or related resource is received, the DNS alias is compared to the list of all known virtual host groups to locate the correct virtual host, and the URI is compared to the list of all known URI groups to locate the correct URI group. If the virtual host group and URI group are found, the request is sent to the corresponding server group for processing and a response is returned to browser. If a matching virtual host group or URI group is not found, an error is returned to the browser.

A virtual host is not associated with a particular node (machine). It is a configuration, rather than a live object, which is why you can create it, but cannot start or stop it. A default virtual host, named default_host, is automatically configured the first time you start an application server. Unless you specifically want to isolate resources from one another on the same node (physical machine), you probably do not need any virtual hosts in addition to the default host.

The DNS aliases for the default virtual host are configured as `*:80` and `*:9080`, where port 80 is the HTTP server port and port 9080 is the port for the default server's HTTP transport. The default virtual host includes common aliases, such as the machine's IP address, short host name, and fully qualified host name. One of these aliases comprises the first part of the path for accessing a resource such as a servlet. For example, the alias `localhost:80` is used in the request `http://localhost:80/myServlet`.

Adding a localhost to the virtual hosts adds the host name and IP address of the localhost machine to the alias table. This allows a remote user to access the administrative console.

You can use the administrative console to add or change DNS aliases if you want to use ports other than the default ports. If you do make a change to a DNS alias, you must regenerate the web server plug-in configuration. You can use the administrative console to initiate the plug-in regeneration.

Note: You might want to add additional aliases or change the default aliases if:
- The HTTP server instance is running on a port other than 80. Add the correct port number to each of the aliases. For example, change `yourhost` to `yourhost:8000`.
- You want to make HTTPS requests, which use Secure Sockets Layer (SSL). To make HTTPS requests you must add port 443 to each of the aliases. Port 443 is the default port for SSL requests.
- Your web server instance is listening for SSL requests on a port other than 443. In this situation, you must add that port number to each of the aliases.
- You want to use a port other then default port (9080) for the application server.
- You want to use other aliases that are not listed.

When you request a resource, the product tries to map the request to an alias of a defined virtual host. The `http://host:port/` portion of the virtual host is not case sensitive, but the URL that follows is case sensitive. The match for the URL must be alphanumerically exact. Different port numbers are treated as different aliases.

For example, the request `http://www.myhost.com/myservlet` maps successfully to `http://WWW.MYHOST.COM/myservlet` but not to `http://WWW.MYHOST.COM/MYSERVLET` or `Www.Myhost.Com/Myservlet`. In the latter two cases, these mappings fail because of case sensitivity. The request `http://www.myhost.com/myservlet` does not map successfully to `http://myhost/myservlet` or to `http://myhost:9876/myservlet`. These mappings fail because they are not alphanumerically correct.

You can use wildcard entries for aliases by port and specify that all valid host name and address combinations on a particular port map to a particular virtual host.

If you request a resource using an alias that cannot be mapped to an alias of a defined virtual host, you receive a 404 error in the browser that you used to issue the request. A message states that the virtual host could not be found.

Two sets of associations occur for virtual hosts. Application deployment associates an application with a virtual host. Virtual host definitions associate the network address of the machine and the HTTP transport or web server port assignment of the application server with the virtual host. Looking at the flow from the web client request for the snoop servlet, for example, the following actions occur:

1. The web client asks for the snoop servlet: at web address `http://www.some_host.some_company.com:9080/snoop`
2. The some_host machine has the 9080 port assigned to the stand-alone application server, server1.
3. server1 looks at the virtual host assignments to determine the virtual host that is assigned to the alias `some_host.some_company.com:9080`.
4. The application server finds that no explicit alias for that DNS string exists. However, a wild card assignment for host name * at port 9080 does exist. This is a match. The virtual host that defines the match is default_host.
5. The application server looks at the applications deployed on the default_host and finds the snoop servlet.
6. The application server serves the application to the web client and the requester is able to use the snoop servlet.

Table 33. Aliases for a vitual host. You can have any number of aliases for a virtual host. You can even have overlapping aliases, such as:

Virtual host	Alias	Port
default_host	*	9080
	localhost	9080
	my_machine	9080
	my_machine.my_company.com	9080
	localhost	80

The Application Server looks for a match using the explicit address specified on the web client address. However, it might resolve the match to any other alias that matches the pattern before matching the explicit address. Simply defining an alias first in the list of aliases does not guarantee the search order whenever the product is looking for a matching alias.

A problem can occur if you use the same alias for two different virtual hosts. For example, assume that you installed the default application and the snoop servlet on the default_host. You also have another virtual host called the admin_host. However, you have not installed the default application or the snoop servlet on the admin_host.

Table 34. Virtual hosts with overlapping aliases. Assume that you define overlapping aliases for both virtual hosts because you accidentally defined port 9080 for the admin_host instead of port 9060:

Virtual host	Alias	Port
default_host	*	9080
	localhost	9080
admin_host	*	9060
	my_machine.com	9080

Assume that a web client request comes in for `http://my_machine.com:9080/snoop.`

If the application server matches the request against `*:9080`, the application is served from the default_host. If the application server matches the request to `my.machine.com:9080`, the application cannot be found. A 404 error occurs in the browser that issues the request. A message states that the virtual host could not be found.

This problem is the result of not finding the requested application in the first virtual host that has a matching alias. The correct way to code aliases is for the alias name on an incoming request to match only one virtual host in all of your virtual host definitions. If the URL can match more than one virtual host, you can see the problem just described.

Virtual host collection

Use this page to create and manage configurations that each let a single host machine resemble multiple host machines. Such configurations are known as *virtual hosts.*

To view this administrative console page, click **Environment > Virtual hosts**.

Each virtual host has a logical name (which you define on this panel) and is known by its list of one or more domain name system (DNS) aliases. A DNS alias is the TCP/IP host name and port number used to request the servlet, for example yourHostName:80. (Port 80 is the default.)

You define one or more alias associations by clicking an existing virtual host or by adding a new virtual host.

When a servlet request is made, the server name and port number entered into the browser are compared to a list of all known aliases in an effort to locate the correct virtual host to serve the servlet. No match returns an error to the browser.

An application server profile provides a default virtual host with some common aliases, such as the internet protocol (IP) address, the DNS short host name, and the DNS fully qualified host name. The alias comprises the first part of the path for accessing a resource such as a servlet.

For example, the alias is localhost:80 in the request http://localhost:80/myServlet.

A virtual host is not associated with a particular profile or node (machine), but is associated with a particular server instead. It is a configuration, rather than a "live object." You can create a virtual host, but you cannot start or stop it.

For many users, creating virtual hosts is unnecessary because the default_host that is provided is sufficient.

Adding the host name and IP address of the localhost machine to the alias table lets a remote user access the administrative console.

Resources associated with one virtual host cannot share data with resources associated with another virtual host, even if the virtual hosts share the same physical machine.

Name

Specifies a logical name for configuring web applications to a particular host name. The default virtual host is suitable for most simple configurations.

Virtual hosts enable you to isolate, and independently manage, multiple sets of resources on the same physical machine. Determine whether you need a virtual host alias for each port associated with an HTTP transport channel or an HTTP transport. There must be a virtual host alias corresponding to each port used by an HTTP transport channel or an HTTP transport. There is one HTTP transport channel or HTTP transport associated with each web container, and there is one Web container in each application server.

When you create a virtual host, a default set of 90 MIME entries is created for the virtual host.

You must create a virtual host for each HTTP port in the following cases:
- You use the internal HTTP transport with a port other than the default value of 9080, or for some reason the virtual host does not contain the usual entry for port 9080.
- You create multiple application servers, such as stand-alone servers, managed servers, or cluster members, that are using the same virtual host. Because each server must be listening on a different HTTP port, you need a virtual host alias for the HTTP port of each server.
- You use a Web server that listens on a port not already specified in your virtual host aliases.

Virtual host settings

Use this page to configure a virtual host instance.

To view this administrative console page, click **Environment > Virtual hosts >** *virtual_host_name*.

Name:

Specifies a logical name for configuring web applications to a particular host name. The default virtual host is suitable for most simple configurations.

Data type	String
Default	default_host

Host alias collection

Use this page to manage host name aliases defined for a virtual host. An alias is the DNS host name and port number that a client uses to form the URL request for a web application resource.

To view this administrative console page, click **Environment > Virtual hosts >** *virtual_host_name* **> Host aliases**.

Host name:

Specifies the IP address, DNS host name with domain name suffix, or just the DNS host name, used by a client to request a web application resource (such as a servlet, JavaServer Pages (JSP) file, or HTML page). For example, the host alias name is `myhost` in a DNS name of `myhost:8080`.

The product provides a default virtual host (named default_host). The virtual host configuration uses the wildcard character * (asterisk) along with the port number for its virtual host entries. Unless you specifically want to isolate resources from one another on the same node (physical machine), you probably do not need any virtual hosts in addition to the default host.

Port:

Specifies the port for which the web server has been configured to accept client requests. For example, the port assignment is *8080* in a DNS name of `myhost:8080`. A URL refers to this DNS as: http://*myhost:8080*/servlet/snoop.

Host alias settings:

Use this page to view and configure a host alias.

To view this administrative console page, click **Environment > Virtual hosts >** *virtual_host_name* **> Host aliases >** *host_alias_name*.

Host name:

Specifies the IP address, domain name system (DNS) host name with domain name suffix, or the DNS host name that clients use to request a Web application resource, such as a servlet, JSP file, or HTML page.

For example, when the DNS name is `myhost`, the host alias is `myhost:8080`, where *8080* is the port. A URL request can refer to the snoop servlet on the host alias as: http://*myhost:8080*/servlet/snoop.

When there is no port number specified for a host alias, the default port is 80. For existing virtual hosts, the default host name and port reflect the values specified at product installation or configuration. For new virtual hosts, the default can be * to allow any value or no specification.

Data type	String
Default	*
	You can also use the IP address or the long or short DNS name.

Port:

Specifies the port where the web server accepts client requests. Specify a port value in conjunction with the host name.

The port value can be any integer between 0 and 65535. You can also specify an asterisk (the wildcard value) if you want the system to select a free port for you.

The default reflects the value specified at product setup. The default might be `80`, `81`, `9060` or a similar value.

Data type	Integer
Range	0 - 65535 or the wildcard value (*)
Default	9060

MIME type collection

Use this page to view and configure multi-purpose internet mail extensions (MIME) object types and their file name extensions.

The list shows a collection of MIME type extension mappings defined for the virtual host. Virtual host MIME entries apply when you do not specify MIME entries at the web module level.

To view a list of current virtual host Mime types in the administrative console, click **Environment > Virtual hosts >** *virtual_host_name* **> Mime types**.

MIME type:

Specifies a MIME type, which can be application, audio, image, text, video, www, or x-world. An example value for MIME type is `text/html`.

Extensions:

Specifies file extensions of files that map the MIME type. Do not specify the period before the extension. Example extensions for a `text/html` MIME type are `htm` and `html`.

MIME type settings:

Use this page to configure a multi-purpose internet mail extensions (MIME) object type.

To view this administrative console page, click **Environment > Virtual hosts >** *virtual_host_name* **> MIME types >** *mime_type*.

MIME type:

Specifies a MIME type, which can be application, audio, image, text, video, www, or x-world. An example value for MIME type is `text/html`.

An example value for MIME type is `text/html`. A default value appears only if you are viewing the configuration for an existing instance.

Data type	String

Extensions:

Specifies file extensions of files that map the MIME type. Do not specify the period before the extension. Example extensions for a `text/html` MIME type are `htm` and `html`.

File extensions for a `text/html` MIME type are `.htm` and `.html`. A default value appears only if you are viewing the configuration for an existing MIME type.

Data type	String

Creating, editing, and deleting WebSphere variables

You can use WebSphere variables to provide settings for any of the string data type attributes that are contained in the product configuration files.

Before you begin

Because applications cannot directly access WebSphere variables, if you define a WebSphere variable inside of an application, an error message, such as "Unknown variable," is returned. If you must reference a WebSphere variable from within an application, include the following method in the application to expand the string that uses the WebSphere variable:

```
private String expandVariable(String s) throws
javax.management.JMException {
com.ibm.websphere.management.AdminService as =
com.ibm.websphere.management.AdminServiceFactory.getAdminService
();

String server = as.getProcessName();

java.util.Set result = as.queryNames(new javax.management.ObjectName("*:*,type=AdminOperations,process="
+ server), null);

return (String)as.invoke((javax.management.ObjectName)
result.iterator().next(),"expandVariable",new Object[]
{"${"+s+"}"}, new String[] {"java.lang.String"});
```

Similarly, you can include the following lines of code in a script file if you want to use a script command to expand WebSphere variables.

- Using Jacl:

```
set mbean [$AdminControl completeObjectName WebSphere:*,type=AdminOperations]
$AdminControl invoke $mbean expandVariable {{"${APP_INSTALL_ROOT}"}}
```

- Using Jython:

```
AdminOperations = AdminControl.completeObjectName('WebSphere:*,type=AdminOperations')
print AdminControl.invoke(AdminOperations, 'expandVariable', '${APP_INSTALL_ROOT}')
```

About this task

WebSphere variables are usually used to specify file paths. The "Variable settings" topic supplies further details about specifying variables and highlights further details about product components that use them.

WebSphere variables are also used to configure:

- Product path names, such as JAVA_HOME, and APP_INSTALL_ROOT.
- Configure certain cell-wide or cluster-wide customization values.
- The location service.
- Environment variables.

The variable scoping mechanism for WebSphere variables enables you to define a variable at the node, cluster, or cell level, as well as at the server level. This mechanism enables you to specify a setting for all of the servers in a node, cluster, or cell, instead of individually specifying the setting for each server.

To define a new variable, change the value of an existing variable, or delete an existing variable complete the following steps, as appropriate.

Procedure

1. Click **Environment > WebSphere variables** in the administrative console
2. Select the scope of the variable from the list of available scopes.

   If you create a new variable, it will be created at the selected scope. If you define the same variable at multiple levels, the more granular definition overrides the higher level setting. For example, if you specify the same variable on a cell level and at a node level, the node level setting overrides the cell level setting.

   Scoping variables is particularly important if you are testing data source objects. Variable scoping can cause a data source to fail the test connection, but to succeed at run time, or to pass the test connection, but fail at run time.
3. Create a new variable.
   a. Click **New**.
   b. Specify a name, a value, and, optionally, a description for the variable.

The application server uses WebSphere Application Server internal variables for its own purposes. The prefixes that indicate that a variable is internal are `WAS_DAEMON_`*<server custom property>*, `WAS_DAEMON_ONLY_`*<server custom property>*, and `WAS_SERVER_ONLY_`*<server custom property>*. Any variables with these tags are not intended for your use. They are reserved exclusively for use by the server run time. Modifying these variables can cause unexpected errors.

You can use WebSphere variables to modify the daemon configuration. By appending a server custom property onto a daemon tag, you can designate that variable specifically for that daemon. Enter `DAEMON_`*<server custom property>* in the **Name** field. For example, if you enter `DAEMON_ras_trace_outputlocation` in the Name field and `SYSOUT` in the Value field, you can direct that particular daemon's trace output to SYSPRINT.

You can create WebSphere variables that support substitution. For example, if you enter `${`*<variable name>*`}` in the **Name** field, the value of *<variable name>* becomes the name of your new WebSphere variable. For example if you enter `${JAVA_HOME}` as the name of your variable, the name of the WebSphere variable that is created is the Java home directory.

   c. Click **OK**.

   d. Click **Environment** > **WebSphere variables** in the administrative console navigation, and verify that the variable is displayed in the list of variables for the selected scope.

      The administrative console does not pick up typing errors. The variable is ignored if it is referred to incorrectly.

4. Modify the setting for an existing variable.

   a. Click on the name of the variable that you want to change.

   b. Modify the content of the Values field.

      The Values field for some of the variables that are already defined when you install the product are read-only because changing the values that are specified for those variables might cause product processing errors.

   c. Click **OK**.

5. Delete an existing variable.

   a. Select the variable that you want to delete.

   b. Click **Delete**.

   c. Click **OK**.

   d. Verify that this variable was removed from the list of variables for the selected scope.

6. Save your configuration.

7. Stop the affected servers and start those servers again to put the variable configuration change into effect.

   If the change you made affects a node, you must stop and restart all of the servers on that node. Similarly if the change you made affects a cell, you must stop and restart all of the servers in that cell.

WebSphere variables collection

Use this page to view and change the defined product variables with their values. You can also use this page to create a new variable, or delete an existing variable. These variables are name and value pairs that are used to provide the settings for the string data type configuration attributes that are contained in one of the XML formatted configuration files that reside in the product repository.

To view this administrative console page, click **Environment > WebSphere variables**.

To display a list of all of the variables that are defined for a specific scope, select that scope.

To view additional information about a specific variable, or to change the setting for that variable, click the variable name. Some of the pre-defined variables, that is, variables that already exist when you install the product, are set at values that are required for the product to function properly. The Value fields for these variables are read-only and cannot be edited.

To define a new variable, select the appropriate scope from the list of available options and then click **New**. The selected scope indicates the level at which the variable setting is visible.

To delete an existing variable, select the appropriate variable, and then click **Delete**. Do not delete any of the pre-defined variables. Before deleting a variable that you defined, make sure that none of your applications require the configuration attribute setting that the variable provides.

Name

Specifies the symbolic name for a WebSphere Application Server variable. For example, a variable name might represent a physical path or URL root used by WebSphere Application Server.

Value

Specifies the value that the symbolic name represents. For example, the value might be an absolute path value for a file or URL root.

Scope

Specifies the level at which a WebSphere variable is visible on the administrative console panel. The scope is specified when a new variable is defined.

A resource can be visible in the administrative console collection table at the node or server scope.

On a multiple-server product, a resource also can be visible at the cell or cluster scope. The cluster scope is only available if a cluster is defined for the cell.

WebSphere variables settings

Use this page to define the name and value of a WebSphere variable. A WebSphere variable is a name and value pair that is used to provide the setting for one of the string data type attributes contained in one of the XML formatted configuration files that reside in the product repository.

To view this administrative console page, click **Environment > WebSphere variables > ** *WebSphere_variable_name*.

Name:

Specifies the symbolic name for a product variable. After the variable is defined, this symbolic name can be specified in the **Value** field of any other product configuration field that accepts a string value. Whenever the application server encounters a configuration field that contains one or more symbolic names, it replaces the symbolic names with their defined values. For example, you might define a variable name that represent a commonly used file path or URL.

WebSphere Application Server variables are used for:
- Configuring WebSphere Application Server path names, such as *JAVA_HOME*, and APP_INSTALL_ROOT.
- Configuring certain cell-wide customization values.
- Configuring the z/OS location service for the product.

For example, WAS_SERVER_NAME is the pre-defined symbolic name of the variable that represents the name of the default application server that is provided with the product..

Value:

Specifies the value that the symbolic name represents.

For example, server1 is the value of a pre-defined variable WAS_SERVER_NAME.

| **Data type** | String |

Description:

Documents the purpose of a variable.

Data type   String

Introduction: Variables

Variables come in many varieties. They are used to control settings and properties relating to the server environment. The three main types of variables that you should understand are environment variables, WebSphere variables, and custom properties.

Environment variables

Environment variables, also called *native environment variables*, are not specific to WebSphere Application Server and are defined by other elements, such as UNIX, Language Environment® (LE), or third-party vendors, among others. Some of the UNIX-specific native variables are LIBPATH and STEPLIB. These variables tend to be operating system-specific.

Environment variables can also be specified as a servant custom property. To specify an environment variable as a servant custom property, in the administrative console, click **Servers > Server Types > WebSphere application servers***server_name*. Then, under Server Infrastructure, click **Java process management > Process definition**, select either **Control**, **Servant**, or **Adjunct**, and then click **Environment entries**. This path is also used to set environment variables that control the collection of application server and Web container information in z/OS System Management Facility (SMF) records.

WebSphere variables

WebSphere variables are name and value pairs that are used to provide settings for any of the string data type attributes contained in one of the XML formatted configuration files that reside in the product repository. After a variable is defined, the value specified for the variable replaces the variable name whenever the variable name is encountered during configuration processing.

WebSphere variables can be used to configure:
- WebSphere Application Server path names, such as `JAVA_HOME`, and `APP_INSTALL_ROOT`
- A path value for the extendedDocumentRoot JSP or file serving attribute. This capability enables you to add an application to each node in a clustered environment without modifying the ibm-web-ext.xmi file for that application on each node.

> **Note:** For IBM extension and binding files, the .xmi or .xml file name extension is different depending on whether you are using a pre-Java EE 5 application or module or a Java EE 5 or later application or module. An IBM extension or binding file is named ibm-*-ext.xmi or ibm-*-bnd.xmi where * is the type of extension or binding file such as app, application, ejb-jar, or web. The following conditions apply:
> - For an application or module that uses a Java EE version prior to version 5, the file extension must be .xmi.
> - For an application or module that uses Java EE 5 or later, the file extension must be .xml. If .xmi files are included with the application or module, the product ignores the .xmi files.
>
> However, a Java EE 5 or later module can exist within an application that includes pre-Java EE 5 files and uses the .xmi file name extension.
>
> The `ibm-webservices-ext.xmi`, `ibm-webservices-bnd.xmi`, `ibm-webservicesclient-bnd.xmi`, `ibm-webservicesclient-ext.xmi`, and `ibm-portlet-ext.xmi` files continue to use the .xmi file extensions.

- Certain cell-wide customization values
- The location service for the z/OS platform.

To create or modify a WebSphere variable, in the administrative console click **Environment > WebSphere variables**.

A variable can apply to a cell, a cluster, a node, or a server.

How the variable is set determines its scope. If the variable is set:
- At the server level, it applies to the entire server.
- At the node level, it applies to all servers in the node, unless you set the same variable at the server level. In that case, for that server, the setting that is specified at the server level overrides the setting that is specified at the node level.
- At the cell level, it applies to all nodes in that cell, unless you set the same variable at the node or server level.
  - If you set the same variable at the server level, for that server, the setting that is specified at the server level overrides the setting that is specified at the cell level.
  - If you set the same variable at the node level, for all servers in that node, the setting that is specified at the node level overrides the setting that is specified at the cell level.

Custom properties

Custom properties are property settings meant for a specific functional component. Any configuration element can have a custom property. Common configuration elements are cell, node, server, web container, and transaction service. A limited number of supported custom properties are available and these properties can be set in the administrative console using the custom properties link that is associated with the functional component.

For example, to set web container custom properties, click **Servers > Server Types > WebSphere application servers >** *server_name*, and then, in the Container settings section, click **Web container > Custom properties**

Custom properties set from the web container custom properties page apply to all transports that are associated with that web container; custom properties set from one of the web container transport chain or HTTP transport custom properties pages apply only to that specific HTTP transport chain or HTTP transport. If the same property is set on both the web container page and either a transport chain or HTTP transport page, the settings on the transport chain or HTTP transport page override the settings that are defined for the web container for that specific transport.

WebSphere variables

WebSphere variables are name and value pairs that are used to provide settings for any of the string data type attributes that are used to configure the product. After a variable is defined, the symbolic name that is specified for that variable can be specified in the **Value** field of any other configuration field for the product that accepts a string value.

WebSphere variables can be used to configure:
- WebSphere Application Server path names, such as `JAVA_HOME`, and `APP_INSTALL_ROOT`
- A path value for the extendedDocumentRoot JSP or file serving attribute. This capability enables you to add an application to each node in a clustered environment without modifying the ibm-web-ext.xmi file for that application on each node.

    Note: For IBM extension and binding files, the .xmi or .xml file name extension is different depending on whether you are using a pre-Java EE 5 application or module or a Java EE 5 or later

application or module. An IBM extension or binding file is named ibm-*-ext.xmi or ibm-*-bnd.xmi where * is the type of extension or binding file such as app, application, ejb-jar, or web. The following conditions apply:

- For an application or module that uses a Java EE version prior to version 5, the file extension must be .xmi.
- For an application or module that uses Java EE 5 or later, the file extension must be .xml. If .xmi files are included with the application or module, the product ignores the .xmi files.

However, a Java EE 5 or later module can exist within an application that includes pre-Java EE 5 files and uses the .xmi file name extension.

The `ibm-webservices-ext.xmi`, `ibm-webservices-bnd.xmi`, `ibm-webservicesclient-bnd.xmi`, `ibm-webservicesclient-ext.xmi`, and `ibm-portlet-ext.xmi` files continue to use the .xmi file extensions.

- Certain cell-wide customization values
- The location service for the z/OS platform.

When a variable is defined, it is given a scope. The scope is the range of locations within the product network where the variable is applicable.

- A variable with a cell-wide scope is available across the entire deployment manager cell.
- A variable with a cluster-wide scope is available across the entire cluster in the cell.
- A variable with a node-level scope is available only on the node and the servers on that node. If a node-level variable has the same name as a cell-wide variable, the node-level variable value takes precedence.
- A server variable is available only on the one server process. A server variable takes precedence over a variable with the same name that is defined at a higher level.

The value of a configuration attribute can contain references to one or more variables. The syntax for such an attribute is the name of the variable, enclosed in either a pair of curly braces { } or a pair of parenthesis ( ). In either case, the variable is proceeded by the dollar sign.

A string configuration attribute value can consist of:

- String literals, including the null value and an empty string
- Variable references that each includes one or more levels of indirection
- Nested variable references.
- Any combination of non-null and non-empty string literals, variable references, and nested variable references.

Table 35. WebSphere variables and attributes. The following table illustrates all of the possible combinations.

Configuration attribute consists of:	Configuration attribute value	Variable name	Second variable value	Third variable value	Fourth variable value	Expanded configuration attribute value
String literal	/IBM/ WebSphere/ AppServer	N/A	N/A	N/A	N/A	/IBM/ WebSphere/ AppServer
Variable reference	$(WAS_ INSTALL_ ROOT)	WAS_ INSTALL_ ROOT	/IBM/ WebSphere/ AppServer	N/A	N/A	/IBM/ WebSphere/ AppServer
Variable reference with a string literal	$(USER_ INSTALL_ ROOT)/temp	USER_ INSTALL_ ROOT	N/A	N/A	/IBM/ WebSphere/ AppServer/ profiles/ AppSrv01	/IBM/ WebSphere/ AppServer/ profiles/ AppSrv01/temp

Table 35. WebSphere variables and attributes (continued). The following table illustrates all of the possible combinations.

Configuration attribute consists of:	Configuration attribute value	Variable name	Second variable value	Third variable value	Fourth variable value	Expanded configuration attribute value
Indirect variable reference with a string literal	$(WAS_ INSTALL_ ROOT)/lib	WAS_ INSTALL_ ROOT	$(MY_INSTALL_ ROOT)	MY_INSTALL_ ROOT	N/A	N/A
Nested variable references with string literal (Example 1)	$(${INSTALL_ TYPE}_ INSTALL_ ROOT)/lib	INSTALL_ TYPE	USER	USER_INSTALL_ ROOT	/IBM/ WebSphere/ AppServer/ profiles/ AppSrv01	/IBM/ WebSphere/ AppServer/ profiles/ AppSrv01/lib
Nested variable references with string literal (Example 2)	$(${INSTALL_ TYPE}_ INSTALL_ ROOT)/lib	INSTALL_ TYPE	WAS	WAS_INSTALL_ ROOT	/IBM/ WebSphere/ AppServer/ AppServer	/IBM/ WebSphere/ AppServer/ AppServer/lib

During the configuration process, whenever a variable is encountered as the value for a configuration attribute, a variable expansion is performed on that variable. A variable expansion is the process of recursively replacing variable references with variable values until only a string literal remains as the value for the configuration attribute. If the expansion process encounters a variable that is not properly defined, the expansion of that variable stops and a VariableExpansionException exception is issued. The product configuration process continues. However, processing errors might occur because the value for this configuration attribute is not properly established.

gotcha: The variable expansion syntax that is provided in Versions 6.0.x, and 6.1.x, of the product, includes a variant that consists of a dollar sign, and a single letter variable name without any surrounding braces or parenthesis. This syntax is not supported in Version 8.0 or higher. All WebSphere variables references must be surrounded by matching parenthesis or braces, even if it is a single letter. That syntax required escaping of dollar signs to avoid ambiguity.

Table 36. Literal dollar sign. For backward compatibility, the escaping of the literal dollar sign is still supported, and the literal dollar sign is interpreted as indicated in the following table.

Input value	Value after expansion
$	$
$$	$
$$$	$$
$$$$	$$
$$$$$	$$$

Configuring the IBM Toolbox for Java

The IBM Toolbox for Java is a library of Java classes that are optimized for accessing IBM i data and resources. You can use the IBM Toolbox for Java JDBC driver to access local or remote DB2® UDB for IBM i databases from server-side and client Java applications that run on any platform that supports Java.

Before you begin

Determine which version of the IBM Toolbox for Java you want to use on your system.

About this task

The IBM Toolbox for Java is available in these versions:

IBM Toolbox for Java licensed program

The licensed program is available with every IBM i release. You can install the licensed program on your IBM i system, and then either copy the IBM Toolbox for Java JAR file (*jt400.jar*) to your system or update your system *classpath* to locate the server installation. Product documentation for IBM Toolbox for Java is available from the IBM i Information Center: http://publib.boulder.ibm.com/eserver/ibmi.html Locate the documentation by traversing the following path in the left-hand navigation window of the iSeries® information center:

Programming > **Java** > **IBM Toolbox for Java**.

JTOpen

JTOpen is the open source version of IBM Toolbox for Java, and is more frequently updated than the licensed program version. You can download JTOpen from http://www.ibm.com/servers/eserver/iseries/toolbox/downloads.htm. You can also download the *JTOpen Programming Guide*. The guide includes instructions for installing JTOpen and information about the JDBC driver.

The IBM Toolbox for Java JDBC driver is included with both versions of the IBM Toolbox for Java. This JDBC driver supports JDBC 3.0. For more information about IBM Toolbox for Java and JTOpen, see the product website at http://www.ibm.com/servers/eserver/iseries/toolbox/index.html.

gotcha: If you are using the product on platforms other than iSeries, use the **JTOpen** version of the Toolbox JDBC driver.

Procedure

1. Download the *jt400.jar* file from the **JTOpen** URL at http://www.ibm.com/servers/eserver/iseries/toolbox/downloads.htm.

   Place it in a directory on your workstation such as `/JDBC_Drivers/Toolbox`.
2. Open the administrative console.
3. Select **Environment > WebSphere variables**.
4. In the list of available scopes, select the appropriate node.
5. Locate the WebSphere variable OS400_TOOLBOX_JDBC_DRIVER_PATH in the list of variables that are defined for that scope.

   Depending on how many variables are defined for the selected node, you might have to navigate through multiple pages of variables to find the OS400_TOOLBOX_JDBC_DRIVER_PATH variable. In this situation, clicking the arrow at the bottom of the page takes you to the next page of variables for the selected node.
6. Click **OS400_TOOLBOX_JDBC_DRIVER_PATH** in the name column.
7. Set the value to the full directory path to the `jt400.jar` file downloaded in step one. Do not include jt400.jar in this value.

   For example, if the fully qualified path to the `jt400.jar` file is:

   `JDBC_Drivers/Toolbox/jt400.jar`

   Specify JDBC_Drivers/Toolbox as the value for the OS400_TOOLBOX_JDBC_DRIVER_PATH variable.
8. Click **Apply** and then click **Save** to save your changes.

Repository service custom properties

Use this page to add custom properties for the repository service.

You can specify repository service custom properties in the administrative console:

1. In the administrative console navigation, click **System Administration > Node agents**.
2. Select a node agent from the list.
3. Under Additional Properties, click **File synchronization service**.
4. Under Additional Properties, click **Custom properties**.

5. Click **New**.
6. Enter the name of the custom property in the Name field, and the value in the Value field. You can leave the Description field blank.

You can use the custom properties page to define the following repository service custom properties:
*   "recoveryNode"

recoveryNode
Specifies that a node is a recovery node. This property is only supported for a z/OS environment.

Set this value to `true` if you want a node in a Network Deployment cell to act as a peer restart and recovery node for another node in the same cell. The recovery node shadows the complete configuration of its recovery peer. Use this property if you need to support peer restart and recovery only and are not using a shared file system.

Data type	Boolean

Application server custom properties for z/OS
Some of the application server custom properties that are provided with the product can only be used with z/OS. This topic describes how to use these properties.

depfeat: Setting these custom properties at the server level is deprecated. However, you can specify them as WebSphere variables with a scope of either a specific server, specific node, or specific cell. Server scoped WebSphere variables still override any settings specified at the node scope, or higher, and are added to the was.env file.

To set one of these custom properties for either an application server or a deployment manager, in the administrative console, click **Environment** > **WebSphere variables**, select the appropriate node or cell from the list of available servers, nodes and cells, and then click **New**.

You can use the custom properties page to define the following application server custom properties for z/OS:
*   "adjunct_jvm_direct_options" on page 254
*   "allow_large_SAF_groups" on page 254
*   "com.ibm.ws.sib.ra.inbound.impl.MessageLockExpiry" on page 254
*   "control_region_confirm_recovery_on_no_srs" on page 255
*   "control_region_dreg_on_no_srs" on page 255
*   "control_region_http_queue_timeout_percent" on page 255
*   "control_region_https_queue_timeout_percent" on page 256
*   "control_region_iiop_queue_timeout_percent" on page 256
*   "control_region_mdb_queue_timeout_percent" on page 256
*   "control_region_mdb_request_timeout" on page 256
*   "control_region_sip_queue_timeout_percent" on page 257
*   "control_region_sips_queue_timeout_percent" on page 257
*   "control_region_timeout_delay" on page 257
*   "control_region_timeout_dump_action" on page 258
*   "control_region_timeout_dump_action_session" on page 258
*   "control_region_timeout_save_last_servant" on page 258
*   "controller_jvm_direct_options" on page 259
*   "Daemon_ras_trace_ctraceParms" on page 259

- "WAS_DAEMON_ONLY_enable_adapter" on page 282
- "wlm_enclavecreate_exstartdefer" on page 283
- "wlm_ae_spreadmin" on page 283
- "wlm_classification_file" on page 283
- "wlm_servant_start_parallel" on page 284
- "wlm_stateful_session_placement_on" on page 284
- "wsadmin_dumpthreads_enable_heapdump" on page 284
- "wsadmin_dumpthreads_enable_javatdump" on page 284

adjunct_jvm_direct_options

Specifies options that you need to pass directly to the Java virtual machine (JVM) launch in the adjunct. This property is typically used for JVM options that the JVM cannot read from the options file that is specified as the value of the control_region_jvm_properties_file property. For example, the JVM cannot read the value that is specified for the -memorycheck option in the options file.

If you specify multiple options, use a semicolon to separate the options.

You can use the servant_jvm_direct_options and control_jvm_direct_options custom properties to specify options that you need to pass directly to the JVM launch in the servant and controller, respectively.

Data Type	String
Default	Empty string
Used by Daemon	No

allow_large_SAF_groups

Specifies that you want to allow the application server to do lookups on large SAF groups.

If you set this property to 1, the size of the buffer that is used to do lookups is tripled from 8192 bytes to 24576 bytes.

You can also set this property to a specific number off bytes up to and including 2147483647. If you specify an integer, other than 1, as the value for this property, the buffer size becomes that number of bytes. For example, if you specify allow_large_SAF_groups=21400000, the size of the buffer used to do lookups on SAF groups is 21400000 bytes.

If you do not specify a value for this property, or specify a value of 0, the buffer size is 8192 bytes.

Data Type	Integer
Range	1 - 2147483647
Default	0

com.ibm.ws.sib.ra.inbound.impl.MessageLockExpiry

When a message arrives on the queue that a message-driven bean (MDB) is consuming from, the message is locked and passed to the MDB in the servant region. If the servant region is disabled, or if there is an error processing the message on the servant region, this property defines how long the messaging engine waits before unlocking the message so that it can be delivered again.

Data Type	Integer
Units	milliseconds
Default	300000

Range	A positive integer. The value 0 indicates that the message lock never expires and the messaging engine waits indefinitely for the servant region to process and unlock the message.

control_region_confirm_recovery_on_no_srs

Specifies whether requests are dispatched to servants following the detection of a no-servants situation. This property is ignored if the control_region_dreg_on_no_srs custom property is set to 0.

When this property is set to 1, the controller does not dispatch requests to the servants until it receives a response to message BBOO0297A. This message is issued following a no-servant situation when the server detects that the required minimal number of servants are available to process requests.

When this property is set to 0 (zero), the controller determines when to allow requests to be dispatched to the servants after a no-servant condition is detected.

Data Type	Integer
Acceptable values	0 or 1
Default	0

control_region_dreg_on_no_srs

Specifies whether the controller rejects requests for dispatch within a servant when it detects that no servants are available to process requests.

When this property is set to 1, if the controller detects that there are no servants available to process requests, it rejects requests for dispatch within the servants. It also removes the application server from the registry of servers that workload management (WLM) uses to assign work, and stops the HTTP and message-driven bean (MDB) listeners. If this property is set to 0, then the function is disabled.

When the minimum number of servants become available, the controller registers the application server with WLM again, starts the HTTP and MDB listeners, and allows requests to be dispatched to the servants.

Data Type	Integer
Acceptable values	0 or 1
Default	0

control_region_http_queue_timeout_percent

Specifies the percentage of the HTTP dispatch time limit that is used as the maximum amount of time that an HTTP request can spend on the workload management (WLM) queue. The protocol_http_timeout_output custom property is used to specify the maximum amount of time that an HTTP request can spend on the queue and in dispatch before an error message is issued, which indicates that an HTTP dispatch timeout has occurred.

The queue_timeout_percent request-level Reliability Availability and Serviceability (RAS) attribute overrides the control_region_http_queue_timeout_percent custom property for HTTP requests. You define the request-level RAS attributes in the workload classification file.

Data Type	Integer
Range	0 - 99
Default	0

control_region_https_queue_timeout_percent

Specifies the percentage of the HTTPS dispatch time limit that is used as the maximum amount of time that an HTTPS request can spend on the workload management (WLM) queue. The protocol_https_timeout_output custom property is used to specify the maximum amount of time that an HTTPS request can spend on the queue and in dispatch before an error message is issued, which indicates that an HTTPS dispatch timeout has occurred.

The queue_timeout_percent request-level RAS attribute overrides the control_region_https_queue_timeout_percent custom property for HTTPS requests. You define the request-level RAS attributes in the workload classification file.

Data Type	Integer
Range	0 - 99
Default	0

control_region_iiop_queue_timeout_percent

Specifies the percentage of the IIOP dispatch time limit that is used as the maximum amount of time that an IIOP request can spend on the workload management (WLM) queue.

This property only applies to the time that the request spends on the WLM queue. Use the control_region_wlm_dispatch_timeout custom property if you want to limit the amount of time that the request spends on both the WLM queue and in dispatch,

The queue_timeout_percent request-level RAS attribute overrides the control_region_iiop_queue_timeout_percent custom property for IIOP requests. You define the request-level RAS attributes in the workload classification file.

Data Type	Integer
Range	0 - 99
Default	0

control_region_mdb_queue_timeout_percent

Specifies the percentage of the MDB dispatch time limit that is used as the maximum amount of time that an MDB request can spend on the workload management (WLM) queue.

This property only applies to the time that the request spends on the WLM queue. Use the control_region_mdb_request_timeout custom property if you want to limit the amount of time that the request spends on both the WLM queue and in dispatch,

The queue_timeout_percent request-level RAS attribute overrides the control_region_mdb_queue_timeout_percent custom property for MDB requests. You define the request-level RAS attributes in the workload classification file.

Data Type	Integer
Range	0 - 99
Default	0

control_region_mdb_request_timeout

Specifies the time, in seconds, that the server waits for a message-driven bean (MDB) request to receive a response. If the response is not received within the specified amount of time, the server removes the MDB request, and issues an error message that indicates that an MDB dispatch timeout has occurred.

The request_timeout request-level RAS attribute overrides the control_region_mdb_request_timeout custom property for MDB requests. You define the request-level RAS attributes in the workload classification file.

Set this value to 0 to disable the function.

Data Type	Integer
Units	Seconds
Default	120

control_region_sip_queue_timeout_percent

Specifies the percentage of the Session Initiation protocol (SIP) dispatch time limit that is used as the maximum amount of time that a SIP request can spend on the workload management (WLM) queue.

This property only applies to the time that the request spends on the WLM queue. Use the protocol_sip_timeout_output custom property if you want to limit the amount of time that the request spends on both the WLM queue and in dispatch.

The queue_timeout_percent request-level RAS attribute overrides the control_region_sip_queue_timeout_percent custom property for MDB requests. You define the request-level RAS attributes in the workload classification file.

Data Type	Integer
Range	0 - 99
Default	0

control_region_sips_queue_timeout_percent

Specifies the percentage of the SIP Secure Sockets Layer (SSL) dispatch time limit that is used as the maximum amount of time that a SIP SSL request can spend on the workload management (WLM) queue.

This property only applies to the time that the request spends on the WLM queue. Use the protocol_sips_timeout_output custom property if you want to limit the amount of time that the request spends on both the WLM queue and in dispatch.

The queue_timeout_percent request-level RAS attribute overrides the control_region_sips_queue_timeout_percent custom property for MDB requests. You define the request-level RAS attributes in the workload classification file.

Data Type	Integer
Range	0 - 99
Default	0

control_region_timeout_delay

Specifies the number of seconds a controller waits after detecting a timeout before it terminates the servant. This time delay gives work that is currently running in the servant a chance to complete before the servant is terminated.

The specified length of time period starts when a timeout occurs. When a servant thread completes its current work item and determines that the servant is being terminated, the servant thread waits for the specified length of time instead of selecting a new work item.

When this field is set to 0 the controller terminates a servant as soon as the controller detects a timeout.

Data Type	Integer

Units	Seconds
Default	0

This property is affected by the setting for the server_use_wlm_to_queue_work property:

- If the server_use_wlm_to_queue_work property is set to 1, during the time period specified for the control_region_timeout_delay property, exactly one new request can be processed by each servant worker thread that is idle when the timeout occurred.

  Only one new request is processed because, when the timeout occurs, the idle servant worker thread are paused waiting for WLM to assign each of them a new requests. When WLM assigns new work to one of these threads, the thread becomes active, processes the work, and then acquiesces. Therefore, WLM cannot assign any additional work to this thread.

- If the server_use_wlm_to_queue_work property is set to 0, during the time period specified for the control_region_timeout_delay property, work requests that were not yet dispatched but were queued without affinity to the terminating servant, are requeued to another available servant after the servant termination process completes.

control_region_timeout_dump_action

Specifies the type of dump that is taken whenever a timeout occurs for work that has been dispatched to a servant. This property only applies if the control_region_timeout_delay custom property is set to a non-zero value.

Valid values for this property are SVCDUMP, JAVACORE, HEAPDUMP, TRACEBACK, JAVATDUMP, and NONE. JAVACORE generates a Java core dump. SVCDUMP generates an SVC dump. JAVATDUMP generates a JVM-initiated TDUMP.

Default	TRACEBACK

control_region_timeout_dump_action_session

Specifies the type of dump that is taken whenever a timeout occurs for an HTTP, HTTPS, SIP, or SIPS request that has been dispatched to a servant.

This property only applies if the following corresponding variable is set to SESSION:

protocol_http_timeout_output_recovery

protocol_https_timeout_output_recovery

protocol_sip_timeout_output_recovery

protocol_sips_timeout_output_recovery

Valid values for this property are SVCDUMP, JAVACORE, HEAPDUMP, TRACEBACK, JAVATDUMP, and NONE. JAVACORE generates a Java core dump. SVCDUMP generates an SVC dump. JAVATDUMP generates a JVM-initiated TDUMP.

Default	TRACEBACK

control_region_timeout_save_last_servant

Specifies whether the controller terminates the last available servant when a timeout situation occurs. If the controller does not terminate the last available servant when a timeout situation occurs, other work continues to be processed until a new servant is initialized. However, not terminating the last available servant might cause the loss of system resources if the dispatched servant thread that encountered the timeout situation continues to loop or stops functioning. For example, if timeouts keep occurring, the system might consume a high percentage of the available servant threads.

The functionality of this property depends on the values that you specify for other custom properties:

- If you set this property to 1, and the wlm_minimumSRCount custom property is set to a value that is greater than 1, when a timeout situation occurs, the controller waits until a new servant is initialized before it terminates the last available servant.
- If you set this property to 0, or when a timeout situation occurs, the controller terminates the last available servant. It does not wait for another servant to be initialized.
- If the wlm_dynapplenv_single_server custom property is set to 1, the value that you specify for this property is ignored.

Data Type	Boolean
Default	0

controller_jvm_direct_options

Specifies options that you need to pass directly to the JVM launch in the controller. This property is typically used for JVM options that the JVM cannot read from the options file that is specified as the value of the control_region_jvm_properties_file property. For example, the JVM cannot read the value that is specified for the -memorycheck option in the options file.

If you specify multiple options, use a semicolon to separate the options.

You can use the servant_jvm_direct_options to specify options that you need to pass directly to the JVM launch in the servant. You can use the adjunct_jvm_direct_options custom properties to specify options that you need to pass directly to the JVM launch in the adjunct.

Data Type	String
Default	Empty string
Used by Daemon	No

Daemon_ras_trace_ctraceParms

Specifies the identity of the CTRACE PARMLIB member. The value can be either a two-character suffix, which is added to the CTIBBO string to form the name of the PARMLIB member, or the fully specified name of the PARMLIB member. For example, you can use the 01 suffix, which the system resolves to CTIBBO01. A fully specified name must conform to the naming requirements for a CTRACE PARMLIB member. For details, see z/OS MVS Diagnosis: Tools and Service Aids, GA22-7589.

If the specified PARMLIB member is not found, then tracing is defined to CTRACE, but no connection is available to a CTRACE external writer.

Data Type	String
Default	Empty string
Used by Daemon	Yes

default_internal_work_transaction_class

Specifies the default transaction class for internally processed work within the server.

If an internal classification element is not listed in the wlm_classification_file, or if the wlm_classification_file is not specified, then the default_internal_work_transaction_class setting is used. If a value is specified for internal classification setting listed in the wlm_classification_file, The value specified for the default_internal_work_transaction_class custom property is ignored.

Data Type	String
Default	null (empty string)
Used by Daemon	No

iiop_max_msg_megsize

Specifies, in megabytes, the maximum size for IIOP requests. For example, if you set the property to 35, then any requests over 35 MB are rejected. The minimum value for this property is 10, and the maximum value is 2048. Set the value to 0 or omit this property if you do not want to limit the size of IIOP requests.

gotcha: This custom property only applies to application servers that are running in 64-bit mode. The maximum size for IIOP requests that are being handled by application servers running in 31-bit mode is 10 MB.

Data Type	Integer
Default	0
Used by Daemon	No

iiop_max_send_queue_megsize

Specifies, in megabytes, the maximum amount of data that can be queued up to send asynchronously over a single IIOP connection. If the amount of data queued exceeds the specified value, future IIOP requests over this connection fail with a C9C26A4D minor code. The minimum value for this property is 0, which indicates that there is no limit to the amount of data that can be queued for sending. The maximum value is 2048.

Data Type	Integer
Default	0
Used by Daemon	No

local_comm_max_msg_megsize

Specifies, in megabytes, the maximum size of locally connected communications requests. For example, if you set the property to 35, then any requests over 35 MB are rejected. The minimum value for this property is 10 and the maximum value is 2048. Set the value to 0 or omit this property if you do not want to limit the size of locally connected communications requests.

gotcha: This custom property only applies to application servers that are running in 64-bit mode. The maximum size for IIOP requests that are being handled by application servers running in 31-bit mode is 10 MB.

Data Type	Integer
Default	0
Used by Daemon	No

ola_cicsuser_identity_propagate

Specifies permission for CICS® application level identities to be used for authentication when the registration request is made.

When an application requests an optimized local adapter registration, the programmer can use two security-related bits in the registration flags structure to control identity propagation. For transactions that are inbound from CICS to WebSphere Application Server, bit 29 (x'00',x'00',x'00',x'04') controls how the identity is determined. When this property is turned *on*, the CICS application identity is used in WebSphere server authentication. When this property is turned *off* , the CICS region identity is used.

WebSphere Application Server administrators must set the environment variable to permit CICS application level identities to be used for authentication when the registration request is made. Set the value of to 1 to allow the use of CICS task level identity. If it is undefined or set to 0 (zero), registration requests can only request CICS region level authentication, otherwise if registration flag bit 29 is set to 1, the BBOA1REG registration request fails with a return code 8 and reason code 21. Bit 21 (x'00',x'00',x'00',x'01') controls outbound transaction security propagation. See outbound transactions for information.

Data Type	Integer
Default	
Used by Daemon	

protocol_accept_http_work_after_min_srs

Specifies whether the application server waits until a minimum number of servants are ready to accept work before the application server starts the HTTP transport channels. If this property is set to `true`, then when the minimum number of servants are ready for work, the HTTP transport channels start accepting work. If this property is set to `false`, the HTTP transport channels start when the controller starts.

When this property is set to `true`, the value specified for the Minimum number of instances property determines the number of servants that must be ready before the HTTP transport channels start. To change the setting of the Minimum number of instances property for an application server, in the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Java and process management** > **Server instance**. To change the setting of this property for a deployment manager, in the administrative console, click **System Administration** > **Deployment manager** > **Java and process management** > **Server instance**.

The job output indicates `protocol_accept_http_work_after_min_srs: 1`, if this property is set to `true`, or `protocol_accept_http_work_after_min_srs: 0`, if this property is set to `false`.

Data Type	Boolean
Default	true
Used by Daemon	No

protocol_accept_iiop_work_after_min_srs

Specifies whether the application server waits for a minimum number of servants to be ready to accept work before the application server starts the IIOP transport channels. If this property is set to `1`, when the minimum number of servants is ready for work, then the IIOP transport channels starts accepting work. If this property is set to `0`, the IIOP transport channels start when the controller starts.

When this property is set to `true`, the value specified for the Minimum number of instances property determines the number of servants that must be ready before the IIOP transport channels start. To change the setting of the Minimum number of instances property for an application server, in the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Java and process management** > **Server instance**. To change the setting of this property for a deployment manager, in the administrative console, click **System Administration** > **Deployment manager** > **Java and process management** > **Server instance**.

Data Type	Boolean
Default	0
Used by Daemon	No

protocol_bboc_log_response_failure

Specifies that if the BBOO0168W message is issued, then the failure that is detected when attempting to send a response to a client is recorded. The message is sent to the error log. The message text contains the request method name, the reply status, and the routing information that identifies the client.

Data Type	Boolean
Default	false
Used by Daemon	Yes

protocol_bboc_log_return_exception

Specifies that if the BBOO0169W message is issued, the response that contains the SystemException is recorded. The message is sent to the error log. The message text contains the exception identifier and minor code, the request method name, and the routing information that identifies the client.

gotcha: This property only applies to application servers. This property is ignored if it is specified for a deployment manager.

Data Type	Boolean
Default	false
Used by Daemon	Yes

protocol_giop_level_highest

Specifies the CORBA General Inter-ORB Protocol (GIOP) protocol version level that is used by the application server object request broker (ORB). Valid values are 1.1 and 1.2. Interoperable object references (IORs) that are exported from this server use the GIOP level indicated.

You might need to change the setting of this property from the default value if you use a client ORB that is not shipped as part of the product, and that supports a previous version of the CORBA standard. For example, you might need to change from the default protocol version level of 1.2 to 1.1 to support a client ORB that supports the 1.1 CORBA standard instead of the 1.2 CORBA standard.

gotcha: The maximum GIOP level that the daemon address space supports is 1.1. The GIOP level has of the daemon has no effect on the GIOP levels of the application servers that connect to the daemon.

Data Type	String
Default	1.2
Used by Daemon	Yes

protocol_http_backlog

Specifies the maximum length for the queue of pending connections using HTTP. The value that you specify can be limited by the specification of the SOMAXCONN statement in the TCP/IP profile.

Data Type	Integer
Default	10
Used by Daemon	No

protocol_http_large_data_inbound_buffer

Specifies, in bytes, the limit of the size of incoming HTTP requests when inbound HTTP chunking is disabled. For example, if you set the property to 15728640, any requests over 15 MB are rejected. Specify 0 to reject any requests that are larger than 10 MB.

gotcha: Use this custom property only for an application server that is running in 31-bit mode. If the application server is running in 64-bit mode, then use the protocol_http_large_data_inbound_buffer_64bit custom property to set a limit for this inbound buffer.

Data Type	Integer
Default	0
Used by Daemon	No

protocol_http_large_data_inbound_buffer_64bit

Specifies, in megabytes, the size limit for incoming HTTP requests when inbound HTTP chunking is disabled. For example, if you set the property to 35, any HTTP requests over 35 MB are rejected. Specify 0 for this property if you do not want to limit the size of unchunked HTTP requests.

gotcha: Use this custom property only for an application server that is running in 64-bit mode. If the application server is running in 31-bit mode, then use the protocol_http_large_data_inbound_buffer custom property to set a limit for this inbound buffer.

Data Type	Integer
Default	0
Used by Daemon	No

protocol_http_large_data_response_buffer

Specifies, in bytes, the maximum length of the response buffer that is used for HTTP requests. Responses larger than this value are rejected. Specify a value of 0 if you do not need a large response buffer because all of your HTTP responses are less than 10 MB.

gotcha: This property only applies to application servers. It does not apply to a deployment manager.

Data Type	Integer
Default	104857600
Used by Daemon	No

protocol_http_resolve_foreign_hostname

Specifies whether to perform Domain Name Server (DNS) resolution of the IP address of a foreign client to a DNS registered host name for each established HTTP session. If this property is set to 1, then the DNS host name resolution is performed. If this property is set to 0, then the DNS host name resolution is not performed, and a textual representation of the IP address of the foreign client is used instead of the DNS host name.

Data Type	Boolean
Default	0
Used by Daemon	Yes

protocol_http_timeout_output_recovery

Specifies the recovery action that is taken when an HTTP request does not complete within a designated length of time. Setting this property to SERVANT allows servants to terminate when a timeout occurs. If an HTTP request is under dispatch in a servant when its timeout value is reached, the servant terminates with an ABEND EC3 RSN=04130007. The HTTP request and socket are then cleaned up. If this property is set to SESSION, no attempt is made to disrupt the processing of a dispatched HTTP request within a servant. However, the HTTP request and socket are still cleaned up. Using the SESSION setting might result in a loss of resources if the dispatched HTTP request loops or becomes inactive.

The timeout_recovery request-level RAS attribute overrides the protocol_http_timeout_output_recovery custom property for HTTP requests. You define the request-level RAS attributes in the workload classification file.

Default	SERVANT
Used by Daemon	No

protocol_https_backlog

Specifies the maximum length for the queue of pending connections using HTTPS. The value that you specify can be limited by the specification of the SOMAXCONN statement in the TCP/IP Profile.

Data Type	Integer
Default	10
Used by Daemon	No

protocol_https_cert_mapping_file

Specifies the name of a file containing entries that map IP addresses to server certificate labels. You can set this property at the cell, node, or server level.

depfeat: The protocol_https_cert_mapping_file property is deprecated. Use a workload classification document instead of a transaction class mapping file to classify work requests in a z/OS environment.

When an HTTP SSL connection request is received, the application server checks the IP address against entries in the file specified for this property. If the application server finds a match, the certificate mapped to the IP address is used for the connection. If the application server does not find a match, it checks the protocol_https_default_cert_label property for the name of a certificate. If a certificate name is specified, the application server uses that certificate to establish the connection. If a certificate name is not specified, the default server certificate specified in the RACF SSL keyring, that is owned by the application server, is used to establish the HTTP SSL connection.

protocol_https_default_cert_label

Specifies the label of the server certificate that the application server uses when establishing HTTP SSL connections with the application server. You can set this property at the cell, node, or server level.

depfeat: The protocol_https_default_cert_label property is deprecated. Use a workload classification document instead of a transaction class mapping file to classify work requests in a z/OS environment.

If the name of a certificate is note specified for this property, the default server certificate specified in the RACF SSL keyring, that is owned by the application server, is used to establish the HTTP SSL connection.

protocol_https_timeout_output_recovery

Specifies the recovery action that is taken when an HTTPS request does not complete within a designated length of time. Setting this property to SERVANT allows servants to terminate when a timeout occurs. If an HTTPS request is under dispatch in a servant when its timeout value is reached, the servant terminates with an ABEND EC3 RSN=04130007. The HTTPS request and socket are then cleaned up. If this property is set to SESSION, no attempt is made to disrupt the processing of a dispatched HTTPS request within a servant. However, the HTTPS request and socket are still cleaned up. Using the SESSION setting might result in a loss of resources if the dispatched HTTPS request loops or becomes inactive.

The timeout_recovery request-level RAS attribute overrides the protocol_https_timeout_output_recovery custom property for HTTPS requests. You define the request-level RAS attributes in the workload classification file.

Default	SERVANT
Used by Daemon	No

protocol_iiop_backlog

Specifies the maximum length for the queue of pending connections using the CORBA Internet Inter-ORB protocol (IIOP). The value that you specify might be limited by the specification of the SOMAXCONN statement in the TCP/IP profile.

Data Type	Integer
Default	10
Used by Daemon	Yes

protocol_iiop_backlog_ssl

Specifies the maximum length for the queue of pending connections using IIOP Secure Sockets Layer (SSL). The value that you specify can be limited by the specification of the SOMAXCONN statement in the TCP/IP profile.

Data Type	Integer
Default	10
Used by Daemon	Yes

protocol_iiop_local_propagate_wlm_enclave

Specifies whether to propagate the workload management (WLM) enclave that is associated with the currently dispatched request on an outbound IIOP request that was made to another server on the same z/OS system over local interaddress space communication protocols.

Data Type	Boolean
Default	1
Used by Daemon	No

protocol_iiop_resolve_foreign_hostname

Specifies whether to perform Domain Name Server (DNS) resolution of the IP address of a foreign client to a DNS registered host name for each established IIOP session. If this property is set to 1, then the DNS host name resolution is performed. If this property is set to 0, then the DNS host name resolution is not performed, and a textual representation of the IP address of the foreign client is used instead of the DNS host name.

Data Type	Boolean
Default	1
Used by Daemon	Yes

protocol_jfap_queue_limit

Specifies the number of messages that can reside on the JFAP protocol message queue that is associated with a servant process. This queue is used to hold pending messages that are being sent between the servant process and the controller.

You can specify any positive integer as the value for this property. However, whenever possible, you should use the default value 0. A value of 0 indicates that the number of messages that can reside in the JFAP protocol message queue is four times the number of dispatch threads in the servant. Therefore, this value provides a calculated limit based on workload, and removes the arbitrary restraints imposed by specifying a specific value for this property.

The number of dispatch threads in the servant is controlled by the ORB services **Workload profile** setting. See the topic *ORB services advanced settings on the z/OS platform* for more information about this setting.

You should also review the topic *Configuring MDB throttling for the default messaging provider* for a description of how to tune message-driven beans.

Data Type	Integer
Default	0
Used by Daemon	No

protocol_sip_timeout_output_recovery

Specifies the recovery action that is taken when a SIP request does not complete within a designated length of time. Setting this property to SERVANT allows servants to terminate when a timeout occurs. If a SIP request is under dispatch in a servant when its timeout value is reached, the servant terminates with an ABEND EC3 RSN=04130007. The SIP request and socket are then cleaned up. If this property is set to SESSION, no attempt is made to disrupt the processing of a dispatched SIP request within a servant. However, the SIP request and socket are still cleaned up. Using the SESSION setting might result in a loss of resources if the dispatched SIP request loops or becomes inactive.

The timeout_recovery request-level RAS attribute overrides the protocol_sip_timeout_output_recovery custom property for SIP requests. You define the request-level RAS attributes in the workload classification file.

Default	SERVANT
Used by Daemon	No

protocol_sips_timeout_output_recovery

Specifies the recovery action that is taken when a SIPS request does not complete within a designated length of time. Setting this property to SERVANT allows servants to terminate when a timeout occurs. If a SIPS request is under dispatch in a servant when its timeout value is reached, the servant terminates with an ABEND EC3 RSN=04130007. The SIPS request and socket are then cleaned up. If this property is set to SESSION, no attempt is made to disrupt the processing of a dispatched SIPS request within a servant. However, the SIPS request and socket are still cleaned up. Using the SESSION setting might result in a loss of resources if the dispatched SIPS request loops or becomes inactive.

The timeout_recovery request-level RAS attribute overrides the protocol_sips_timeout_output_recovery custom property for SIPS requests. You define the request-level RAS attributes in the workload classification file.

Default	SERVANT
Used by Daemon	No

ras_debugEnabled

Specifies to use an external debugger tool with the application server for tracing and debugging client and server application components such as JavaServer Pages (JSP) files, servlets, and enterprise beans.

Data Type	Boolean
Default	false
Used by Daemon	Yes

ras_default_msg_dd

Specifies whether to redirect write-to-operator (WTO) messages that use the default routing to SYSPRINT. These messages are redirected to the location identified through the DD card on the JCL start procedure for the server. These WTO messages are typically issued during initialization.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_dumpoptions_dumptype

Specifies the default dump that is used by the signal handler. Do not change this property unless directed to do so by IBM Support personnel.

0	No dump is generated.

1	A ctrace dump is taken.
2	A cdump dump is taken.
3	A csnap dump is taken.
4	A CEE3DMP dump is taken.

Data Type	Integer
Default	3
Used by Daemon	Yes

ras_dumpoptions_ledumpoptions

Specifies the dump options to use with a CEE3DMP dump. If you want more than one option, then separate each option with a blank space. Do not change this property unless directed to do so by IBM service personnel.

Data Type	String
Default	THREAD(ALL) BLOCKS
Used by Daemon	Yes

ras_enhanced_serviceability_level

Specifies the level of diagnostic information that the application server collects. Reducing the value of this property increases the performance of the application server. However, less data is available to IBM Support personnel when diagnosing a problem. The opposite is true when increasing the value of this property. The default value of 5 provides a balance of performance and diagnostic information.

Data type	Integer
Range	0 - 11
Default	5
Used by Daemon	Yes

ras_error_log_version=n

Specifies the version of trace log to display. Valid values are 1 through 3. Version 3 displays a message tag that is defined in the classification file if an application request is associated with the current thread at the time the error occurs, in addition to previous version information. Version 2 displays the server name, cluster name, and cell name, in addition to the base information. For example, `ras_error_log_version=2`.

Data Type	String
Default	3
Used by Daemon	Yes

ras_hardcopy_msg_dd

Specifies to redirect write-to-operator (WTO) messages that are routed to hardcopy. These messages are redirected to the location that is identified through the DD card on the server JCL start procedure. These WTO messages are primarily audit messages that are issued from Java code during initialization.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_log_logstreamName

Specifies the log stream that the product uses for error information. If the specified log stream is not found or not accessible, a message is issued and errors are written to the server job log. If this variable is not specified, the product uses the SYSOUT stream.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_message_routing_console

Specifies a comma-delimited list of WebSphere Application Server message IDs to be rerouted to the operator console (route code 2). Each message displays in the operator console, instead of in the default location.

Data Type	String
Default	None
Used by Daemon	Yes

ras_message_routing_errorlog

Specifies a comma-delimited list of WebSphere Application Server message IDs to be rerouted to the error log. Each message displays in the error log, rather than its default location.

Data Type	String
Default	None
Used by Daemon	Yes

ras_message_routing_hardcopy

Specifies a comma delimited list of WebSphere message IDs to be rerouted to hardcopy (route code 11). Each message shows up in the hardcopy instead of in its default location.

Data Type	String
Default	None
Used by Daemon	Yes

ras_message_routing_none

Specifies a comma-delimited list of WebSphere Application Server message IDs to be ignored. Each message is hidden from one or more normal output locations.

Data Type	String
Default	None
Used by Daemon	Yes

ras_minorcode_action

Specifies the default behavior for gathering documentation about system exception minor codes.

Data Type	String
Default	NODIAGNOSTICDATA
Used by Daemon	Yes

You can also specify the following values.

`CEEDUMP`	Captures callback and offsets. Taking a CEE dumps is a lengthy process, and transaction timeouts can occur during this process.
`TRACEBACK`	Captures Language Environment and UNIX traceback data for the z/OS operating system.
`SVCDUMP`	Captures an MVS dump, but does not produce a dump in the client.

ras_stderr_ff_interval

Specifies the interval of time, in minutes, that the system waits before writing the next form-feed character to standard error (SYSOUT).

gotcha: If you are using JES2, this environment setting can be combined with the SEGMENT= parameter on the SYSPRINT DD statement if you want to segment the output. The value of the SEGMENT= parameter is the number of form-feeds required before the first segment is closed and a new segment is allocated. The SEBMENT= parameter is not supported on JES3.

Data Type	Integer
Default	0
Used by Daemon	Yes

ras_stderr_ff_line_interval

Specifies the number of lines of output that is written between the writing of form-feed characters to standard error (SYSOUT).

Because of uncontrollable factors, such as line wrapping, the product can only approximate the number of lines of output it has written. Therefore the actual number of lines written between the writing of form-feed characters might be plus or minus 5 percent of the value specified for the property.

gotcha: If you are using JES2, this environment setting can be combined with the SEGMENT= parameter on the SYSPRINT DD statement if you want to segment the output. The value of the SEGMENT= parameter is the number of form-feeds required before the first segment is closed and a new segment is allocated. The SEBMENT= parameter is not supported on JES3.

Data Type	Integer
Default	0
Used by Daemon	Yes

ras_stdout_ff_interval

Specifies the interval of time, in minutes, between the writing of a form-feed character to standard output(SYSPRINT).

gotcha: If you are using JES2, this environment setting can be combined with the SEGMENT= parameter on the SYSPRINT DD statement if you want to segment the output. The value of the SEGMENT= parameter is the number of form-feeds required before the first segment is closed and a new segment is allocated. The SEBMENT= parameter is not supported on JES3.

Data Type	Integer
Default	0
Used by Daemon	Yes

ras_stdout_ff_line_interval

Specifies the number of lines of output between the writing of form-feed characters to standard output (SYSPRINT).

Because of uncontrollable factors, such as line wrapping, the product can only approximate the number of lines of output it has written. Therefore the actual number of lines written between the writing of form-feed characters might be plus or minus 5 percent of the value specified for the property.

gotcha: If you are using JES2, this environment setting can be combined with the SEGMENT= parameter on the SYSOUT DD statement if you want to segment the output. The value of the SEGMENT=

parameter is the number of form-feeds required before the first segment is closed and a new segment is allocated. The SEBMENT= parameter is not supported on JES3.

Data Type	Integer
Default	0
Used by Daemon	Yes

ras_tag_wto_messages=n

Specifies whether messages that are written to the operator console will be tagged, if appropriate. Valid values are 0 and 1. For example, `ras_tag_wto_messages=0`.

Data Type	Integer
Default	1
Used by Daemon	Yes

ras_time_local

Specifies whether time stamps in the error log display are in local time or Greenwich Mean Time (GMT). The time stamp is in GMT if this property is set to `false`.

Data Type	Boolean
Default	false
Used by Daemon	Yes

ras_trace_basic

Specifies tracing overrides for particular product subcomponents. Subcomponents, specified by numbers, receive basic and exception traces. If you specify more than one subcomponent, use parentheses and separate the numbers with commas. Do not change this property unless directed to do so by IBM service personnel.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_trace_BufferCount

Specifies the number of trace buffers to allocate.

Data Type	Integer
Valid values	4 through 8
Default	4
Used by Daemon	Yes

ras_trace_BufferSize

Specifies, in bytes, the size of a single trace buffer. You can use the letters `K`, for kilobytes, or `M`, for megabytes.

Data Type	String
Valid values	128K through 4M
Default	1M
Used by Daemon	Yes

ras_trace_defaultTracingLevel

Specifies the default tracing level for the product. Use this variable with the ras_trace_basic and ras_trace_detail variables to set tracing levels for product subcomponents. Do not change this property unless directed by IBM Support personnel.

0	No tracing
1	Exception tracing
2	Basic and exception tracing
3	Detailed tracing, including basic and exception tracing

Data Type	Integer
Default	1
Used by Daemon	Yes

ras_trace_detail

Specifies tracing overrides for particular product subcomponents. Subcomponents, specified by numbers, receive detailed traces. If you specify more than one subcomponent, use parentheses and separate the numbers with commas. Do not change this property unless directed to do so by IBM Support personnel.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_trace_exclude_specific

Specifies product trace points to exclude from tracing activity.

Trace points are specified by 8-digit, hexadecimal numbers. Do not use this property unless directed to do so by IBM service personnel. If IBM service personnel directs you to specify more than one trace point, use parentheses and separate the numbers with commas. You also can specify a variable name by enclosing the name in single quotation marks.

Data Type	String
Default	Empty string
Used by Daemon	Yes

gotcha: Results sometimes depend on the value specified for the ras_trace_minorCodeDefault environment variable. If you specify `ras_trace_minorCodeTraceBacks=ALL` and `ras_minorcode_action=NODIAGNOSTICDATA`, you get a traceback. However, if you specify `ras_trace_minorCodeTraceBacks=(null value)` and `ras_minorcode_action=TRACEBACK`, you also get a traceback. Specifying `ras_trace_minorCodeTraceBacks=(null value)` causes TRACEBACK data to be collected, instead of canceled.

ras_trace_log_version=n

Specifies the version of trace log to display. Valid values are 1 and 2. Version 2 displays a message tag that is defined in the classification file if an application request is associated with the current thread at the time the trace is issued. For example, `ras_trace_log_version=1`.

Data Type	String
Default	2
Used by Daemon	Yes

ras_trace_outputLocation

Specifies where to send trace records. You can specify:

- SYSPRINT
- BUFFER, which sends the trace records to a memory buffer, the contents of which are later written to a CTRACE data set
- TRCFILE, which sends the trace records to the trace data set that is specified on the TRCFILE DD statement in the start procedure for the server.

For servers, you can specify one or more values, separated by a space.

Data Type	String
Default	SYSPRINT BUFFER
Used by Daemon	Yes

ras_trace_specific

Specifies tracing overrides for specific product trace points. Trace points are indicated by 8-digit, hexadecimal numbers. To specify more than one trace point, use parentheses and separate the numbers with commas. You can also specify tracing on a specific environment variable by using the name enclosed in single quotation marks. Do not use this property unless directed to do so by IBM Support personnel.

Data Type	String
Default	Empty string
Used by Daemon	Yes

register_ifaedreg_also

Specifies whether you want z/OS to create SMF Type 89 Subtype 2 records, in addition to SMF Type 89 Subtype 1 records. In previous releases of the product, after the product registered with z/OS, z/OS created SMF Type 89 Subtype 2 records to collect product usage data.

transition: Now, after the product registers with z/OS, z/OS produces SMF Type 89 Subtype 1 records instead of SMF Type 89 Subtype 2 records. If you want to have SMF Type 89 Subtype 2 records created in addition to the SMF Type 89 Subtype 1 records, add the register_ifaedreg_also variable to your WebSphere variables and set this property to 1. To turn off the creation of SMF Type 89 Subtype 2 records, set this variable to 0.

Data Type	Boolean
Default	0
Used by Daemon	No

security_SMF_record_first_auth_user

Specifies whether to record the first authenticated user under request dispatch in the SM120CRE field in the System Management Facility (SMF) server activity record.

If this property is set to 1, then the first authenticated user under request dispatch is written to the SM120CRE field. If this property is set to 0, then the ID of the user under which the server activity began is written to the SM120CRE field.

Data Type	Boolean
Default	0
Used by Daemon	No

servant_jvm_direct_options

Specifies options that you need to pass directly to the JVM launch in the servant. This property is typically used for Java virtual machine (JVM) options that the JVM cannot read from the options file that is specified as the value of the control_region_jvm_properties_file property. For example, the JVM cannot read the value that is specified for the -memorycheck option in the options file.

If you specify multiple options, use a semicolon to separate the options.

You can use the controller_jvm_direct_options and adjunct_jvm_direct_options custom properties to specify options that you need to pass directly to the JVM launch in the controller and adjunct, respectively.

Data Type	String
Default	Empty string
Used by Daemon	No

servant_region_custom_thread_count

Specifies the number of application threads that are used in each of the servants that are running in an application server.

If you specify a value for this custom property, you must set the Workload profile property on the ORB services z/OS additional settings page in the administrative console to CUSTOM before this setting becomes effective. To navigate to this page, in the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Container services** > **ORB service** > **z/OS additional settings**.

bprac: Before you specify a value greater than 100 in a production environment, you should try the value in a test environment to ensure you do not negatively affect performance.

Data Type	Integer
Range	1 - 500
Used by Daemon	No

server_dlls_in_hfs

Specifies whether product DLLs are loaded from the Hierarchical File System (HFS) or from the STEPLIB, LPA, or link list.

Data type	Boolean
Default	1
Used by Daemon	Yes

Setting this property to 1 (the default) indicates that the DLLs should be loaded form the HFS. Setting this property to 0 indicates that the DLLs have been put into the STEPLIB, LPA, or link list.

Tip: You can use the switchModules.sh script to load the DLLs from the HFS into a dataset.

server_region_connect_to_wlm_early

Specifies whether the servant should connect to Workload Manager (WLM) at the beginning or the end of servant initialization.

Data Type	Boolean
Default	0
Used by Daemon	No

Setting this property to 1 enables the servant to connect to WLM at the beginning of servant initialization. After the servant connects to WLM, WLM is able to classify asynchronous work that is started during the remainder of the servant initialization process. A consequence of enabling the servant to connect to WLM at the beginning of servant initialization is that when there are multiple servants defined, the servants all start almost concurrently. Concurrent initialization of multiple servants might cause high CPU usage that needs to be considered.

Setting this property to 0 prevents any asynchronous work that is started during servant initialization from being classified by WLM because the servant does not connect to WLM until the end of the servant initialization process.

server_region_cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the server_region_request_cputimeused_limit custom property.

Valid values	SVCDUMP, JAVACORE, HEAPDUMP, TRACEBACK, JAVATDUMP, and NONE
Range	TRACEBACK
Used by Daemon	No

The cputimeused_dump_action request-level RAS attribute overrides the server_region_cputimeused_dump_action custom property for HTTP, IIOP, SIP, MDB, and optimized local adapter requests. You define the request-level RAS attributes in the workload classification file.

server_region_dpm_dump_action

Specifies the type of dump that is taken when the dispatch progress monitor (DPM) interval has expired for a request. After the dump is taken the DPM interval is reset.

The dpm_dump_action request-level RAS attribute overrides the server_region_dpm_dump_action custom property. You define the request-level RAS attributes in the workload classification file.

Valid values	none, svcdump, javacore, heapdump, javatdump, and traceback
Default	traceback
Used by Daemon	No

server_region_http_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is taken the controller is notified of the unresponsive request. The controller might then terminate the servant based on the values specified for other custom properties, such as server_region_stalled_thread_threshold_percent, control_region_timeout_delay, and control_region_timeout_save_last_servant.

The stalled_thread_dump_action request-level RAS attribute overrides the server_region_http_stalled_thread_dump_action custom property. You define the request-level RAS attributes in the workload classification file.

Valid values	none, svcdump, javacore, heapdump, and traceback.
Default	traceback
Used by Daemon	No

server_region_https_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is taken the controller is notified of the unresponsive request. The controller might then terminate the servant

based on the values specified for other custom properties, such as
server_region_stalled_thread_threshold_percent, control_region_timeout_delay, and
control_region_timeout_save_last_servant.

The stalled_thread_dump_action request-level RAS attribute overrides the server_region_
https_stalled_thread_dump_action custom property. You define the request-level RAS attributes in the
workload classification file.

Valid values   none, svcdump, javacore, heapdump, and traceback.
Default   traceback
Used by Daemon   No

server_region_iiop_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is
taken the controller is notified of the unresponsive request. The controller might then terminate the servant
based on the values specified for other custom properties, such as
server_region_stalled_thread_threshold_percent, control_region_timeout_delay, and
control_region_timeout_save_last_servant.

The stalled_thread_dump_action request-level RAS attribute overrides the server_region_
iiop_stalled_thread_dump_action custom property. You define the request-level RAS attributes in the
workload classification file.

Valid values   none, svcdump, javacore, heapdump, and traceback.
Default   traceback
Used by Daemon   No

server_region_jvm_localrefs

Do not use this property unless directed to do so by IBM Support personnel.

Data Type   Integer
Default   128
Used by Daemon   No

server_region_jvm_logfile

Specifies the Hierarchical File System (HFS) file in which Java Native Interface (JNI) and class debug
messages from the Java virtual machine (JVM) are logged. Use this variable only in a single-server
environment. If you use this property in a multiple-server environment, then all of the servers write to the
same file, and you might have difficulty using the file for diagnostic purposes.

Data Type   String (file name)
Default   Empty string
Used by Daemon   No

server_region_mdb_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is
taken the controller is notified of the unresponsive request. The controller might then terminate the servant
based on the values specified for other custom properties, such as
server_region_stalled_thread_threshold_percent, control_region_timeout_delay, and
control_region_timeout_save_last_servant.

The stalled_thread_dump_action request-level RAS attribute overrides the
server_region_mdb_stalled_thread_dump_action custom property for MDB requests. You define the
request-level RAS attributes in the workload classification file.

Valid values	none, svcdump, javacore, heapdump, and traceback.
Default	traceback
Used by Daemon	No

server_region_recycle_count

Specifies the number of transactions that are processed by a servant process after which the servant process is recycled. Workload management (WLM) ends the servant after all affinity requirements are met. Specify a nonzero value to enable recycling.

You might want to enable recycling if, after running for an extended period, your application is experiencing out-of-memory exceptions. Out-of-memory exceptions can result from memory leakage by your application.

Data Type	Integer
Default	0
Used by Daemon	No

server_region_request_cputimeused_limit

Specifies, in milliseconds, the amount of CPU time that an application request can consume.

The server_region_request_cputimeused_limit custom property helps you to prevent a single application request from monopolizing the available CPU time because it allows you to limit the amount of CPU time that a single request can use. A CPU monitor is invoked when a request is dispatched. If the request exceeds the specified amount of CPU time, the controller considers the request unresponsive. The controller then issues message BBOO0327, to let the requesting application know that the request was unresponsive.

The monitor, that monitors the amount of CPU time that a request is using, typically sends a signal to the dispatched thread when the amount of CPU time used exceeds the specified amount. However, there are situations when this signal cannot be delivered, and the request remains pending. For example, if the thread goes native and invokes a PC routine, the signal remains pending until the PC routine returns.

The cputimeused_limit request-level RAS attribute overrides the server_region_request_cputimeused_limit custom property for HTTP, IIOP, SIP, MDB, and optimized local adapter requests. You define the request-level RAS attributes in the workload classification file.

Data type	integer
Default	0
Used by Daemon	No

server_region_sip_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is taken the controller is notified of the unresponsive request. The controller might then terminate the servant based on the values specified for other custom properties, such as server_region_stalled_thread_threshold_percent, control_region_timeout_delay, and control_region_timeout_save_last_servant.

The stalled_thread_dump_action request-level RAS attribute overrides the server_region_sip_stalled_thread_dump_action custom property. You define the request-level RAS attributes in the workload classification file.

Valid values	none, svcdump, javacore, heapdump, and traceback.
Default	traceback
Used by Daemon	No

server_region_sips_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is taken the controller is notified of the unresponsive request. The controller might then terminate the servant based on the values specified for other custom properties, such as server_region_stalled_thread_threshold_percent, control_region_timeout_delay, and control_region_timeout_save_last_servant.

The stalled_thread_dump_action request-level RAS attribute overrides the server_region_sips_stalled_thread_dump_action custom property. You define the request-level RAS attributes in the workload classification file.

Valid values	none, svcdump, javacore, heapdump, and traceback.
Default	traceback
Used by Daemon	No

server_region_stalled_thread_threshold_percent

Specifies the percentage of threads that can become unresponsive before the controller terminates the servant.

If 0 is specified, the controller terminates the servant as soon as the controller determines that at least one thread has become unresponsive.

Data Type	Integer
Default	0
Used by Daemon	No

server_SMF_request_activity_CPU_detail

Specifies whether you want the CPU usage breakdown section included in any SMF 120 Subtype 9 record that is created.

If this property is set to true, the CPU usage breakdown section is included in any SMF 120 Subtype 9 record that is created.

The setting for this property is ignored if the server_SMF_request_activity_enabled property is set to false.

The SMF_request_activity_CPU_detail request-level RAS attribute overrides the server_SMF_request_activity_CPU_detail custom property for HTTP requests and IIOP requests. You define the request-level RAS attributes in the workload classification file.

Data Type	Boolean
Default	0
Used by Daemon	No

server_SMF_request_activity_enabled

Specifies whether you want the z/OS System Management Facility (SMF) to create an SMF 120 Subtype 9 record.

If you specify true for this property, an SMF 120 Subtype 9 record is created. Because the record is a relatively large, and collecting the data for this record could have an impact on performance, do not enable this property unless you have a specific reason for collecting the data that is included in this record.

You can also lower the performance impact of creating this record by disabling one or more of the following properties.

- server_SMF_request_activity_CPU_detail
- server_SMF_request_activity_security
- server_SMF_request_activity_timestamps

The SMF_request_activity_enabled request-level RAS attribute overrides the server_SMF_request_activity_enabled custom property for HTTP requests and IIOP requests. You define the request-level RAS attributes in the workload classification file.

Data Type	Boolean
Default	0
Used by Daemon	No

server_SMF_request_activity_security

Specifies whether you want the Security data section included in any SMF 120 Subtype 9 record that is created.

If this property is set to true, the Security data section is included in any SMF 120 Subtype 9 record that is created.

The setting for this property is ignored if the server_SMF_request_activity_enabled property is set to false.

The SMF_request_activity_security request-level RAS attribute overrides the server_SMF_request_activity_security custom property for HTTP requests and IIOP requests. You define the request-level RAS attributes in the workload classification file.

Data Type	Boolean
Default	0
Used by Daemon	No

server_SMF_request_activity_timestamps

Specifies whether you want the z/OS formatted timestamps section included in any SMF 120 Subtype 9 record that is created.

If this property is set to true, the z/OS formatted timestamps section is included in any SMF 120 Subtype 9 record that is created.

The setting for this property is ignored if the server_SMF_request_activity_enabled property is set to false.

The SMF_request_activity_timestamps request-level RAS attribute overrides the server_SMF_request_activity_timestamps custom property for HTTP requests and IIOP requests. You define the request-level RAS attributes in the workload classification file.

Data Type	Boolean
Default	0
Used by Daemon	No

server_start_wait_for_initialization_Timeout

Specifies how long the startServer.sh command processing waits for the product initialization process to complete. By default, startServer.sh command processing waits indefinitely until initialization is complete.

Use this property if you want to complete one of the following actions:
- Control how long the application server waits for other dependent servers to start.

- Limit the amount of wait time when trying to debug problems with application initialization. For example, you might not want to wait if web applications, that are automatically started, unexpectedly enter a longer than typical wait state.

Data Type	Integer
Default	0
Used by Daemon	No

server_use_wlm_to_queue_work

Specifies whether WLM is used for workload queuing.

Set this property to 1 if you are using stateless application models. With these models, application objects, such as Enterprise JavaBeans (EJB) and HTTP sessions, are only resident in memory for the life of an individual request. In this situation, you want WLM to dynamically balance individual requests. This configuration allows linear scalability and consistent, repeatable response times.

Set this property to 0 if you are using conversational application models. With these models, a client might hold, and periodically interact with, a reference to a stateful object which is pinned in the memory of one of the JVMs for time that is greater than the duration of an individual request. For example, the client might be using HTTP sessions, stateful session beans, or entity beans that are maintained in memory instead of being stored in a database or file system between requests, as is done in stateless application models.

Conversational application models prevent WLM from dynamically routing individual requests in a clustered environment because the client has affinity to a specific JVM. In this situation, a round robin algorithm is used to handle the initial request from the client. This algorithm evenly distributes the creation of long-term affinities and is the best technique for achieving a balanced utilization of system resources in this type of environment. If you set this property to 0 for conversational application models, you must also set the server_work_distribution_algorithm property to 1.

If you prefer, you can use the round robin capability that WLM provides, instead of the previously described product round robin capability. The differences between the round robin capability that WLM provides and the product round robin capability is explained in the following scenario.

Scenario: A customer starts two clients to talk to a server. The server has two servants, and each servant has multiple threads. The customer expects one client to go to one servant, and the second client to go to the other servant. The behavior of product-provided round robin, that is initiated by specifying `server_use_wlm_to_queue_work=0`, and `server_work_distribution_algorithm=1`, adheres to this expectation. However, the WLM-provided round robin uses up all the threads in the first servant before it starts to use the threads in the next servant. Therefore, in this situation, both clients go to the same servant and the second servant remains idle.

When the server_use_wlm_to_queue_work property is set to 0, the wlm_minimumSRCount and wlm_maximumSRCount properties are set to the same value. Because the work is not going through WLM, WLM only starts the number of servants that are specified for the wlm_minimumSRCount property.

0	The z/OS WLM function is not used.
1	The z/OS WLM function is used.
Default	1
Used by Daemon	No

server_work_distribution_algorithm

Specifies the type of work distribution algorithm that the application server uses for workload balancing. This property is only used if the server_use_wlm_to_queue_work property is set to 0. If the server_use_wlm_to_queue_work property is set to 1, then the value specified for this property is ignored.

0	The hot thread algorithm is used.
	When the hot thread algorithm is used, each new work request is assigned to the first servant that has a thread available to process the request. If none of the servants have an available thread, the request is queued into the global work queue that is shared by all servants. The request is then selected from the global work queue when the next thread becomes available, regardless of which servant owns that thread.
	The goal of the hot thread algorithm is to route requests to the fewest number of servants possible. If the number of concurrent requests is not greater than the number of threads in a servant, all requests are processed by the same servant.
1	The round robin algorithm is used.
	When the round robin algorithm is used, new work requests are distributed evenly across all servants. If all of the servant threads are already processing other work requests, the new request is added to the request queue for a specific servant. The queued request is then selected when it becomes the top request in the queue and a thread becomes available in that servant.
2	The hot robin algorithm is used.
	When the hot robin algorithm is used, new work requests are distributed evenly across all servants. If the assigned servant does not have a thread available to process the request, the request is reassigned to another servant that has an available thread. If none of the servants have an available thread, the new request is queued into the global work queue that all of the servants share. The request is then selected from the global work queue when the next thread becomes available, regardless of which servant owns that thread.
Default	0
Used by Daemon	No

server_wto_on_write_error

Indicates whether the error message `BB000384I ERROR OCCURRED WRITING TO {0}` is written to the SYSLOG when an error occurs while writing to SYSPRINT or SYSOUT.

This property can be set to 0 or 1. When this property is set to `1`, this error message is written to the SYSLOG. When this property is set to `0` no message is issued.

Data Type	Integer
Default	0
Used by Daemon	No

suppress_hung_thread_abend

Controls whether an ABENDDC3 or an ABENDSDC3 code is issued when a hung thread is found during the server stop process. This property disables the detection of a hung thread.

When a STOP command is issued for a server, a hung thread might fail to terminate. By default, the application server issues an ABENDDC3 or ABENDSDC3 code with a 000C000B reason code when this

condition is encountered to avoid any delays and to provide a diagnostic dump earlier in the shutdown process. If you set this property to 1, the application server does not abend the address space for this reason.

Data Type	Boolean
Default	0
Used by Daemon	No

suppress_hung_thread_dump

Controls whether a dump is taken when an ABENDDC3 or an ABENDSDC3 occurs because a hung thread did not terminate in response to a STOP command.

When a STOP command is issued for a server, a hung thread might fail to terminate, which causes an ABENDDC3 or an ABENDSDC3 to occur with reason code 000C000B. Setting this property to 1 prevents a dump from being taken if this abend occurs.

Data Type	Boolean
Default	0
Used by Daemon	No

transaction_recoveryTimeout

Specifies the time, in minutes, that this controller uses to attempt to complete all restarted transactions before issuing a write-to-operator-with-reply (WTOR) message to the console, requesting whether to complete one of the following options:

- Continue the recovery process.
- Stop trying to resolve all restart transactions.
- Write transaction-related information to the job log or hard copy log.
- Terminate.

If the operator replies to continue the recovery process, then the controller attempts recovery for the specified amount of time before reissuing the write-to-operator message. After all the transactions are resolved, the controller terminates. This variable applies only to controllers that are running in peer restart and recovery mode.

Data Type	Integer
Default	15
Used by Daemon	No

WAS_DAEMON_ONLY_adapter_max_conn

Specifies the maximum number of connections supported for a daemon group. A connection is defined as a local communication connection between a client address space and a server within the daemon group.

As client address spaces register with the optimized local adapter APIs, the number of physical connections that the client address spaces want to make with the WebSphere address space is defined. To prevent the connectors from over-running the WebSphere Application Server with requests, the number of connections across the entire daemon group can be set to a maximum number by specifying the adapter_max_conn variable.

If a subsystem connects to the WebSphere server and requests a number of connections that is the maximum for the daemon group, the register request fails with a reason code that indicates that the maximum number of connections has been reached.

The maximum number of physical connections between the WebSphere Application Server and external address spaces is set on the connection factory.

Data Type	Integer
Default	
Used by Daemon	Yes

WAS_DAEMON_ONLY_adapter_max_serv

Specifies support for an optimized local adapter. Support is enabled when this variable is used to start the daemon.

The WebSphere daemon needs to know whether this daemon group is going to support the optimized local adapters, since it manages the shared storage used by the WebSphere server and the associated address spaces. All daemon configuration exists in the `was.env` file for the daemon. The daemon does not have a JVM, therefore it does not have access to the WCCM model.

This variable sets the maximum number of adapter outbound services that can be active for a single registration. When the value is not specified, it defaults to 100.

Data Type	Integer
Default	100
Used by Daemon	Yes

WAS_DAEMON_ONLY_adapter_max_shremem

Specifies support for an optimized local adapter. Support is enabled when this variable is used to start the daemon.

The WebSphere daemon needs to know whether this daemon group is going to support the optimized local adapters, since it manages the shared storage used by the WebSphere server and the associated address spaces. All daemon configuration exists in the `was.env` file for the daemon. The daemon does not have a JVM, therefore it does not have access to the WCCM model.

This variable sets the maximum size of the adapters shared 64-bit memory for adapters control structures for the daemon group. When this value is not specified, it defaults to 32M (33554432).

Data Type	Integer
Default	32M
Used by Daemon	Yes

WAS_DAEMON_ONLY_enable_adapter

Specifies support for an optimized local adapter. Support is enabled when this variable is used to start the daemon.

The WebSphere daemon needs to know whether or not this daemon group is going to support the optimized local adapters, since it manages the shared storage used by the WebSphere server and the associated address spaces. All daemon configuration exists in the `was.env` file for the daemon. The daemon does not have a JVM, therefore it does not have access to the WCCM model.

Set this property to `true` if you want to start the daemon with this variable and enable support for the optimized local adapters. Set this property to `false` if you do not want to start the daemon with this variable and enable support for the optimized local adapters.

Data Type	Boolean
Default	true

| **Used by Daemon** | Yes |

wlm_ae_spreadmin

Specifies the setting of the AE_SPREADMIN parameter on the IWM4SLIP service, which you use for servant address spaces. The IWM4SLIP service is the application environment limit service for workload management (WLM) on the z/OS system. The AE_SPREADMIN parameter indicates how WLM distributes the minimum number of servants across the service classes being used to process work requests.

- If you specify `1` for this property, the application server sets the AE_SPREADMIN parameter to `YES`. WLM distributes the number of servants specified on the **Minimum Number of Instances** field as evenly as possible to all service classes being used to run work requests.
- If you specify `0` for this property, the application server sets the AE_SPREADMIN parameter to `NO`. WLM distributes the number of servants specified on the **Minimum Number of Instances** to service classes as needed in order to meet goals.

Read the topic on controlling the minimum and maximum number of servants for more information about setting the **Minimum Number of Instances** field.

Data Type	Boolean
Default	1
Used by Daemon	No

wlm_classification_file

Specifies the location of your workload classification document. You can set this property at the cell, node, or server level.

The workload classification document is a common XML file that classifies inbound HTTP, IIOP, and message-driven bean (MDB) work. Any rules that are defined in this XML file override the old format HTTP classification. The rules in the common XML file also override any rules that are in the MDB classification file defined by the endpoint_config_file property.

For example, your configuration has the common XML file defined in a server that also has the old format HTTP classification document specified. There are no HTTP classification rules defined in your common XML file, so the old format file is used to classify inbound HTTP requests. However, if your common XML file contains HTTP rules, these classification rules are used instead of classification rules that you defined in the old style HTTP classification document.

wlm_enclavecreate_exstartdefer

Specifies the value that you want used for the Workload Management Service EXSTARTDEFER parameter when you use the IWMECREA Workload Management Service to create an Enclave.

The supported values are `YES` and `NO`.

- Specifying `YES` indicates that the Enclave execution start time should begin when the first IWMSTBGN or IWMEJOIN is executed.
- Specifying `NO` indicates that the Enclave execution start time should not begin when the first IWMSTBGN or IWMEJOIN is executed.

See the z/OS Information Center for the version of z/OS that is running on your system for more information about the IWMECREA Workload Management Service and the EXSTARTDEFER parameter.

Data Type	String
Default	YES
Used by Daemon	No

wlm_servant_start_parallel

Specifies how a server, that is configured to start more than one servant address space, starts these address spaces.

- If you specify `1`for this property, then the server starts the first servant address space. After the first servant is completely initialized, the server then starts the remaining address spaces in parallel, which means that these remaining servant address spaces are all started at the same time.
- If you specify `0` for this property, then the server starts all the address spaces sequentially, which means that a servant address space must be completely initialized before the next servant address space is started.

gotcha: This property requires z/WLM support. Therefore, before specifying this property, verify that you are running the product on z/OS Version 1, Release 9 or higher. If you are running the product on a lower-level z/OS, specifying this property has no effect.

Data Type	Boolean
Default	0
Used by Daemon	No

wlm_stateful_session_placement_on

Specifies whether round robin queuing of HTTP sessions is enabled among servants. You can set this property at the cell, node, or server level.

Set this property to `true` if you want round robin queuing of HTTP sessions enabled among servants. Set this property to `false` if you do not want round robin queuing of HTTP sessions enabled among servants.

See the use of the custom property, **WLMStatefulSession**, for more information on the effect of using **wlm_stateful_session_placement_on**.

Data Type	Boolean
Default	true
Used by Daemon	No

wsadmin_dumpthreads_enable_heapdump

Specifies whether a Java heap dump is generated, in addition to a Java core dump when you issue the wsadmin dumpThreads command.

Set this property to `1` if you want a Java heap dump generated when you issue the wsadmin dumpThreads command. Set this property to `0` if you do not want a Java heap dump generated when you issue the wsadmin dumpThreads command.

Data Type	Boolean
Default	1
Used by Daemon	No

wsadmin_dumpthreads_enable_javatdump

Specifies whether a system TDUMP is generated, in addition to a Java core dump when you issue the wsadmin dumpThreads command.

Set this property to `1` if you want a system TDUMP generated when you issue the wsadmin dumpThreads command. Set this property to `0` if you do not want a system TDUMP generated when you issue the wsadmin dumpThreads command.

Data Type	Boolean
Default	1

Used by Daemon	No

Certificate mapping file entries

The following is the syntax for entries in a certificate mapping file.

```
SSLServerCert label ipaddress
```

where:

label   Is the label of the server certificate in single or double quotes. If the label itself contains a single quote, double quotes are required as the delimiter.

ipaddress
        Is the IP address of the server from which the request was received.

Examples:

```
SSLServerCert 'My Certificate Label' 9.57.4.29
```

```
SSLServerCert "My Co.'s Certificate" 9.57.4.30
```

Managing shared libraries

Shared libraries are files used by multiple applications. Each shared library consists of a symbolic name, a Java class path, and a native path for loading Java Native Interface (JNI) libraries. You can use shared libraries to reduce the number of duplicate library files on your system.

Before you begin

Your applications use the same library files. The applications already are deployed on a server or you currently are deploying the applications.

About this task

Suppose that you have four applications that use the same library file, `my_sample.jar`. Instead of having four copies of `my_sample.jar` on your system after the four applications are deployed, you can define a shared library for `my_sample.jar` and have the four deployed applications use that one `my_sample.jar` library file.

Isolated shared libraries provide another way to reduce the number of library files. Isolated shared libraries each have their own class loader, enabling a single instance of the classes to be shared across the applications. Each application can specify which isolated shared libraries that it wants to reference. Different applications can reference different versions of the isolated shared library, resulting in a set of applications sharing an isolated shared library. With isolated shared libraries, some applications can share a single copy of Library A, Version 1 while other applications share a single copy of Library A, Version 2, for a total of two instances in memory.

Using the administrative console, you can define shared libraries for the library files that multiple applications use and then associate the libraries with specific applications or modules or with an application server. Guidelines for associating shared libraries are as follows:

- Associate a shared library file with an application or module to load the classes represented by the shared library in a local class loader, which can be an application-wide or module-wide class loader.
- Associate an isolated shared library file with an application or module to load the classes represented by the shared library in a separate class loader created for that shared library.
- Associate a shared library file with a server to load the classes represented by the shared library in a server-wide class loader. This class loader is the parent of the application class loader, and the

WebSphere Application Server extensions class loader is its parent. Associating a shared library file with a server associates the file with all applications on the server.

- Do not associate an isolated shared library file with a server if you want a separate class loader for a shared library. If you associate the shared library with a server, the product ignores the isolation setting and still adds files in the shared library to the application server class loader. That is, associating an isolated shared library file with a server associates the file with all applications on the server. The product does not use an isolated shared library when you associate the shared library with a server. Associate an isolated shared library with an application or module.

Instead of using the administrative console to associate a shared library with an application, you can use an installed optional package. You associate a shared library to an application by declaring the dependent library `.jar` file in the `MANIFEST.MF` file of the application. Refer to the Java 2 Platform, Enterprise Edition (J2EE) 1.4 specification, section 8.2 for an example.

Procedure

- Use the administrative console to define a shared library.
  1. Create a shared library.

     On a single-server product, you can define a shared library at the cell, node, or server level.

     On a multiple-server product, you can define a shared library at the cell, node, server, or cluster level.

     Defining a library at one of the these levels does not automatically place the library into a class loader. You must associate the library with an application, module, or server before the product loads the classes represented by the shared library into a local or server-wide class loader.

  2. Associate each shared library with an application, module, or server.
     - Associate a shared library with an application or module that uses the shared library file.

       If you enabled the **Use an isolated class loader for this shared library** setting when creating the shared library, associate the isolated shared library with an application or module to use a separate class loader for the shared library.
     - Associate a shared library with an application server so every application on the server can use the shared library file.
- Use an installed optional package to declare a shared library for an application.
- Remove a shared library.
  1. Click **Environment** > **Shared libraries** in the console navigation tree to access the Shared libraries page.
  2. Select the library to be removed.
  3. Click **Delete**.

  The list of shared libraries is refreshed. The library file no longer displays in the list.

Creating shared libraries

Shared libraries are files used by multiple applications. Create a shared library to reduce the number of duplicate library files on your system.

Before you begin

Determine the full path name or directory of each library file for which you want a shared library.

About this task

To make a library file available to multiple applications deployed on a server, create one or more shared libraries for library files that your applications need. When you create the shared libraries, you can use variables within the library file class paths.

You can create one shared library that points to multiple files or directories. This enables you to maintain a single shared library for files that your applications need.

Or you can create a shared library for each library file that your applications need. This approach is recommended only when you have few library files and few applications that use the files. After you create a shared library, you associate it with each application that uses the library files. If you have multiple shared libraries and multiple applications that use the library files, you must complete many steps to create and associate those shared libraries. It is simpler to use one shared library for related files.

Use the Shared libraries page to create and configure shared libraries.

Procedure

1. Go to the Shared libraries page.

   Click **Environment** > **Shared libraries** in the console navigation tree.

2. Select a shared library scope.

   Change the scope of the collection table to see what shared libraries are in a particular cell, node or server.

   a. Select a cell, node, or server.

      On a multiple-server product, you also can select a cluster. To see the cluster scope, you first must create a cluster on the Server clusters page (**Servers** > **Clusters** > **WebSphere application server clusters**).

   b. Click **Apply**.

   After creating a shared library, you can see whether a shared library can be used on a specific node. Select a scope to see what shared libraries are available to applications installed on or mapped to that scope.

3. Click **New**.

4. Configure the shared library.

   a. On the shared library settings page, specify the name, class path, and any other variables for the library file that are needed.

      If the shared library specifies a native library path, refer to "Configuring native libraries in shared libraries" on page 288.

      To have only one instance of a version of a class shared among applications or modules, make the shared library an isolated shared library. Select **Use an isolated class loader for this shared library**. Using an isolated shared library can reduce the memory footprint when a large number of applications share the library.

   b. Click **Apply**.

What to do next

Using the administrative console, associate your shared libraries with specific applications or modules or with the class loader of an application server. Associating a shared library file with a server class loader associates the file with all applications on the server.

If you enabled the **Use an isolated class loader for this shared library** setting when creating your shared library, associate the shared library with applications or web modules. If you associate the shared library with a server, the product ignores this setting and still adds files in the shared library to the application server class loader. The product does not use an isolated shared library when you associate the shared library with a server.

Alternatively, you can use an installed optional package to associate your shared libraries with an application.

Configuring native libraries in shared libraries

Native libraries are platform-specific library files, including `.dll`, `.so`, or *SRVPGM objects, that can be configured within shared libraries. Native libraries are visible to an application class loader whenever the shared library is associated with an application. Similarly, native libraries are visible to an application server class loader whenever the shared library is associated with an application server.

Before you begin

When designing a shared library, consider the following conditions regarding Java native library support:

- The Java virtual machine (JVM) allows only one class loader to load a particular native library.
- There is no application programming interface (API) to unload a native library from a class loader.

  Native libraries are unloaded by the JVM when the class loader that found the library is collected from the heap during garbage collection.

- Application server class loaders, unlike the native JVM class loader, only load native shared libraries that use the default operating system extension for the current platform. For example, on AIX, native shared libraries must end in `.a` when loaded by application server class loaders. The JVM class loader loads files ending in `.a` or `.so`.
- Application server class loaders persist for the duration of the application server.
- Application class loaders persist until an application is stopped or dynamically reloaded.

  If a shared library that is configured with a native library path is associated with an application, whenever the application is restarted or dynamically reloaded the application might fail with an UnsatisfiedLinkError indicating that the library is already loaded. The error occurs because, when the application restarts, it invokes the shared library class to reload the native library. The native library, however, is still loaded in memory because the application class loader which previously loaded the native library has not yet been garbage collected.

- Only the JVM class loader can load a dependent native library.

  For example, if *NativeLib1* is dependent on *NativeLib2*, then *NativeLib2* must be visible to the JVM class loader. The path containing *NativeLib2* must be specified on Java library path defined by the LIBPATH environment variable.

  The LIBPATH (java.library.path) property is configured using the *process_name*_region_libpath environment variable, such as `control_region_libpath`, `server_region_libpath`, or `adjunct_region_libpath`. For instructions on how to set the region LIBPATH variables, see "Changing the values of variables referenced in BBOM0001I messages" on page 333.

  If a native library configured in a shared library is dependent on other native libraries, the dependent libraries must be configured on the LIBPATH of the JVM hosting the application server in order for that library to load successfully.

About this task

When configuring a shared library on a shared library settings page, if you specify a value for **Native library path**, the native libraries on this path are not located by the WebSphere Application Server application or shared library class loaders unless the class which loads the native library was itself loaded by the same class loader.

Because a native library cannot be loaded more than once by a class loader, it is preferable for native libraries to be loaded within shared libraries associated with the class loader of an application server, because these class loaders persist for the lifetime of the server.

Procedure

1. Implement a static method in the class that loads the native library.

   In the class that loads the native library, call `System.loadLibrary(`*native_library*`)` in a static block. For example:

```
static {System.loadLibrary("native_library");
```
native_library loads during the static initialization of the class, which occurs exactly once when the class loads.

2. On the shared library settings page, set values for **Classpath** and **Native library path** that enable the shared library to load the native library.

   If you want to associate your shared library with an application or module, also select **Use an isolated class loader for this shared library**. If you do not enable this setting, associate the shared library with an application server.

3. Associate the shared library.

   - If you did not enable **Use an isolated class loader for this shared library**, associate the shared library with an application server.

     Associating a shared library with the class loader of an application server, rather than with an application, ensures that the shared library is loaded exactly once by the application server class loader, even though applications on the server are restarted or dynamically reloaded. Because the native library is loaded within a static block, the native library is never loaded more than once.

   - If you enabled **Use an isolated class loader for this shared library**, associate the shared library with an application or module.

     Associating an isolated shared library file with an application or module loads the classes represented by the shared library in a separate class loader created for that shared library. Do not associate an isolated shared library file with a server if you want a separate class loader for a shared library. If you associate the shared library with a server, the product ignores the isolation setting and still adds files in the shared library to the application server class loader. That is, associating an isolated shared library file with a server associates the file with all applications on the server.

     The class loader created for an isolated shared library does not reload and, like a server class loader, exists for the lifetime of a server. For shared native libraries, you can use an isolated shared library to avoid errors resulting from reloading of native libraries.

What to do next

To verify that an application can use a shared library, test the application or examine the class loader in the Class loader viewer. Click **Troubleshooting** > **Class loader viewer** > *module_name* > **Table View**. The classpath of the application module class loader lists the classes used by the shared library.

Shared library collection

Use this page to define a list of shared library files that deployed applications can use.

To view this administrative console page, click **Environment** > **Shared libraries**.

Change the scope to see what shared libraries are in a particular node or server. By default, a shared library is accessible to applications deployed (or installed) on the same node as the shared library file. To change the scope, select the cell, a node, or a server under **Scope**.

On a multiple-server product, you also can select a cluster. To see the cluster scope, you first must create a cluster on the Server clusters page (**Servers** > **Clusters** > **WebSphere application server clusters**). The cluster scope limits the scope of a shared library to cluster members of a particular cluster.

Select a scope before you click **New** and create a shared library. After you create a shared library and map an application to the selected scope, you can associate the shared library with the application or its modules.

- To associate a shared library with an application or module, use the Shared library references page for the application. Click **Applications** > **Application Types** > **WebSphere enterprise applications** > *application_name* > **Shared library references**.

- To associate a shared library with a server class loader, use the settings page for the library reference for the server class loader. Click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Java and Process Management** > **Class loader** > *class_loader_ID* > **Shared library references** > *shared_library_name*.

Name
Specifies a name for the shared library.

Description
Describes the shared library file.

Shared library settings
Use this page to make a library file available to deployed applications.

To view this administrative console page, click **Environment** > **Shared libraries** > *shared_library_name*.

Scope:

Specifies the level of the location of the shared library configuration file.

On single-server installations, the shared library has its configuration file in a location that pertains to the cell, node, or server level.

On multiple-server installations, the shared library has its configuration file in a location that pertains to the cell, node, server, or cluster level.

Data type	String

Name:

Specifies a name for the shared library.

Data type	String

Description:

Describes the shared library.

Data type	String

Classpath:

Specifies a list of paths that the product searches for classes and resources of the shared library.

If a path in the list is a file, the product searches the contents of that Java archive (JAR) or compressed `.zip` file. If a path in the list is a directory, then the product searches the contents of JAR and compressed files in that directory. For performance reasons, the product searches the directory itself only if the directory contains subdirectories or files other than JAR or compressed files.

Press Enter to separate class path entries. Entries must not contain path separator characters such as a semicolon (`;`) or colon (`:`). Class paths can contain variable names that can be substituted using a variable map.

Data type	String
Units	Class path

Native library path:

Specifies the class path for locating platform-specific library files for shared library support; for example, `.dll`, `.so`, or *SRVPGM objects.

If you specify a value for **Native library path**, the native libraries are not located by application or shared library class loaders unless the following conditions exist:

- A class loads the native libraries.
- The application invokes a method in this class which loads the libraries.

  For example, in the class that loads the native library, call `System.loadLibrary(native_library)` in a static block:

  `static {System.loadLibrary("native_library");`

- The **Classpath** specified on this page contains the class that loads the libraries.

Native libraries cannot be loaded more than once by a class loader. Thus, it is preferable for native libraries to use an isolated shared library or to be loaded within shared libraries associated with the class loader of an application server. See the **Use an isolated class loader for this shared library** setting.

Data type	String
Units	Class path

Use an isolated class loader for this shared library:

Specifies whether the shared library has a single isolated shared library shared across its associated applications or web modules.

An isolated shared library enables one instance of the library classes to be shared only among associated applications and web modules. An isolated shared library enables multiple applications or web modules to share a common set of classes across a subset of the applications. Further, an isolated shared library supports versioning and loads the minimum number of library copies. The class loader created for an isolated shared library does not reload and, like a server class loader, exists for the lifetime of a server. For shared native libraries, you can use an isolated shared library to avoid errors resulting from reloading of native libraries.

The default, `false`, is not to isolate the shared library so that each application loads its own instances of the shared library classes.

Using an isolated shared library can reduce the memory footprint when a large number of applications share the library. If you select this option, associate the shared library with applications or Web modules.

Restriction: If you associate the shared library with a server, the product ignores this setting and still adds files in the shared library to the application server class loader. The product does not use an isolated shared library when you associate the shared library with a server. To use an isolated shared library, you must associate the shared library with applications or web modules.

Selecting this option affects the class loader order of the associated application or web module. If the class loader order for a class loader associated with an isolated shared library is **Classes loaded with the parent class loader first** (Parent first), the class loader checks whether a class can be loaded in the following order:
1. Checks whether the associated library class loaders can load the class.
2. Checks whether its parent class loader can load the class.

3. Checks whether it (application or WAR module class loader) can load the class.

If the order is **Classes loaded with the local class loader first (Parent last)**, the class loader checks in the following order:
1. Checks whether it (application or WAR module class loader) can load the class.
2. Checks whether the associated library class loaders can load the class.
3. Checks whether its parent class loader can load the class.

This setting maps to the `isolatedClassLoader` Boolean attribute of the Library object.

Boolean   false

Associating shared libraries with applications or modules

You can associate a shared library with an application or module. Classes represented by the shared library are then loaded in the application's class loader, making the classes available to the application.

Before you begin

This topic assumes that you have created a shared library. The shared library represents a library file used by multiple deployed applications.

You can define a shared library at the cell, node, server, or cluster level.

On a multiple-server product, you also can define a shared library at the cluster level. To see the cluster scope, you first must create a cluster on the Server clusters page (**Servers** > **Clusters** > **WebSphere application server clusters**).

This topic also assumes that you want to use the administrative console, and not an installed optional package, to associate a shared library with an application.

About this task

To associate a shared library with an application or module, create and configure a library reference using the administrative console. A library reference specifies the name of the shared library file.

If you associate a shared library with an application, do not associate the same shared library with a server class loader.

Procedure

1. If you have not done so already, map your application to a target server that is within the scope of the shared library.

   For example, if the shared library scope is the *my_cluster* cluster, map your application to the target *my_cluster* cluster.

2. Click **Applications** > **Application Types** > **WebSphere enterprise applications** > *application_name* > **Shared library references** in the console navigation tree to access the Shared library references page.

3. On the Shared library references page, select an application or module to which you want to associate a shared library.

4. Click **Reference shared libraries**.

5. On the Shared library mapping page, select one or more shared libraries that the application or modules use in the **Available** list, click **>>** to add them to the **Selected** list, and click **OK**.

6. Repeat steps 2 through 4 until you define a library reference instance for each shared library that your application or module requires.

7. On the Shared library references page, click **OK**.

8. Save the changes to the configuration.

Results

When you run the application, classes represented by the shared library are loaded in the application class loader.

The classes are now available to the application or module.

What to do next

To verify an association between an application and a shared library, examine the application class loader in the Class loader viewer. Click **Troubleshooting** > **Class loader viewer** > *module_name* > **Table View**. The classpath of the application module class loader lists the classes used by the shared library.

Shared library reference and mapping settings

Use the Shared library references and Shared library mapping pages to associate defined shared libraries with an application or web module. A shared library is an external Java archive (JAR) file that is used by one or more applications. Using shared libraries enables multiple applications deployed on a server to use a single library, rather than use multiple copies of the same library. After you associate shared libraries with an application or module, the application or module class loader loads classes represented by the shared libraries and makes those classes available to the application or module.

To view the Shared library references console page, click **Applications** > **Application Types** > **WebSphere enterprise applications** > *application_name* > **Shared library references**. To view the Shared library mapping page, click **Reference shared libraries** on the Shared library references page. These pages are the same as the Map shared libraries and Map shared libraries to an entire application or module pages in the application installation and update wizards.

On the Shared library references page, the first element listed is the application. The other elements are modules in the application.

To associate shared libraries with your application or module:

1. Select an application or module.

2. Click **Reference shared libraries**.

3. On the Shared library mapping page, select one or more shared libraries that the application or modules uses in the **Available** list, click **>>** to add them to the **Selected** list, and click **OK**.

A defined shared library for a file that your application or module uses must exist to associate your application or module to the library.

If no shared libraries are defined and the application is installed already, on the Shared library mapping page, click **New** and define a shared library.

You can otherwise define a shared library as follows:

1. Click **Environment** > **Shared libraries**.

2. Specify whether the shared library is visible at the cell, node or server level.

3. Click **New**.

4. On the settings page for the new shared library, specify a name and one or more class paths. If the libraries are platform-specific files such as `.dll`, `.so`, or *SRVPGM objects, also specify a native library path. Then, click **Apply**.

5. Save the administrative configuration.

Application:

Specifies the name of the application that you are installing or that you selected on the Enterprise applications page.

Module:

Specifies the name of the module associated with the shared libraries.

URI:

Specifies the location of the module relative to the root of the application EAR file.

Shared libraries:

Specifies the name of the shared library files associated with the application or module.

Associating shared libraries with servers

You can associate shared libraries with the class loader of a server. Classes represented by the shared library are then loaded in a server-wide class loader, making the classes available to all applications deployed on the server.

Before you begin

This topic assumes that you have created a shared library. The shared library represents a library file used by multiple deployed applications.

About this task

To associate a shared library with the class loader of a server, create and configure a library reference using the administrative console. A library reference specifies the name of the shared library file.

If you associate a shared library with a server class loader, do not associate the same shared library with an application.

Procedure

1. Configure class loaders for applications deployed on the server.
    a. Click **Servers** > **Server Types** > **WebSphere application servers** > ***server_name*** to access the application server setting page.
    b. Set values for the application **Class loader policy** and **Class loading mode** of the server.
2. Create a library reference for each shared library file that your application needs.
    a. In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > ***server_name*** > **Java and Process Management** > **Class loader** > ***class_loader_ID*** .
    b. Click **Shared library references** to access the Library reference page.
    c. Click **Add**.
    d. On the library reference settings page, name the library reference. The name identifies the shared library file that your application uses.
    e. Click **Apply**. The name of the library reference is shown in the list on the Library reference page.

    Repeat the previous steps until you define a library reference for each shared library that your application needs.

What to do next

To verify that an application can use a shared library, test the application or examine the class loader in the Class loader viewer. Click **Troubleshooting** > **Class loader viewer** > *module_name* > **Table View**. The classpath of the application module class loader lists the classes used by the shared library.

Installed optional packages

Installed optional packages enable applications to use the classes in Java archive (.jar) files without having to include them explicitly in a class path. An installed optional package is a .jar file containing specialized tags in its manifest file that enable the application server to identify it. An installed optional package declares one or more shared library .jar files in the manifest file of an application. When the application is installed on a server or cluster, the classes represented by the shared libraries are loaded in the class loader of the application, making the classes available to the application.

When a Java Platform, Enterprise Edition (Java EE) application is installed on a server or cluster, dependency information is specified in its manifest file. The product reads the dependency information of the application (.ear file) to automatically associate the application with an installed optional package .jar file. The product adds the .jar files in associated optional packages to the application class path. Classes in the installed optional packages are then available to application classes.

Installed optional packages used by the product are described in section 8.2 of the Java 2 Platform, Enterprise Edition (J2EE) specification, Version 1.4 at http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf.

The product supports using the manifest file (manifest.mf) in shared library .jar files and application .ear files. The product does not support the Java 2 Platform Standard Edition (J2SE) Installed Optional Package semantics used in the J2SE specification (http://java.sun.com/j2se/1.3/docs/guide/extensions/spec.html), which primarily serve the applet environment. The product ignores applet-specific tags within manifest files.

Sample manifest.mf file

A sample manifest file follows for an application app1.ear that refers to a single shared library file util.jar:

```
app1.ear:
    META-INF/application.xml
    ejb1.jar:
        META-INF/MANIFEST.MF:
            Extension-List: util
            util-Extension-Name: com/example/util
            util-Specification-Version: 1.4
        META-INF/ejb-jar.xml

util.jar:
    META-INF/MANIFEST.MF:
        Extension-Name: com/example/util
        Specification-Title: example.com's util package
        Specification-Version: 1.4
        Specification-Vendor: example.com
        Implementation-Version: build96
```

The syntax of a manifest entry depends on whether the entry applies to a member with a defining role (the shared library) or a member with a referencing role (a Java EE application or a module within a Java EE application).

Manifest entry tagging

Main tags used for manifest entries include the following:

Extension-List

A required tag with variable syntax. Within the context of the referencing role (application's manifest), this is a space delimited list that identifies and constructs unique Extension-Name, Extension-Specification tags for each element in the list. Within the context of the defining role (shared library), this tag is not valid.

Extension-Name

A required tag that provides a name and links the defining and referencing members. The syntax of the element within the referencing role is to prefix the element with the `<ListElement>` string. For each element in the Extension-List, there is a corresponding `<ListElement>`-Extension-Name tag. The defining string literal value for this tag (in the above sample `com/example/util`) is used to match (in an equality test) the corresponding tags between the defining and referencing roles.

Specification-Version

A required tag that identifies the specification version and links the defining and referencing members.

Implementation-Version

An optional tag that identifies the implementation version and links the defining and referencing members.

Further information on these tags is in the `.jar` file specification at http://java.sun.com/j2se/1.4.2/docs/guide/jar/jar.html#Manifest%20Specification.

Using installed optional packages

You can associate one or more shared libraries with an application using an installed optional package that declares the shared libraries in the application's manifest file. Classes represented by the shared libraries are then loaded in the application's class loader, making the classes available to the application.

Before you begin

Read about installed optional packages in "Installed optional packages" on page 295 and in section 8.2 of the Java 2 Platform, Enterprise Edition (J2EE) specification, Version 1.4 at http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf.

WebSphere Application Server does not support the Java 2 Platform Standard Edition (J2SE) Installed Optional Package semantics used in the J2SE specification (http://java.sun.com/j2se/1.3/docs/guide/extensions/spec.html), which primarily serve the applet environment. WebSphere Application Server ignores applet-specific tags within manifest files.

About this task

Installed optional packages expand the existing shared library capabilities of an application server. Prior to Version 6.0, an administrator was required to associate a shared library to an application or server. Installed optional packages enable an administrator to declare a dependency in an application's manifest file to a shared library, with installed optional package elements listed in the manifest file, and automatically associate the application to the shared library. During application installation, the shared library `.jar` file is added to the class path of the application class loader.

If you use an installed optional package to associate a shared library with an application, do not associate the same shared library with an application class loader or a server class loader using the administrative console.

Procedure

1. Assemble the library file, including the manifest information that identifies it as an extension.

Two sample manifest files follow. The first sample manifest file has application `app1.ear` refer to a single shared library file `util.jar`:

app1.ear:
```
    META-INF/application.xml
    ejb1.jar:
        META-INF/MANIFEST.MF:
            Extension-List: util
            util-Extension-Name: com/example/util
            util-Specification-Version: 1.4
        META-INF/ejb-jar.xml
```

util.jar:
```
    META-INF/MANIFEST.MF:
        Extension-Name: com/example/util
        Specification-Title: example.com's util package
        Specification-Version: 1.4
        Specification-Vendor: example.com
        Implementation-Version: build96
```

The second sample manifest file has application `app1.ear` refer to multiple shared library `.jar` files:

app1.ear:
```
    META-INF/application.xml
    ejb1.jar:
        META-INF/MANIFEST.MF:
            Extension-List: util1 util2 util3
            Util1-Extension-Name: com/example/util1
            Util1-Specification-Version: 1.4
            Util2-Extension-Name: com/example/util2
            Util2-Specification-Version: 1.4
            Util3-Extension-Name: com/example/util3
            Util3-Specification-Version: 1.4
        META-INF/ejb-jar.xml
```

util1.jar:
```
    META-INF/MANIFEST.MF:
        Extension-Name: com/example/util1
        Specification-Title: example.com's util package
        Specification-Version: 1.4
        Specification-Vendor: example.com
        Implementation-Version: build96
```

util2.jar:
```
    META-INF/MANIFEST.MF:
        Extension-Name: com/example/util2
        Specification-Title: example.com's util package
        Specification-Version: 1.4
        Specification-Vendor: example.com
        Implementation-Version: build96
```

util3.jar:
```
    META-INF/MANIFEST.MF:
        Extension-Name: com/example/util3
        Specification-Title: example.com's util package
        Specification-Version: 1.4
        Specification-Vendor: example.com
        Implementation-Version: build96
```

2. Create a shared library that represents the library file assembled in step 1. This installs the library file as a shared library.

3. Copy the shared library `.jar` file to the cluster members.

4. Assemble the application, declaring in the application manifest file dependencies to the library files named the manifest created for step 1.

5. Install the application on the server or cluster.

Results

During application installation, the shared library `.jar` files are added to the class path of the application class loader.

Library reference collection

Use this page to view and manage library references that define how to use global libraries. For example, you can use this page to associate shared library files with a deployed application.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Java and Process Management** > **Class loader** > *class_loader_ID* > **Shared library references** .

If no shared libraries are defined in your environment, such as at the node or server scope, after you click **Add** a message is displayed stating that you must define a shared library before you can create a library reference. A shared library is a container-wide library file that deployed applications can use. To define a shared library, click **Environment** > **Shared libraries** and specify the scope of the container. Then, click **New** and specify a name and one or more paths for the shared library. After you define a shared library, return to this page, click **Add**, and create a library reference.

Library name

Specifies a name for the library reference.

Library reference settings

Use this page to define library references, which specify how to use global libraries.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Java and Process Management** > **Class loader** > *class_loader_ID* > **Shared library references** > *library_reference_name*.

A shared library is a container-wide library file that deployed applications can use. To define a shared library, click **Environment** > **Shared libraries** and specify the scope of the container. Then, click **New** and specify a name and one or more paths for the shared library.

Library name:

Specifies the name of the shared library to use for the library reference.

Data type   String

Setting up peer restart and recovery

To allow the product to restart on an alternate system, the following prerequisites must be installed on every system (your original system as well as any systems intended for recovery) before reconfiguring the ARM policies to enable peer restart and recovery.

Before you begin

depfeat: Peer Restart and Recovery (PRR) functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of Peer Restart and Recovery for transaction recovery. See the topic *Transaction support in WebSphere Application Server* for more information about the integrated high availability support for the transaction service subcomponent and how to configure it for peer recovery of transactions being processed on a application server that fails.

You must also make sure all of the systems, where you might need to perform restart, are part of the same RRS log group.
- z/OS Version 1.2 or higher
- BCP APAR OA01584
- RRS APARs OA02556 and OA2556
- WebSphere Application Server Version 5 or higher

Installing the prerequisite service updates on all of these systems will not hinder your current running environment if you want to continue to only restart in place. However, if this service is not installed, there is a possibility that the controller will not be able to move back. OTS will attempt to restart on the alternate system and fail. If there are any URs that are unresolved with RRS once this happens, the controller will not be allowed to restart on the home system until RRS is cancelled on the alternate system. For more information on OTS and RRS, see *z/OS MVS Programming: Resource Recovery*.

If you do not plan to use peer restart and recovery, you do not need to abide by these functional prerequisites. Your system will instead use the restart-in-place function.

The following products all support RRS. Individually, they also support peer restart and recovery, providing the above prerequisites are all properly installed:
- DB2 Version 7 or higher
- IMS™ Version 8 or higher
- CICS Version 1.3 or higher
- MQSeries® Version 5.2 or higher

In addition to the preceding products, many JTA XAResource Managers can be used to assist in a the product peer restart and recovery. Consult your JTA XAResource Manager's documentation to determine if it supports restarting on an alternate system.

gotcha: When setting up the ARM policy for a sysplex, make sure that both systems have the same level of the Application Server installed. For example, you cannot use an application server that is running WebSphere Application Server Version 5.1 to perform peer restart and recovery for an application server that is running WebSphere Application Server Version 6.0.1.

Prior to using peer restart and recovery:

- You must ensure that the location service Daemon and node agent are already running on all systems that might be used for recovery. Otherwise, the recovering system might attempt to recover on a system that is not running the location service Daemon and node agent. If this happens, the server will fail to start, and recovery will fail.

Clients will see a performance impact if the systems are running at capacity. In an attempt to minimize the memory and CPU impact on the alternate system, the enterprise bean and web containers are not restarted for servers running in peer-restart mode. This means that application servers that are in the state of being recovered will not be able to accept any inbound work.

About this task

After the prerequisites are installed, starting a server on a system to which it was not configured implicitly places the server into peer restart and recovery mode. If you configured your XA Partner log to write to a non-shared HFS, or if you are using a JTA XA Resource Manager, you need to perform the following steps before starting a server:

Procedure

1. (Required only if you are using a non-shared HFS.) Enable non-shared HFS support. When using a non-shared HFS, the configuration settings must be replicated across the different systems in the

sysplex. This is done automatically by the deployment manager and node agent. To enable this support, each node agent in your configuration must be set as a recovery node. This change is made in the administrative console:

   a. In the administrative console navigation, select **System Administration > Node agents**.

   b. Select a node agent from the list.

   c. In the Additional Properties section, select **File Synchronization Service**.

   d. In the Additional Properties section, , select **Custom properties**.

   e. Select **New**.

   f. Enter `recoveryNode` for **Name**, and `true` for **Value**. The **Description** field can be left blank.

   g. Repeat steps 3-7 for each node agent in your configuration.

   h. Save your configuration.

2. (Required only if you are using JTA XAResource Managers.) Make appropriate logs and classes are available on the alternate system If you plan to use peer restart and recovery, and your applications access JTA XAResource Managers, you must ensure that the appropriate logs and classes are available on the alternate system.

   a. Point the product variable TRANLOG_ROOT to a shared HFS. The TRANLOG_ROOT variable must point to a shared HFS, to which all systems in the cell can write. The XA partner log is stored here, and the alternate system must be able to read and update this log.

     1) In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*.

     2) Under Container Services, click **Transaction Service**.

     3) Enter the directory of the shared HFS in the **Transaction log directory** field.

   b. Store the driver (i.e., JDBC Driver, JMS Provider, or JCA Resource Adapter, etc.) for each JTA XAResource Manager in an HFS that is readable by all systems in the cell. For example, if your connector is a JDBC driver for a database, the driver would likely be stored in a read-only HFS that is accessible by all systems in the sysplex. This allows the alternate system to read the saved classpath for the resource, and reconstruct it during a restart.

    If the connector used to access a JTA XAResource Manager is not stored in an HFS that is readable by all systems that might be used for recovery, when an application server restarts on an alternate system, it will either appear that there is no XA recovery work to do, or it will be impossible to load the classes necessary to communicate with the JTA XAResource Manager

3. Resolve InDoubt units.

   During a recovery, there will be instances when manual intervention is required to resolve InDoubt units. You will need to use RRS panels for this manual intervention.

Peer restart and recovery

The goal of every system is to have as little downtime as possible. Sometimes, however, system failures are inevitable. For example, a system failure might occur because the power unexpectedly goes out in your main system. When a system failure occurs, a restart action you can take is to restart on a peer system in the sysplex. This type of restart uses the peer restart and recovery function. Starting a server on a system to which it was not configured implicitly places it into peer restart and recovery mode.

depfeat: Peer restart and recovery (PRR) functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of Peer Restart and Recovery for transaction recovery.

When you experience a main system failure that results in InDoubt transactions with unknown outcomes, you need to obtain those intended transactional outcomes (ideally correctly) before the data can be utilized again. Peer restart and recovery provides an automated means of accomplishing this by restarting the controller on a peer system so that the "locks" that block the data can be dropped and the outcomes determined. This is in contrast to how a system usually handles a failure by automatically rolling back.

If a failure occurs, automatic restart management:

* Can restart the product and related servers on the same system, or
* Can use the peer restart and recovery function to restart related servers on an alternate system in the cell.

  The server is not a recoverable *resource* manager. It is a recoverable *communication manager*. It has no recoverable locks of its own and it does not need to manage locks nor manage lock states in a log. It just needs to make sure that both callers and callees are connected in each of the communications sessions of a distributed transaction.

Peer restart and recovery restarts the controller on another system and goes through the transaction restart and recovery process so that we can assign outcomes to transactions that were in progress at the time of failure. During this transaction restart and recovery process, data might be temporarily inaccessible until the recovery process is complete. The restart and recovery process does not result in lost data.

Resource managers, such as DB2, that were being accessed at the time of failure may hold locks that are scoped to a transaction UR (unit of recovery). Once an outcome has been assigned to a UR, the resource managers will, generally, drop those locks.

When might PRR fail to recover servers

The major reason for peer restart and recovery (PRR) failure is if you experience a network outage while in the process of recovering. If the system cannot reach the superior or subordinate because the network is dead, communications cannot reestablish and the transaction cannot completely resolve.

depfeat: Peer restart and recovery functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of peer restart and recovery for transaction recovery.

When the product cannot automatically resolve all of the URs returned from RRS at restart, RRS will not allow the application server to move back to the home (original) system. If the application serve tries to go back while URs are still incomplete, you will receive an error code (C9C2186A) and a message describing an F02 return code from ATRIBRS. In order to get around this, manual resolution is required to mark the server for "restart anywhere." RRS will do that once all of the URs in which the product is involved are *forgotten*. If RRS fails to mark the server *restart anywhere*, the server, upon failure, is required to start on the recovery system. This is not good because it doesn't allow you to move the server back to its true home system.

The ultimate goal of this is to resolve all transactions that the application server (the server instance-owned interests that could not complete recovery) is involved in, and then, if necessary, remove all of the application serve interests that remain in those URs. Once that is complete, browsing the RM data log will show if the resource manager is marked "restart anywhere."

You **want** to see:

```
RESOURCE MANAGER=BSS00.SY1.BBOASR4A.IBM
RESOURCE MANAGER MAY RESTART ON ANY SYSTEM
```

You do **not** want to see:

```
RESOURCE MANAGER=BSS00.SY2.BBOASR4A.IBM
RESOURCE MANAGER MUST RESTART ON SYSTEM SY2
```

Using RRS panels to resolve InDoubt units of recovery

Use this task to better understand messages received when using peer restart and recovery.

Before you begin

gotcha: Peer Restart and Recovery (PRR) functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of Peer Restart and Recovery for transaction recovery. See the topic *Transaction support in WebSphere Application Server* for more information about the integrated high availability support for the transaction service subcomponent.

There are RRS version requirements that you must heed when using peer restart and recovery. For more information on these requirements, see *z/OS MVS Programming: Resource Recovery*.

About this task

If you receive the console message:

```
BBOT0015D OTS UNABLE TO RESOLVE ALL INCOMPLETE TRANSACTIONS FOR SERVER
string.  REPLY CONTINUE OR TERMINATE.
```

Procedure

1. Note the server name specified for *string.* in the message.
2. Go to the SYSPRINT, that is the status queue for that server, and search for messages BBOT0019 - BBOT0022 that refer to that server.
3. Read the resulting messages.
4. Stop the server.

   RRS does not allow an operator to resolve an InDoubt UR if the DSRM for that UR is active at the time. Therefore, you must stop the server. To do this, reply `TERMINATE` to the CONTINUE/TERMINATE WTOR.

What to do next

You can use the following non-console messages to trigger restart and recovery automation. These messages provide information about daemon activities. Pay particular attention to the URID, XID FormatId, XID Gtrid, and XIDBqual attributes. You need to use these pieces of information when you manually resolve the relevant units of work via the RRS panels.

- **BBOO003E** WEBSPHERE FOR z/OS CONTROL REGION string ENDED ABNORMALLY, REASON= *hstring*.
- **BBOO009E** WEBSPHERE FOR z/OS DAEMON string ENDED ABNORMALLY, REASON= *hstring*.
- **BBOO0171I** WEBSPHERE FOR z/OS CONTROL REGION string NOT STARTING ON CONFIGURED SYSTEM *string*
- The following messages, which are written only in recovery and restart mode, provide details about transactions that cannot be resolved during restart and recovery:
  - **BBOT0008I** TRANSACTION SERVICE RESTART INITIATED ON SERVER *string*
  - **BBOT0009I** TRANSACTION SERVICE RESTART UR STATUS COUNTS FOR SERVER string: IN-BACKOUT= *dstring*, IN-DOUBT= *dstring*, IN-COMMIT= *dstring*
  - **BBOT0010I** TRANSACTION SERVICE RESTART AND RECOVERY ON SERVER string IS COMPLETE
  - **BBOT0011I** SERVER *string* IS COLD STARTING WITH RRS
  - **BBOT0012I** SERVER *string* IS WARM STARTING WITH RRS
  - **BBOT0013I** TRANSACTION SERVICE RESTART AND RECOVERY ON SERVER *string* IS COMPLETE. THE SERVER IS STOPPING.
  - **BBOT0014I** TRANSACTION SERVICE RECOVERY PROCESSING FOR RRS URID ' *string* ' IN SERVER *string* IS COMPLETE.
  - **BBOT0016I** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER *string* IS NOT COMPLETE. THE SERVER IS STOPPING DUE TO OPERATOR REPLY.
  - **BBOT0017I** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER *string* IS CONTINUING DUE TO OPERATOR REPLY.

- **BBOT0018I** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER *string* IS STILL PROCESSING *dstring* INCOMPLETE UNIT(S) OF RECOVERY.
- **BBOT0019W** UNABLE TO RESOLVE THE OUTCOME OF THE TRANSACTION BRANCH DESCRIBED BY URID: ' *string* ' XID FORMATID: ' *string* ' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' BECAUSE THE OTS RECOVERY COORDINATOR FOR SERVER *string* ON HOST *string*: *dstring* COULD NOT BE REACHED.
- **BBOT0020W** UNABLE TO PROVIDE THE SUBORDINATE OTS RESOURCE IN SERVER string ON HOST *string*: *dstring* WITH THE OUTCOME OF THE TRANSACTION DESCRIBED ON THIS SERVER BY URID: ' *string* ' XID FORMATID: ' *string*' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' BECAUSE THIS SERVER HAS BEEN UNABLE TO RESOLVE THE OUTCOME WITH A SUPERIOR NODE.
- **BBOT0021W** UNABLE TO *string* THE SUBORDINATE OTS RESOURCE IN SERVER *string* ON HOST string: dstring FOR THE TRANSACTION DESCRIBED ON THIS SERVER BY URID: ' *string* ' XID FORMATID: ' *string* ' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' OR ANOTHER RESOURCE INVOLVED IN THIS UNIT OF RECOVERY BECAUSE ONE OR MORE RESOURCES COULD NOT BE REACHED OR HAVE NOT YET REPLIED.
- **BBOT0022W** UNABLE TO FORGET THE TRANSACTION WITH HEURISTIC OUTCOME DESCRIBED ON THIS SERVER BY URID: ' *string* ' XID FORMATID: '*string* ' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' BECAUSE THE SUPERIOR COORDINATOR FOR SERVER *string* ON HOST *string*: *dstring* HAS NOT INVOKED FORGET ON THE REGISTERED RESOURCE.

Tips for RRS Operations

See *z/OS MVS Programming: Resource Recovery*, for RRS operations guidelines.

Tips for RRS operations:
- If you have configured your log streams to the coupling facility, then monitor your log streams to ensure offload is not occurring. RRS will perform better if its recovery logs do not offload.

  gotcha: Proper sizing of the RRS logs is important. Too small and you get reduced throughput since logger is off-loading the logs too frequently. Too large and you could overflow your coupling facility.
- Keep the main and delayed (only contains active or live data) logs in your coupling facility. Make sure the CF definitions don't overflow.

  gotcha: A commit cannot occur until the log record is written.
- Until you stabilize your workloads, it is a good idea to use the archive log. If you have an archive log configured, RRS will unconditionally use it. However, there is a performance penalty for using it.

Resolving InDoubt units if you receive a BBOT00*xx*W message

If you receive a BBOT00*xx*W message, you must use RRS panels to view the outcome of other branches in the transaction and set the outcomes of InDoubt units to match the outcome of those other branches.

Before you begin

When a BBOT00*xx*W message is displayed, there is a possibility of an heuristic outcome. See *z/OS MVS Programming: Resource Recovery* for more information on how to use the RRS panels and what administrative access (RACF access to the facility class, for example) is needed to resolve URs and remove interests.

About this task

Perform the following steps to remove an expression of interest in this UR if you receive any of the following error messages:

- Messages BBOT0019W and BBOT0020W, which appear together, indicate that this server could not determine the outcome from its superior. Message BBOT0020W, describes the resource to which the product could not provide an outcome.
- Message BBOT0021W, which indicates that a server has determined the transaction outcome but has not been able to communicate it to its subordinates.
- Message BBOT0022W, which indicates that the transaction outcome was determined and communicated to the subordinate, but the subordinate resource has not been "forgotten."

Procedure

1. Select option 3, "Display/Update RRS Unit of Recovery information" on the main RRS panel.
2. To view the details of this URID, enter it in the "URID Pattern" field on the query panel. Press the **Enter** key to execute the query. The query results should display the UR. Take note of whether the state of this UR is InCommit, InBackout or InForget. In the column labeled S, enter v to display the details for this UR.
3. Remove the OTS interest. The RRS Unit of Recovery Details panel opens. Near the bottom of the panel, find the Expressions of Interest heading. This heading is followed by one or more rows that represent each individual expression of interest in this UR. Complete these steps to remove the OTS interest:
   a. Find the row that represents the OTS interest. This row will have an RM name in the form of BSS00.*xxx*.*yyy*.IBM, where *xxx* is the system to which the server was configured and *yyy* is the specific server name.
   b. Type r in the column labeled S to indicate that you want to remove this interest.
   c. Press the **Enter** key to execute the query.
4. Press the **Enter** key to confirm the removal of this interest. The RRS Remove Interest Confirmation panel opens. The RM name and UR identifier fields are pre-filled. Press the **Enter** key to confirm the removal of this interest.

Results

You know you are done when RRS marks the subordinate server as restart anywhere. Determine this by choosing option 1 under Browse and RRS log stream, and then choosing sub-option 4 under RRS Resource Manager Data log.

What to do next

Any subordinate nodes that restart and ask this server about this UR can not obtain this information. If you restart the server containing these nodes, they might be assigned an outcome that is different from the outcome of the transaction. You must manually resolve these nodes before you bring up the servers and start the server for which you just released the UR.

Resource Recovery Services Operations

This topic provides tips for using the z/OS Resource Recovery Services with this product.

Tips for RRS Operations

See *z/OS MVS Programming: Resource Recovery*, for RRS operations guidelines.

Tips for RRS operations:

- If you have configured your logstreams to the coupling facility, then monitor your log streams to ensure offload is not occurring. RRS will perform better if its recovery logs do not offload.

   gotcha: Proper sizing of the RRS logs is important. Too small and you get reduced throughput since logger is off-loading the logs too frequently. Too large and you could overflow your coupling facility.

- Keep the main and delayed (only contains active or live data) logs in your coupling facility. Make sure the CF definitions don't overflow.

  gotcha: A commit cannot occur until the log record is written.
- Until you stabilize your workloads, it is a good idea to use the archive log. If you have an archive log configured, RRS will unconditionally use it. However, there is a performance penalty for using it.

Recovering with JTA XAResource managers

When a JTA XAResource manager is enlisted in a global transaction, it cannot express an interest in the z/OS Resource Recovery Services unit of recovery (UR) like an RRS resource manager can. Instead, the product transaction service will save information in its RRS interest indicating that a JTA Resource Manager was enlisted in the transaction.

Purpose

When you look at the UR through the RRS panels, you will not see an interest for each XA transaction branch, as you would for a resource manager like DB2 or CICS interest.

Because of the differences between RRS and JTA XAResource Managers, there is a different set of errors that can occur when dealing with a JTA XAResource. The following sections describe errors you might see when recovering with a JTA XAResource Manager. Some of these errors are expected, while others may indicate that there is another type of problem, such as connectivity, that needs to be addressed.

This topic describes Peer Restart and Recovery messages that are unique to the z/OS environment.

Messages
- **BBOT0025D:** OTS HAS ENCOUNTERED A LOG DATA MISMATCH. REPLY CONTINUE IF THIS IS EXPECTED OR TERMINATE IF UNEXPECTED.

  This message is issued when the restart epoch in the XA partner log for the product does not match the restart epoch in RRS. These logs must remain in sync to guarantee the atomic outcomes of distributed transactions.

  If one or the other, but not both logs, were restored from a backup, a mismatch will occur. Since the XA partner log is maintained in the JVM, this error can also occur if the controller is started but then is canceled before the JVM is initialized. The RRS log stream will have been replayed before the XA partner log was initialized.

  This message gives the operator an opportunity to cancel recovery and determine why the logs are not in sync. If the machine is not in production and data integrity is not an issue, the operator may reply **CONTINUE** and recovery will attempt to complete with the mismatched logs. However, the results of this response are unpredictable. If the operator replies **TERMINATE**, the application server will shut down, and the problem can be investigated before completing recovery.
- **BBOT0026I:** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER %s IS STILL PROCESSING AN UNKNOWN NUMBER OF XA TRANSACTIONS.

  This message is issued when the application server is unable to initiate contact with each JTA XAResource in its log. Since each JTA XAResource maintains its own logs, it is impossible to know how many transactions there are to recover. Look in the servant region for messages **WTRN0019**, and **WTRN0025**. These messages will help you determine what may be preventing the application server from communicating with these JTA XAResource Manager.

Creating application servers

During the installation process, the product creates a default application server, named server1. Most installations require several application servers to handle the application serving needs of their production environment. You can use the command-line tool or the administrative console to create additional application servers.

Before you begin

Determine if you want to use the application server that you are creating as part of a cluster. If this application server is going to be part of a cluster, you must use the Create a new cluster wizard instead of the Create a new application server wizard to create this application server. The topic *Adding members to a cluster* describes how to use the Create a new cluster wizard.

About this task

To create a new application server that is not part of a cluster, you can either use the Profile Management tool, the createApplicationServer, createWebServer, or createGenericServer wsadmin command, or you can use the administrative console.

If you are migrating from a previous version of the product, you can upgrade a portion of the nodes in a cell, while leaving others at the previous product level. This means that, for a period of time, you might be managing servers that are running at two different release levels in the same cell. However, when you create a new server definition, you must use a server configuration template, and that template must be created from a server instance that matches the version of the node for which you are creating the server.

There are no restrictions on what you can do with the servers running on the more current release level.

gotcha: If you are using a global resource serialization (GRS) ring to attach one or more monoplexes to a sysplex environment, the cell name of any servers running in any of the monoplexes must be unique within the entire GRS environment. This requirement means that the cell name of a server running in any of the monoplexes:

- Must be different than the cell name of any servers running in the sysplex
- Must be different than the cell name of any servers running in another monoplex that is attached to the sysplex

If you have servers with duplicate cell names within the GRS environment, WebSphere Application Server cannot differentiate between the sysplex cell and the monoplex cell, and treats both servers as part of the same cell, This inaccurate cell association typically causes unpredictable processing results.

Note: If you use additional servers with unique ports, WebSphere Application Server does not automatically configure the virtual host for the server. Specifically, WebSphere Application Server does not automatically add the host alias ports to a virtual host. However, you can use the administrative console to add a new host alias for each of the ports that are used by the new server. For more information, see the documentation about configuring virtual hosts.

Complete the following steps if you want to use the administrative console to create a new application server that is not part of a cluster.

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > New**.

   The Create a new application server wizard starts.
2. Select a node for the application server.
3. Enter a name for the application server. The name must be unique within the node.
4. Click **Next**.
5. Select a server template for the new server.

   You can use a default application server template for your new server, or you can use the template that is optimized for development uses. The new application server inherits all of the configuration settings of the template server.

6. Click **Next**.

   By default, this option is enabled. If you select this option, then you might need to update the alias list for the virtual host that you plan to use with this server to contain these new port values. If you deselect this option, then ensure that the default port values do not conflict with other servers on the same physical machine.

7. Select **Generate unique HTTP ports** if you want the wizard to generate unique ports for the application server.

8. Optional: Click **Next** and specify a short name for the server.

   The short name is also used as the JOBNAME for the server. If you do not specify a value for the short name field, the short name defaults to BBOSnnn, where nnn is the first free number in the cell that can be used to create a unique short name. For example, if default short names are already assigned to two other servers in the cell, the short name BBOS003 will be assigned to this server if you do not specify a short name when you create this server

   gotcha: Make sure that you set up a RACF SERVER class profile that includes this short name.

9. Optional: Specify a generic short name for the server.

   The generic short name for the server becomes the cluster transition name. If you do not specify a value for the generic short name field, the generic short name defaults to BBOCnnn, where nnn is the first free number in the cell that can be used to create a unique generic short name. For example, if default generic short names are already assigned to three other servers in the cell, the generic short name BBOC004 is assigned to this server if you do not specify a generic short name when you create this server.

   gotcha: Make sure that you set up a RACF SERVER class profile that includes this generic short name.

10. Click **Next**. Review the settings for the new server.

11. If you want to change any of the settings, click **Previous** until you return to a page where you can change that setting.

12. Click **Finish** when you do not want to make any additional changes.

13. Click **Review**, select **Synchronize changes with nodes**, and then click **Save** to save your changes.

14. Optional: Run the updateZOSStartArgs script to enable an application server to use the z/OS reusable ASID function, if it is not already enabled for the node that is associated with this application server.

    This function enables an application server to reuse all ASIDs, including those that are associated with cross-process services.

    gotcha: Before running this script, verify that you are running on z/OS Version 1.9 or higher, and that the reuse ASID function is enabled during the z/OS startup process. If the function is not enabled on z/OS, running this script has no affect on how ASIDs are handled.

Results

The new application server is in the list of servers on the administrative console Application servers page.

What to do next

This newly created application server is configured with default settings that are not displayed when you run the Create New Application Server wizard.

You can:

- In the administrative console, click **Servers > Server Types > WebSphere application servers** , and then click the name of this application server to view all of the configuration settings for this application server. You can then use this page to change some of the configuration settings for this server.

For example, if you do not need to have all of the sever components start during the server startup process, you might want to select **Start components as needed**, which is not automatically selected when a new server is created. When this property is selected, server components are dynamically started as they are needed. When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

gotcha: If you are running other WebSphere products on top of this product, make sure that those other products support this functionality before you select this property.

depfeat: The default addressing mode for a new server is 64-bit. You can deselect the Run in 64 bit mode field if you need to use 31-bit addressing mode. However, support for running a server in 31-bit mode is deprecated.

- Use server custom properties to modify the timer settings if you need to change the default timer settings for certain operations.
- Set the client.encoding.override Java virtual machine (JVM) argument to `UTF-8` if you need to use multiple language encoding support in the administrative console.

Application server naming conventions

Before you install a new WebSphere Application Server for z/OS environment, it is important to carefully plan your naming convention. Your naming convention should be able to grow with your system when you increase the number of cells, nodes, servers, and clusters. It should also be able to accommodate Sysplex and LPAR names, as well as instances such as test, integration, and production stages in your environment.

Application servers are like IMS or CICS regions.

- They contain tailored procedures for the controllers and servants.
- They contain tailored environmental variables for each instance of a server.
- They use WLM Classification of regions, working within the regions, and are defined as application environments.
- They may be self-contained or dependent on other servers.
- They need RACF definitions for Control and Server STC (user IDs, resource profiles), as well as UNIX permissions.
- Their users must be allowed to access the servers and to use various objects within them.

gotcha: If you are using a global resource serialization (GRS) ring to attach one or more monoplexes to a sysplex environment, the cell name of any servers running in any of the monoplexes must be unique within the entire GRS environment. This requirement means that the cell name of a server running in any of the monoplexes:

- Must be different than the cell name of any servers running in the sysplex
- Must be different than the cell name of any servers running in another monoplex that is attached to the sysplex

If you have servers with duplicate cell names within the GRS environment, WebSphere Application Server cannot differentiate between the sysplex cell and the monoplex cell, and treats both servers as part of the same cell, This inaccurate cell association typically causes unpredictable processing results.

The product environment consists of a number of address spaces which require the installation to manage security profiles, workload classification constructs, and so on. To create, manage, and recognize application servers, it may be helpful to create a template for naming your servers and server instances. You can find an example template in the *Installing your application serving environment* PDF.

It is also important to plan the naming conventions for your data sets carefully.

- SMP/E target data sets, depending on your maintenance process (regular data sets and the HFS, including its mount points)
- Customization HFS, including its mount point
- HLQ for your customization data sets (*.CNTL, *.DATA, and *.SAVDCFG)
- Error logstream names
- DB2 collection and package names

Creating server templates

A server template is used to define the configuration settings for a new application server. When you create a new application server, you either select the default server template or a template you previously created, that is based on another, already existing application server. The default template is used if you do not specify a different template when you create the server.

About this task

You can also use the **createApplicationServerTemplate** command for the AdminTask object to create a server template.

If you do not create any additional server templates, the defaultZOS template is used as the template for the first cluster member. This template uses the default port assignments for the z/OS platform. If some of these ports are already defined for use elsewhere in your z/OS system, your newly created cluster member might not start, might function incorrectly, or might generate unexpected error messages. Therefore, you must resolve any port conflicts before you start this server.

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere application servers >** , and then click **Templates**.
2. On the Server templates page, click **New**.
3. From the list of servers, select the server that you want to use to create the new template, and then click **OK**.
4. Enter the name of the new template and, optionally, a description of that template that distinguishes it from your other templates.
5. click **OK**.

Results

Your new template is on the list of server templates that you can use to create a new application server or cluster member.

What to do next

You can perform one of the following actions to display a list of all of the server templates that are available on your system:

- In the administrative console, click **Servers > Server Types > WebSphere application servers >** , and then click **Templates**.
- Issue the **listServerTemplates** wsadmin command.

Server template options

Use this page to select the server that you want used to create a new server template. Server templates are copies of server configuration data that can be used as a starting point for creating new servers.

To view this page, in the administrative console, click **Servers > Server Types > WebSphere application servers > Templates > New**.

On this page, you can select the server from which you want to create the new template, and then click **OK**.

Server:

Specifies the name of a server that is already defined on your system.

Node:

Specifies the node on which the server resides.

Version:

Specifies the version of the product to which the template applies.

Server templates collection

Use this page to view the list of server templates that exist on your system.

To view the application server templates that are available for creating a new application server, in the administrative console, click **Servers > Server Types > WebSphere application servers > Templates**. A list of the application server templates that are defined for you system displays.

To view the web server templates that are available for creating a new web server, in the administrative console, click **Servers > Server Types > Web servers > Templates**. A list of the web server templates that are defined for you system displays.

To change the server attributes of an existing template, click the name of that template in the list, and then update the attributes for that template. If you modify a template, all new cluster members are created with the server property settings of the modified template. However, the property settings of existing cluster members do not change. If you make any change to a cluster member template, you should make the same change to all of the existing cluster members.

To delete an existing template, select that template, and then click **Delete**.

Name:

Specifies the name of the server template.

Platform:

Specifies the operating system platform to which this template applies.

Version:

Specifies the version of the product to which the template applies.

Description:

Specifies a description of the template. This field is optional and might be blank.

Server template Settings

Use this page to specify a name and description for the new server template that you are creating. The new template is created based on the configuration of the selected server.

This page displays after you select the server from which you want to create the template, and click **OK**.

Server Name:

Specifies the name of the server that you are using to create this template. This field is read-only.

Name:

Specifies the name of this template.

Description:

Specifies a description of this new server template. If you have multiple templates defined for your system, use this field to help distinguish this template from the other templates.

Deleting server templates

The steps below describe how to delete a server template that you no longer need.

Procedure

Use the **deleteServerTemplate** wsadmin command.

Results

The template you chose is removed from the list and cannot be used to create a new application server.

What to do next

You can perform one of the following actions to display a list of the server templates that are still available on your system:
1. In the administrative console, click **Servers > Server Types > WebSphere application servers**, and then click **Templates**.
2. Issue the **listServerTemplates** wsadmin command.

Configuring an application server to use the WLM even distribution of HTTP requests function

By configuring your application server to use the WLM even distribution of HTTP requests function, HTTP session objects can be evenly distributed by workload management (WLM) to the servants in your configuration. You can use this task to distribute HTTP session objects in a round-robin manner among several servants instead of the normal situation where there is a servant affinity, and HTTP session objects reside in one or two servants.

Before you begin

Your application server should be running on a z/OS system that is at Version 1.4 or later. Because you are distributing HTTP requests among multiple servants in this task, you should also have multiple servants enabled to use this function. See the topic about enabling multiple servants on z/OS for more information.

About this task

Use this task if your application server is experiencing problems with the default workload distribution strategy. The default workload distribution strategy uses a hot servant for running requests that create

HTTP session objects. Consider configuring the product and the z/OS Workload Manager to distribute your HTTP session objects in a round-robin manner in the following conditions:

- HTTP session objects in memory are used, causing dispatching affinities.
- The HTTP sessions in memory last for many hours or days.
- A large number of clients with HTTP session objects must be kept in memory.
- The loss of a session object is disruptive to the client or server.
- There is a large amount of time between requests that create HTTP sessions.

For more background about when to use this task, see the information about WLM even distribution of HTTP requests.

Procedure

1. In the administrative console, set the WLMStatefulSession property to `true`.
   a. Expand **Servers** > **Server Types** and click **WebSphere application servers.**
   b. Click the name of the server that you want to use the WLM even distribution of HTTP requests function.
   c. Under Server Infrastructure, expand **Administration** and click **Administration services**.
   d. Under Additional properties, click **Custom properties**.
   e. Click **WLMStatefulSession** and change the value in the Value field to `true` if it is currently set to `false`. If the custom property does not exist, click **New**, add `WLMStatefulSession` to the **Name** field, and add `true` to the **Value** field.
   f. Click **Apply** and then click **Save** to save your changes to the master configuration.

2. Set the optimal minimum and maximum number of servants for the workload. Set the minimum and maximum number of servants to handle the expected number of HTTP sessions with affinity. The minimum number of servants should be greater than one. If, for example, you expect 15,000 HTTP session objects are established in the server during the day, then you might set the minimum number of servants to some value larger than one. The minimum of servants is dependent upon the size and number of the HTTP session objects. However, the initial arrival rate of client requests establishing the affinity, the frequency of client interaction, the duration of each client interaction (CPU time and thread occupancy time), and the length of time that the HTTP session object is maintained also need to be considered when establishing the minimum value for the number of servants.
   a. To set the number of servants, click **Servers > Server Types > WebSphere application servers** *server_name* **Server instance**.
   b. Set the minimum and maximum number of servants.
   c. Click **Save and synchronize** to apply the changes.

3. If you use a classification mapping file instead of a common workload classification document, and you specify more than one transaction class on a mapping rule for the managed round robin support that the product provides, you should remove this section from your classification mapping file. You should use a common workload classification document instead of a classification mapping file because support for the classification mapping file is deprecated. However if you use a classification mapping file, and that file contains a line similar to the following:

   ```
   TransClassMap *:8080 /Dynacache1Web1/Servlet1 TCLASS1 TCLASS2 TCLASS3
   ```

   Modify this line such that it specifies only one transaction class. For example, you might change the preceding line to the following line:

   ```
   TransClassMap *:8080 /Dynacache1Web1/Servlet1 TCLASS1
   ```

   You also must update the z/OS workload manager policy to remove the extra service classes that are only required fi you want to use the managed round robin support that the product provides. Following is an example of how to remove the extra service classes:

```
 Subsystem-Type  Xref  Notes  Options  Help
 ------------------------------------------------------------------------
                 Modify Rules for the Subsystem Type     Row 9 to 16 of 16
 Command ===> _____   SCROLL ===> CSR

 Subsystem Type . : CB        Fold qualifier names?   Y  (Y or N)
 Description  . . . Component Broker requests

 Action codes:   A=After      C=Copy       M=Move       I=Insert rule
                 B=Before     D=Delete row R=Repeat      IS=Insert Sub-rule
                                                                More ===>
          --------Qualifier--------            -------Class--------
 Action    Type      Name     Start            Service     Report
                                      DEFAULTS: AZAMS1      RBBDEFLT
 ____   1  CN      AZSR01    ___               AZAMS1      RAZAMS1
 ____   2   TC       TCLASS1  ___              AZAMS1      RAZAMS1
 _d__   2   TC       TCLASS2  ___              AZAMS2      RAZAMS1
 _d__   2   TC       TCLASS3  ___              AZAMS3      RAZAMS1
 ____   1  CN      AZSR02    ___               AZAMS2      RAZAMS2
 ____   1  CN      AZSR02    ___               AZAMS3      RAZAMS3
 **************************** BOTTOM OF DATA ****************************
```

4. Restart the server. The server recognizes the `WLMStatefulSession` property after it is restarted.

Results

The application server uses the WLM even distribution of HTTP requests function to handle its workload instead of showing affinity to a certain servant.

What to do next

See the topic about detecting and handling problems with runtime components for information on how to handle problems with server clusters and workloads.

WLM even distribution of HTTP requests

The z/OS workload management (WLM) component supports distributing incoming HTTP requests without servant affinity in a round robin manner across the servants. This functionality is intended for, but not limited to long lasting HTTP session objects that are maintained in memory, stateless session Enterprise JavaBeans (EJB), and the create method for stateful session enterprise beans. You can configure the product to use this functionality to spread HTTP requests among active servants that are currently bound to the same work queue as the inbound requests.

Background

The following diagram represents one clustered server instance. The azsr01 cluster contains the azsr01a application server instance. In the application server instance is a controller, the workload manager (WLM) queue, and the servants where applications run. The controller is the HTTP and IIOP termination point. The WLM queue controls the flow of work from the controller to one of the servants. Each of the servants contains worker threads that select work from the WLM queue.

Figure 1. The contents of one clustered server instance

In the preceding diagram, the application server is configured to have the minimum and maximum number of servants set to three.

There are WLM definitions for the application servers in this cluster. All of the requests for any application server instance in the azsr01 cluster are assigned to the same service class. The WLM classification rules assign all enclaves that are running in the azsr01a application server to the AZAMS1 service class. See

the following diagrams for an example of the WLM service class definition and the classification rules.

```
   Service-Class  Xref  Notes  Options  Help
 --------------------------------------------------------------------------
                         Modify a Service Class            Row 1 to 2 of 2
 Command ===> _____

 Service Class Name . . . . . : AZAMS1
 Description  . . . . . . . . . WAS Enclave Work
 Workload Name  . . . . . . . . ONL_WKL   (name or ?)
 Base Resource Group  . . . . . _____   (name or ?)
 Cpu Critical . . . . . . . . . NO        (YES or NO)

 Specify BASE GOAL information.  Action Codes: I=Insert new period,
 E=Edit period, D=Delete period.


         ---Period---  --------------------Goal---------------------
 Action  #  Duration   Imp.  Description

    __    1             1     Execution velocity of 50
 ***************************** Bottom of data *******************************
```

Figure 2. The WLM service class definition

```
   Subsystem-Type  Xref  Notes  Options  Help
 --------------------------------------------------------------------------
                 Modify Rules for the Subsystem Type    Row 11 to 20 of 20
 Command ===> _____    SCROLL ===> CSR

 Subsystem Type . : CB         Fold qualifier names?   Y  (Y or N)
 Description  . . . Component Broker requests

 Action codes:   A=After     C=Copy       M=Move      I=Insert rule
                 B=Before    D=Delete row  R=Repeat    IS=Insert Sub-rule
                                                           More ===>
          --------Qualifier--------            -------Class--------
 Action    Type    Name    Start               Service    Report
                                       DEFAULTS: AZAMS1    RBBDEFLT

   ____  1  CN      AZSR01   __                 AZAMS1     RAZAMS1
   ____  1  CN      AZSR02   __                 AZAMS2     RAZAMS2
   ____  1  CN      AZSR03   __                 AZAMS3     RAZAMS3
 ***************************** BOTTOM OF DATA ******************************
```

Figure 3. The WLM CB subsystem classification rules

The *hot* servant strategy

The product supports the use of HTTP session objects in memory for application servers with multiple servants. In the following diagram, two users accessed an application in the azsr01a application server instance. User 1 established an HTTP session object in servant 3. User 2 established an HTTP session object in servant 2.

Figure 4. Users establish HTTP session objects

When a user accesses a servant region without an established HTTP session object , no servant region affinity exists. Therefore, the request can be dispatched to any servant that is available. WLM might start a new servant if all of the following conditions exist:
- The configuration allows creating new servants
- The workload manager logic determines that the system can sustain an additional servant

- Adding another servant leads to reduced queue delay and allows enclaves to be completed within the specified goal

When multiple servants are bound to the same service class, WLM attempts to dispatch the new requests to a *hot* servant. A *hot* servant has a recent request dispatched to it and has threads available. If the *hot* servant has a backlog of work, WLM dispatches the work to another servant.

Normally running this *hot* servant strategy is good because the *hot* servant likely has all its necessary pages in storage, has the just-in-time (JIT) compiled application methods saved close by, and has a cache full of data for fast data retrieval. However, this strategy presents a problem in the following situations:
- HTTP session objects in memory are used, causing dispatching affinities.
- The HTTP session objects last for many hours or days.
- A large number of clients with HTTP session objects that must be kept in memory.
- The loss of a session object is disruptive to the client or server and the amount of time between requests that create HTTP sessions is large.

In the last situation, an undesired skew in the distribution of HTTP session objects is the result. In the following diagram, most of the HTTP session objects were assigned to servant 1.

Figure 5. HTTP Session objects assigned to a hot servant

A large percentage of HTTP session objects reside in one or two servants because most of the time, there are not enough requests in the WLM queue to warrant dispatching work among many servants. This behavior can lead to the following undesirable results.
- If the application creates a large number of objects in a single servant, long garbage collection times might result.
- If all the HTTP session objects are bound to one servant, requests might be held in the queue for a long time because the work cannot be managed by WLM and cannot be dispatched in any servant.
- If all HTTP session objects reside in one or two servants, a timeout in a single servant can affect a larger number of users than if the HTTP session objects are divided equally among several servants.

Distribute incoming HTTP requests without servant affinity

If your configuration experiences one of the described situations that cause a problem with the *hot* servant strategy, you can configure your application server to support the distribution of incoming HTTP requests across servants without servant affinity. When you enable this functionality, the application server uses a round-robin distribution of HTTP requests to the servants.

In the following example, assume that the application server was configured to use the round-robin distribution of HTTP requests among the servants and multiple servants are started for the work queue requests that have the same service class assigned.

When a new HTTP request without affinity arrives on a work queue, the WLM checks to see if there is a servant that has at least one worker thread waiting for work. If there are no available worker threads in any servants, WLM queues the request until a worker thread in any of the servants becomes available. If there are available worker threads, WLM finds the servant with the smallest number of affinities. If there are servant regions with equal number of affinities, then WLM dispatches the work to the servant region with the smaller number of busy server threads.

The goal of this algorithm is for WLM to balance the incoming requests without servant affinity among waiting servants while considering changing conditions. The algorithm does not blindly assign requests to servers in a true round-robin manner. The following diagram shows the balanced distribution of HTTP session objects across servants.

Figure 6. HTTP Session objects assigned to servants without affinity

This distribution mechanism works for all inbound requests without affinity. After the HTTP session object is created, all the client requests are directed to that servant until the HTTP session object is removed.

If you decide to enable the distribution of incoming HTTP requests without servant affinity, you might need to make some changes to your classification mapping file. If you have set up your classification mapping file to specify more than one transaction class on a mapping rule for the managed round robin support that the product provides, you should remove this section from your classification mapping file.

Managing application servers

You can use either the administrative console or command-line tools to manage your application servers.

Before you begin

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

About this task

transition: If you are migrating from a previous version of the product, you can upgrade a portion of the nodes in a cell, while leaving others at the previous release level. This means that, for a period of time, you might be managing servers that are running at different release levels in the same cell. In this mixed environment, some restrictions exist for what you can do with servers that are running at a previous release level. No restrictions exist for what you can do with the servers that are running on the newest release level.

You can perform the following steps to view and manage an application server from the administrative console.

Procedure

1. In the administrative console click **Servers** > **Server Types** > **WebSphere application servers**.

   The Application servers page lists the application servers in your environment and the status of each of these servers. You can use this page to complete the following actions:
   - Create additional servers.
   - Monitor running servers.
   - Control the status of a server.
   - Create a server template
   - Delete a server. When you select a server for deletion, you must click **Delete** and **OK** before the server is deleted.

     Tip: If the server you are deleting has applications or modules mapped to it and is not part of a cluster, remap the modules to another server, or create a new server and remap the modules to the new server, before you delete this server. After a server to which modules are mapped is deleted, you cannot remap these modules to another server. Therefore, if you do not remap

the modules to another server before you delete this server, you must uninstall all of the modules that were mapped to this server, and then reinstall them on a different server.

If the server you are deleting is part of a cluster, any application that is installed on this server is automatically installed on all of the other servers in the cluster. Therefore, deleting one cluster member does not affect the other cluster members, and the application remains installed in the cluster. Similarly when a new member is added to an existing cluster, any applications that are installed on the servers in that cluster are automatically installed on the new cluster member.

2. Click the name of a listed server to view or change the configuration settings for that server.

   You can use this administrative console page to:

   - Change the configuration settings for the selected server.

     For example, if you do not need to have all of the sever components start during the server startup process, you might want to select **Start components as needed**, which is not automatically selected when a new server is created. When this property is selected, server components are dynamically started as they are needed. When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

   - View the status of applications running on the selected server. To view the status of applications running on this server, under Applications, click **Installed Applications**.

3. Click **Custom properties** to add new custom properties, update existing custom properties, or modify the timer settings if the current settings are causing timeout problems.

4. Click **Review**, select **Synchronize changes with Nodes**.

5. Click **Save** to save any configuration changes that you made.

6. If you made any configuration or custom property changes, start the application server, or stop and restart the application server if it is already running.

Results

When you click **Servers** > **Server Types** > **WebSphere application servers**, you can view the state of each server.

When you click **Servers** > **Server Types** > **WebSphere application servers** > *server_name*, you can view any configuration changes you made.

What to do next

You can deploy applications or components to your application servers.

Server collection

Use this topic to learn how to navigate within the administrative console to the pages where you can view information about the application servers, generic servers, Java message service (JMS) servers, and web servers that are defined for your system.

You can use these respective administrative console pages to view the status of the listed servers. The status indicates whether a server is running, stopped, or encountering problems. You can also use these pages to perform the following actions for the listed servers:

- Select one or more of the listed servers, and then click **Start** to start those servers.
- Select one or more of the listed servers, and then click one of the following options to stop those servers:

  STOP   When you click this option, the normal server quiesce process is followed. This process allows in-flight requests to complete before the entire server process shuts down.

Immediate Stop

> This option is only available for application servers.
>
> When you click this button, the selected sever stops but the normal server quiesce process is not followed. This shutdown mode is faster than the normal server stop processing, but some application clients might receive exceptions if an in-flight request does not complete before the server process shuts down.

Terminate

> You should only click **Terminate** if the server does not respond when you click **Stop**, or, **Immediate Stop** or when you issue the Stop or ImmediateStop commands. Some application clients can receive exceptions. Therefore, you should always attempt an immediate stop before clicking **Terminate**.

- Click **New** to create a new server.
- Click **Templates** to create a new server template.
- Select one or more of the listed servers, and then click **Delete** those servers.

To view the Application servers page, in the administrative console page, click **Servers** > **Server Types** > **WebSphere application servers**. This page lists all of the application servers in the cell.

To view the Generic servers page, in the administrative console, click **Servers** > **Server Types** > **Generic servers**. This page lists all of the generic servers in the cell.

To view the web servers page, in the administrative console, click **Servers** > **Server Types** > **Web servers**. This page lists all of the web servers in your administrative domain. In addition to the previously mentioned actions, you can use this page to generate and propagate a web server plug-in configuration file.

Name

Specifies a logical name for the server. For WebSphere Application Server, server names must be unique within a node.

On the z/OS platform, this is sometimes called the long name. Server names must be unique within a node. If you have multiple nodes in a cluster, the server names must also be unique within the cluster. You cannot use the same server name within two nodes that are part of the same cluster. WebSphere uses the server name for administrative actions, such as referencing the server in scripting.

Node

Specifies the node on which the server resides.

Host Name

Specifies the IP address, the full domain name system (DNS) host name with a domain name suffix, or the short DNS host name for the server.

Version

Specifies the version of the product on which the server runs.

Cluster Name

Specifies the name of the cluster to which the application server belongs. This field only displays when the **Include cluster members in the collection** console page preference is selected on the Applications server page.

If you have created clusters, and if the **Include cluster members in the collection** console page preference is selected, application servers that are cluster members are included in the list of application servers that displays on the Application servers page. These cluster members can be managed in the same manner as any of the other application servers in the list.

If the **Include cluster members in the collection** console page preference is not selected, application servers that are cluster members are not listed in the list of application servers that can be managed from this page.

Status

Specifies whether the server is started, stopped, partially stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

Table 37. Server status and meaning. The following table describes the server status.

⬌	**Started**	The server is running.
⊗	**Partially stopped**	The server is in the process of changing from a started state to a stopped state.
✖	**Stopped**	The server is not running.
⊘	**Unavailable**	The server status cannot be determined.

Application server settings

Use this page to configure an application server or a cluster member template. An application server is a server that provides services required to run enterprise applications. A cluster member template is the set of application server configuration settings that are assigned to new members of a cluster.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log`, `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name*.

On the **Configuration** tab, you can change field settings. You can also click **Installed applications** to view the status of applications that are running on this server. On the **Runtime** tab, you can view read only information. The **Runtime** tab is available only when the server is running.

Name

Specifies a logical name for the server. Server names must be unique within a node. However, for multiple nodes within a cluster, you might have different servers with the same server name as long as the server and node pair are unique. You cannot change the value that appears in this field.

For example, a server named *server1* in a node named *node1* in the same cluster with a server named *server1* in a node named *node2* is allowed. However, you cannot have two servers named *server1* in the same node. The product uses the server name for administrative actions, such as referencing the server in scripting.

On the z/OS platform, this name is sometimes referred to as the long name.

Default   server1

gotcha: If you are using a global resource serialization (GRS) ring to attach one or more monoplexes to a sysplex environment, the cell name of any servers running in any of the monoplexes must be unique within the entire GRS environment. This requirement means that the cell name of a server running in any of the monoplexes:

- Must be different than the cell name of any servers running in the sysplex
- Must be different than the cell name of any servers running in another monoplex that is attached to the sysplex

If you have servers with duplicate cell names within the GRS environment, WebSphere Application Server cannot differentiate between the sysplex cell and the monoplex cell, and treats both servers as part of the same cell, This inaccurate cell association typically causes unpredictable processing results.

Short name

Specifies the short name of the server and must be unique within a cell. This field only displays for the z/OS platform. The short name is also the default z/OS job name and identifies the server to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), and started task control.

This field is optional, and only displays if you are running on z/OS. If you do not specify a value for the short name field, the short name defaults to BBOSnnn, where nnn is the first free number in the cell that can be used to create a unique short name. For example, if default short names are already assigned to two other servers in the cell, the short name BBOS003 will be assigned to this server if you do not specify a short name when you create this server. After the application server is created, you can change this generated short name to a name that conforms to your naming conventions.

The default values for the servant and adjunct jobnames are this short name with either an S, for the servant, or an A, for the adjunct, appended. If you must use an 8-character server short name, the servant and adjunct jobnames become 9-character names. Therefore, you must update the start command arguments for the servant and the adjunct process definitions to use the new 8-character server short name. The topic "Converting a 7-character server short name to 8 characters" describes how to perform this update.

If you specify a short name, the name:
- Must be one to eight characters in length. By default, z/OS assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only uppercase alphanumeric characters
- Cannot start with a number.
- Must be unique in the cell.

Generic short name

Specifies the generic short name of the server and must be unique within a cell. This field is optional, and only displays for the z/OS platform. The generic short name for the server becomes either the cluster transition name, if you are creating an unclustered server, or the cluster short name, if you are creating a clustered server.

If you do not specify a value for the generic short name field, the generic short name defaults to BBOCnnn, where nnn is the first free number in the cell that can be used to create a unique generic short name. For example, if default generic short names are already assigned to three other servers in the cell, the generic short name BBOC004 is assigned to this server if you do not specify a generic short name when you create this server.

If you specify a generic short name, the name:
- Must be one to eight characters in length.
- Must contain only alpha-numeric or national language characters.

- Cannot start with a number.
- Must be unique in the cell.

Run in development mode

Enabling this option might reduce application server start-up time because it changes some of the JVM settings, such as disabling bytecode verification, and reducing just-in-time (JIT) compiler compilation costs. Do not enable this setting on production servers. This setting is only available on an application server that is running in a Version 6.0 or later cell.

Specifies that you want to use the **-Xverify** and **-Xquickstart** JVM properties as startup values. Before selecting this option, add the **-Xverify** and **-Xquickstart** properties as generic arguments to the JVM configuration.

If you select this option, then you must save the configuration, and restart the server before this configuration change takes effect.

The default setting for this option is `false`, which indicates that the server does not start in development mode. Setting this option to `true` specifies that the server starts in development mode with settings that decrease server start-up time.

Data type	Boolean
Default	false

Parallel start

Select this field to start the server on multiple threads. This might shorten the startup time.

Specifies that you want the server components, services, and applications to start in parallel rather than sequentially.

The default setting for this option is `true`, which indicates that when the server starts, the server components, services, and applications start on multiple threads. Setting this option to `false` specifies that when the server starts, the server components, services, and applications start on a single thread, which might lengthen start-up time.

The order in which the applications start depends on the weights that you assign to them. Applications that have the same weight start in parallel.

To set the weight of an application, in the administrative console, click **Applications > Application Types > WebSphere enterprise applications >** *application_name* **> Startup behavior**, and then specify an appropriate value in the **Startup order** field. The more important an application is, the lower the startup order value should be. For example, you might specify a startup order value of 1 for your most important application, and a value of 2 for the next most important application. You might then specify a startup order of 3 for the next four applications because you want all four of those applications to start in parallel.

Data type	Integer
Default	1
Range	0 - 2147483647

Start components as needed

Select this property if you want the server components started as they are needed by an application that is running on this server.

When this property is selected, server components are dynamically started as they are needed. When this property is not selected, all of the server components are started during the server startup process.

Therefore, selecting this option can improve startup time, and reduce the memory footprint of the server, because fewer components are started during the startup process.

Starting components as they are needed is most effective if all of the applications, that are deployed on the server, are of the same type. For example, using this option works better if all of your applications are web applications that use servlets, and JavaServer Pages (JSP). This option works less effectively if your applications use servlets, JSPs and Enterprise JavaBeans (EJB).

gotcha: To ensure compatibility with other WebSphere products, the default setting for this option is deselected. Before selecting this option, verify that any other WebSphere products, that you are running in conjunction with this product, support this functionality.

Run in 64 bit JVM mode

Specifies that the application server runs in 64-bit mode, which is the default setting. Running in 64-bit mode provides additional virtual storage for user applications. This field only displays for the z/OS platform.

By default, the WebSphere Customization Toolbox is set up to start all of your application servers in 64-bit mode. However, you can change the settings for the WebSphere Customization Toolbox such that all of your application servers start in 31-bit mode. You can also unselect this setting for a subset of your application servers if you only want specific application servers to start in 31-bit mode.

There is no interdependence between the modes in which you are running different servers. Therefore, you can run some of your servers in 64-bit mode and some of your servers in 31-bit mode. However, you should eventually convert all of your servers to run in 64-bit mode because support for running servers in 31-bit mode is deprecated.

Access to internal server classes

Specifies whether the server can run in Restrict or Allow mode.

The Restrict mode is a diagnostic mode that you can use to help determine the suitability of applications for migration. This mode determines whether internal application server classes are accessed. The use of these internal classes might preclude the successful operation of these applications in future releases. However, the Restrict mode is not intended to exclude all classes from general use even if the classes might change. Some classes that might change are unrestricted in order to enable correct operation of the application server. The Restrict mode is not intended to provide complete isolation between an application and application server internal classes. Do not use the Restrict mode in a production runtime environment; use the results for guidance only.

The default value for this property is `Allow`.

Class loader policy

Select whether there is a single class loader to load all applications or a different class loader for each application.

Class loading mode

Specifies whether the class loader searches in the parent class loader or in the application class loader first to load a class. The standard for Developer Kit class loaders and the product class loaders is `Parent first`.

This field only applies if you set the Class loader policy field to `S*ingle`.

If you select `Application first`, your application can override classes contained in the parent class loader, but this action can potentially result in ClassCastException or linkage errors if you have mixed use of overridden classes and non-overridden classes.

Process ID
The process ID for this server on the native operating system.

This property is read only. The system automatically generates the value.

Cell name
The name of the cell in which this server is running.

This property is read only.

Node name
The name of the node in which this server is running.

This property is read only.

State
The runtime start state for this server.

This property is read only.

Product information
This link under Additional properties, displays the product information for your installation of the product. This information includes the product name, ID, version, build date, and build level.

From the Product Information page, you can click on the following links for additional product information:
- Components, for a list of all of the components that are installed.
- e-Fixes, for a list of all of the service updates that are installed.
- Extensions, for a list of the extensions that are installed.
- History report, for a detailed report of all installation events that have occurred since the product was installed, such as the installation of a specific service level.
- Product report, for a detailed report of the versions of the product that are installed.
- PTFs, for a list of all of PTFs that are installed.

Ports collection
Use this page to view and manage communication ports used by run-time components running within a process. Communication ports provide host and port specifications for a server.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Communications > Ports**.

This page displays only when you are working with ports for application servers.

Port Name:

Specifies the name of a port. Each name must be unique within the server.

Host:

Specifies the IP address, domain name server (DNS) host name with domain name suffix, or just the DNS host name, used by a client to request a resource (such as the naming service, or administrative service).

Port:

Specifies the port for which the service is configured to accept client requests. The port value is used in conjunction with the host name.

Transport Details:

Provides a link to the transport chains associated with this port. If no transport chains are associated with this port, the string "No associated transports" appears in this column.

Ports settings:

Use this to view and change the configuration for a communication port used by run-time components running within a process. A communication port provides host and port specifications for a server.

If you are running on z/OS, you can view this administrative console page by clicking one of the following paths:
* **Servers > Server Types > WebSphere application servers >** *server_name* **> Ports >** *end_point_name*
* **Servers > Server Types > JMS servers >** *server_name* **> Ports >** *end_point_name*

Port Name:

Specifies the name of the port. The name must be unique within the server.

Note that this field displays only when you are defining a port for an application server. You can select either:

Well-known Port
> When you select this option , you can select a previously defined port from the drop down list

User-defined Port
> When you select this option, you must create a port with a new name by entering the name of the new port in the text box

Table 38. Data type. The following table describes the data type for the Port Name setting.

Data type	String

The following ports apply to the z/OS platform only.

Table 39. Ports. The following table describes ports for the z/OS platform.

End point	Description
JMSSERVER Queued Address	Specifies the host and port number used to configure the WebSphere JMS Provider topic connection factory. The JMS Server Queued Address port is the listener port used for full-function JMS-compliant, publish/subscribe support. The default Queued Address port number is 5558. Since the queue manager and queue broker for the WebSphere JMS Provider are configured outside the administrative console, changes to this port require corresponding configuration changes to the queue manager and queue broker.

Table 39. Ports (continued).   The following table describes ports for the z/OS platform.

End point	Description
JMSSERVER Direct Address	Specifies the host and port number used to configure the WebSphere JMS Provider topic connection factory. The JMS Server Direct Address port is the listener port used for direct TCP/IP connection (non-transactional, nonpersistent, and nondurable subscriptions only) for publish/subscribe support. The default Direct Address port number is 5559. Since the queue manager and queue broker for the WebSphere JMS Provider are configured outside the administrative console, changes to this port require corresponding configuration changes to the queue manager and queue broker.

Host:

Specifies the IP address, domain name server (DNS) host name with domain name suffix, or just the DNS host name, used by a client to request a resource (such as the naming service, administrative service, or JMS broker).

For example, if the host name is `myhost`, the fully qualified DNS name can be `myhost.myco.com` and the IP address can be `155.123.88.201`.

Host names on the ports can be resolvable names or IP addresses. The server will bind to the specific host name or IP address that is supplied. That port will only be accessible through the IP address that is resolved from the given host name or IP address. The IP address may be of the IPv4 (Internet Protocol Version 4) format for all platforms, and IPv6 (Internet Protocol Version 6) format on specific operating systems where the server supports IPv6.

gotcha: If your TCP/IP network is set up to use distributed dynamic virtual IP addresses (DVIPAs), and if the node agent is in the process of starting the application server, TCP/IP waits until the JVM TCP/IP timeout period expires before notifying the node agent that the target application server is not responsive.

Table 40. Data type and Default.   The following table describes the data type and default for the Host setting.

Data type	String
Default	* (asterisk)

Port:

Specifies the port for which the service is configured to accept client requests. The port value is used in conjunction with the host name.

Port numbers in the server can be reused among multiple ports as long as they have host names that resolve to unique IP addresses and there is not a port with the same port number and a wildcard (*) host name. A port number is valid in the range of 0 and 65535. 0 specifies that the server should bind to any ephemeral port available. Specifying the wildcard value is equivalent to specifying the loopback address or 127.0.0.1.

gotcha: Port sharing cannot be created using the administrative console. If you need to share a port, you must use wsadmin commands to define that port. You must also make sure that the same discrimination weights are defined for all of the transport channels associated with that port.

Protocol channels only accept their own protocol. However, application channels usually accept anything that reaches them. Therefore, for application channels, such as WebContainer, or Proxy, you should specify larger discrimination weights when sharing levels with protocol channels, such as HTTP or SSL. The one exception to this rule is if you have application channels that perform discrimination tests faster than the protocol channels. For example, a JFAP channel is faster at deciding on a request than the SSL protocol channel, and should go first for performance reasons. However, the WebContainer and Proxy channels must always be last because they accept everything that is handed to them.

Table 41. Data type and Default.   The following table describes the data type and default for the Port setting.

Data type	Integer
Default	None

gotcha: The following table lists server endpoints and their respective port ranges. In contrast to an IBM i or distributed platform environment, for a z/OS environment, the ORB_LISTENER_ADDRESS and the BOOTSTRAP_ADDRESS must specify the same port.

Table 42. Server endpoints and their respective port ranges.   The following table lists server endpoints and their respective port ranges.

Endpoint (port)	Acceptable values for the port field
BOOTSTRAP_ADDRESS	0 - 65535
DATAPOWERMGR_INBOUND_SECURE	1 - 65536
DCS_UNICAST_ADDRESS	1 - 65536
ORB_LISTENER_ADDRESS	1 - 65535 (If 0 is specified, the server starts on any available port and does not use the location service daemon)
SIB_ENDPOINT_ADDRESS	1 - 65536
SIB_ENDPOINT_SECURE_ADDRESS	1 - 65536
SIB_MQ_ENDPOINT_ADDRESS	1 - 65536
SIB_MQ_ENDPOINT_SECURE_ ADDRESS	1 - 65536
SOAP_CONNECTOR_ ADDRESS	1 - 65536
WC_adminhost	1 - 65536
WC_adminhost_secure	1 - 65536
WC_defaulthost	1 - 65536
WC_defaulthost_secure	1 - 65536
ORB_SSL_LISTENER_ADDRESS	0 - 65535 (0 specifies that the server should bind to any ephemeral port that is available.)

Custom property collection

Use this page to view and manage arbitrary name-value pairs of data, where the name is a property key and the value is a string value that can be used to set internal system configuration properties.

The administrative console contains several Custom Properties pages that work similarly. To view one of these administrative pages, click one of the **Custom properties** links.

Name:

Specifies the name (or key) for the property.

Each property name must be unique. If the same name is used for multiple properties, the value specified for the first property is used.

Do not start your property names with `was.` because this prefix is reserved for properties that are predefined in the application server.

Value:

Specifies the value paired with the specified name.

Description:

Provides information about the name-value pair.

Custom property settings:

Use this page to configure arbitrary name-value pairs of data, where the name is a property key and the value is a string value that can be used to set internal system configuration properties. Defining a new property enables you to configure a setting beyond that which is available in the administrative console.

depfeat: Setting custom properties at the server level is deprecated. However, you can specify a custom property for a server or the deployment manager as a WebSphere variable. Server scoped WebSphere variables still override any settings specified at the node scope, or higher, and are added to the was.env file.

To set a custom properties for either the deployment manager, or an application server, as an environment variable, in the administrative console, click **Environment > WebSphere variables**. You can then select the deployment manager or appropriate server from the pull-down list of available servers, nodes and cells and click **New** to create a new custom property, click on the name of an existing property to change the settings of that custom property, or click **Delete** to delete an existing property.

Name:

Specifies the name (or key) for the property.

Each property name must be unique. If the same name is used for multiple properties, the value specified for the first property is used.

Do not start your property names with `was.` because this prefix is reserved for properties that are predefined in the product.

Data type	String

Value:

Specifies the value paired with the specified name.

Data type	String

Description:

Provides information about the name and value pair.

Data type	String

Native processes

Use this page to view and modify properties of the JMS Integral Provider native processes.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, in the Server Infrastructure section, click **Administration > Server components > JMS servers**.

Short name:

Specifies the short name of the JMS queue manager.

The name is 1-4 characters, alphanumeric or national language. It cannot start with a numeric.

The system assigns a unique default short name for the JMS queue manager.

Data type	String

Command Prefix:

Specifies the subsystem command prefix for the JMS queue manager.

This field is read only because it is only configured using either the Profile Management Tool, or the zpmt command.

Data type	String

Server component collection

Use this page to view information about and manage the types of server components that a specific application server uses during application processing. The list of server components varies according to the type of applications a specific application server processes.

For example, SIP Container might be listed as a server component for an application server that handles Session Initiation Protocol (SIP) requests, while EJB Container might be listed as a server component for an application server that handles Enterprise JavaBeans (EJB) requests. However, Messaging Server might be listed as a server component for both application servers.

You can also use this page to manage the settings for these server component, as they relate to request processing. In particular, you can specify either started or stopped as the initial state for the server component when the server process starts.

To view this administrative console page, click **System administration > Deployment Manager***server_name*. Then, in the Server Infrastructure section, click **Administration > Server components**.

To view this administrative console page for a node agent, click **System administration > Node agents >** *node_agent_name*. Then, in the Server Infrastructure section, click **Administration > Server components**.

Type:

Specifies the server component type, such as Name Server or Messaging Server.

Server component settings:

Use this page to view or configure a server component instance.

To view this administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, in the Server Infrastructure section, click **Administration > Server components >** *server_component_name*.

Name:

Specifies the name of the component.

Data type	String

Initial State:

Specifies the desired state of the component when the server process starts. The options are: *Started* and *Stopped*. The default is *Started*.

Data type	String
Default	Started

Server instance settings

Use this page to view and manage servant instance settings. These settings control the number of servant processes that are allowed.

To view this page, in the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*. Next, in the Server Infrastructure section, click **Java and process management > Server instance**

Multiple Instances Enabled:

Specifies whether multiple servant process instances are enabled for this server.

When the Multiple Instances Enabled setting is disabled, the server has exactly one servant process instance. When the Multiple Instances Enabled setting is enabled:
- The Minimum number of instances setting determines the minimum number of process instances that are active for this server.
- The Maximum number of instances setting determines the maximum number of process instances that can be active for this server.

The z/OS Workload Manager dynamically determines the actual number of servant process instances.

Minimum number of instances:

Specifies the minimum number of servant process instances to activate.

Data type	Integer
Range	1 to 20, inclusive

Maximum number of instances:

Specifies the maximum number of servant process instances to activate.

Data type	Integer
Range	Any value. If zero is specified, the number of instances is unlimited.

Core group service settings

Use this page to set up the application server properties that relate to core groups.

To view this administrative console page, click **Servers > Server Types > Application servers >** *server*. Then in the Additional Properties section, select **Core group service**.

Click **Save** to save and synchronize your changes with all managed nodes.

Enable service at server startup

Select if you want the core group service, also known as the high availability manager service, to start on this process when the server starts. The core group service must be started before high availability functions, such as routing, and failover, work properly.

gotcha: The default value for this setting is selected Before disabling the core group service for a server process, make sure that none of the components that this process uses require high availability functions.

Default	Core group service starts when the server starts.

Core group name

Specifies the name of the core group that contains this application server as a member. To move a server to a different core group, in the administrative console, click **Servers > Core groups > Core group settings >** *core_group* **> Core group servers**.

Data type	String

Allow activation

Select if high availability group members can be activated on this application server.

Is alive timer

Specifies the time interval, in seconds, at which the high availability manager will check the health of all of the active high availability group members that are running in this application server process. An active group member is a member that is able to accept work. If a group member fails, the application server on which the group member resides is restarted. If -1 is specified, the timer is disabled. If 0 (zero) is specified, the default value of 120 seconds is used.

gotcha: The value specified for this property can be overridden for the high availability groups using a particular policy if the Is alive timer property for that policy specifies a different time interval. If the Is alive timer setting specified for a policy is greater than 0 (zero), the high availability manager uses that time interval, instead of the one specified at this level, when determining how frequently it should check the health of a high availability group member using that particular policy.

Data type	Any integer between -1 and 600, inclusive
Default	120 seconds

Transport buffer size

Specifies the buffer size, in megabytes, of the underlying group communication transport. The minimum buffer size is 10 megabytes.

Data type	String
Default	10 megabytes

Switching between 64 and 31 bit modes

When you create a new application server, it is automatically configured to run in 64-bit mode. You can deselect the **Run in 64 bit JVM mode** setting if you need to run the server in 31-bit mode. Whenever possible, however, leave your servers configured to run in 64-bit mode because support for running servers in 31-bit mode is deprecated. If you have any servers, that you migrated from a previous version of the product, that are running in 31-bit mode, consider reconfiguring them to run in 64-bit mode.

depfeat: Because support for running a server in 31-bit mode is deprecated, whenever you start a server that is configured to run in 31-bit mode, you receive the following message in your system log, where *server_name* is the name of the server that is running in 31-bit mode:

```
BBOO0340W: 31-BIT MODE IS DEPRECATED FOR THE APPLICATION SERVER RUNNING ON THE Z/OS OPERATING SYSTEM.
CONSIDER USING 64-BIT MODE FOR server_name AS AN ALTERNATIVE.
```

When a server runs in 31-bit mode, the following conditions exist:
- Each server address space can access a maximum of 2 gigabytes of virtual memory.
- The 31-bit JVM, that is located in the *app_server_root*/`java` directory, is used for the server.

When a server runs in 64-bit mode, the following conditions exist:
- Each server address space can access a maximum of 16 exabytes (16 thousand million gigabytes) of virtual memory.
- The 64-bit JVM, that is located in the *app_server_root*/`java64` directory, is used for the server.

System requirements

Before starting to use a server that is configured to run in 64-bit mode, note that:
- The AMODE parameter can be used to specify a particular addressing mode, either 31-bit or 64-bit, for the server. This parameter can also be set to a value of `00`, which indicates that the server is to use the configured addressing mode. In the generated procedures, 00 is the default value for the AMODE parameter.

  If you convert a 31-bit server to the 64-bit addressing mode, make sure that your automation does not specify AMODE=31 on the MVS START command. If a server is started with an AMODE value that does not match the configured addressing mode, the server does not start.

  gotcha: If the AMODE parameter is omitted, or set to 00 on the MVS START command, then the server starts in the currently configured addressing mode. If any other AMODE parameter is specified, that parameter must match the currently configured addressing mode. If the parameter does not match the currently configured addressing mode, the server terminates with one of the following error messages:

```
BBOO0336E START OF WEBSPHERE FOR Z/OS PROCESS FAILED BECAUSE INPUT
AMODE 31 DOES NOT MATCH CONFIGURED AMODE 64
```

```
BBOO0336E START OF WEBSPHERE FOR Z/OS PROCESS FAILED BECAUSE INPUT
AMODE 64 DOES NOT MATCH CONFIGURED AMODE 31
```

- The REGION parameter on the JCL JOB or EXEC statement, and the MEMLIMIT parameter on the JCL JOB or EXEC statement, or in the SMFPRMxx parmlib member, determines the amount of virtual storage that a particular server can obtain. If you do not specify REGION=0M in the server cataloged procedure, you must use the MEMLIMIT parameter to set a virtual storage limit larger than the 2 gigabyte limit associated with 31-bit mode.

  gotcha: Make sure that your system exits do not limit the address space size for 64-bit servers inappropriately.
- Running servers in 64-bit mode requires additional auxiliary storage, in the form of either expanded storage or page data set space. Before running servers in 64-bit mode, review your page data set

allocations for each z/OS target system, and add additional page data set space as needed. Also monitor paging and page data set utilization to ensure that the allocated auxiliary storage is sufficient.

For more information, see the following z/OS publications:
- *MVS Initialization and Tuning Reference*, SA22-7592
- *MVS Programming: Extended Addressability Guide*, SA22-7614

Converting a migrated server to run in 64-bit mode

Before converting an application server from 31-bit to 64-bit mode, complete the following actions:
- Verify that your system meets the requirements specified in the Systems requirements section.
- Verify that all applications that you plan to run on this application server are updated to use 64-bit native code and DLLs.

  The DLLs and other native code that your applications call must match the addressing mode of the server on which the applications are running. If you convert an existing application server from 31-bit mode to 64-bit mode, you must change any Java applications containing native code, for example, C++ or Cobol, that you plan to run on the converted server, to run in 64-bit mode. Java applications can use the com.ibm.vm.bitmode Java property to determine the mode in which the server is running, and then load the correct 31-bit or 64-bit DLL to the native code.

  An abend might occur if a server that is running in 64-bit mode tries to invoke an application that contains a 31-bit native module. Similarly, An abend might occur if a server that is running in 31-bit mode tries to invoke an application that contains a 64-bit native module.

  For more information about converting language-environment (LE) applications to run in 64-bit mode, see the z/OS publication *Language Environment Programming Guide for 64-bit Virtual Addressing Mode*, SA22-7569.

To convert an application server from 31-bit mode to 64-bit mode, in the administrative console select the **Run in 64 bit JVM mode** option on the configuration settings page for that application server, and change the minimum and maximum JVM heap sizes to values that are appropriate for a 64-bit process. Similarly to convert an application server from 64-bit mode to 31-bit mode, deselect the **Run in 64 bit JVM mode** option on the configuration settings page for that application server, and change the minimum and maximum JVM heap sizes to values that are appropriate for a 31-bit process.

If you use the MVS START command to start a 64-bit server, make sure that the AMODE parameter is set to 00 or 64, or is allowed to default to 00, on the START command. For example, you might issue one of the following commands:

```
S BBO7ACR,JOBNAME=BBOS001,ENV=BBOBASE.BBONODE.BBOS001,AMODE=64
```

```
S BBO7ACR,JOBNAME=BBOS001,ENV=BBOBASE.BBONODE.BBOS001
```

The startServer.sh command, and the administrative console, automatically add the AMODE=64 parameter when they are used to start a 64-bit application server.

Converting a migrated deployment manager to run in 64-bit mode

Before converting a deployment manager from 31-bit to 64-bit mode, complete the following actions:
- Verify that your system meets the requirements specified in the Systems requirements section.
- Verify that all applications that you plan to run on the deployment manager are updated to use 64-bit native code and DLLs.

  The DLLs and other native code that your applications call must match the addressing mode of the server on which the applications are running. If you convert an existing application server from 31-bit mode to 64-bit mode, you must change any Java applications containing native code, for example, C++ or Cobol, that you plan to run on the converted server, to run in 64-bit mode. Java applications can use

the com.ibm.vm.bitmode Java property to determine the mode in which the server is running, and then load the correct 31-bit or 64-bit DLL to the native code.

An abend might occur if a server that is running in 64-bit mode tries to invoke an application that contains a 31-bit native module. Similarly, An abend might occur if a server that is running in 31-bit mode tries to invoke an application that contains a 64-bit native module.

For more information about converting language-environment (LE) applications to run in 64-bit mode, see the z/OS publication *Language Environment Programming Guide for 64-bit Virtual Addressing Mode*, SA22-7569.

To convert a deployment manager from 31-bit mode to 64-bit mode, go to **Servers > Server types > WebSphere application servers > *server_name*** in the administrative console and select the **Run in 64 bit JVM mode** option on the configuration settings page for the deployment manager as well as change the minimum and maximum JVM heap sizes to values that are appropriate for a 64-bit process.

If you use the MVS START command to start a 64-bit deployment manager, make sure that the AMODE parameter is set to 00 or 64, or is allowed to default to 00, on the START command. For example, you might issue one of the following commands:

```
S BBO7DCR,JOBNAME=BBODMGR,ENV=PLEXA.PLEXABBOCELL.BBODMGR.BBODMGR,AMODE=64
S BBO7DCR,JOBNAME=BBODMGR,ENV=BBOCELL.BBODMGR.BBODMGR
```

The startServer.sh command and the administrative console automatically add the AMODE=64 parameter when they are used to start a 64-bit deployment manager.

Converting a server to run in 31-bit mode

Before converting a server from 64-bit to 31-bit mode, verify that all applications that you plan to run on the server use 31-bit native code and DLLs.

To convert a server from 64-bit mode to 31-bit mode, complete the following actions:

- Start the server.

  The server tells you what mode it is in when it starts:

  ```
  BBO00309I CONTROL PROCESS BBOBASE/BBONODE/BBOC001/BBOS001 IS EXECUTING IN 64-BIT ADDRESSING MODE
  ```

- Go to **Servers > Server types > WebSphere application servers > *server_name*** in the administrative console.

- Deselect the **Run in 64 bit JVM mode** option on the configuration settings page for the server.

- Restart the server.

  Now, the following messages appear on the console and in the server job log:

  ```
  09.31.42 STC00104  BBO00309I CONTROL PROCESS  BBOBASE/BBONODE/BBOC001/BBOS001 IS EXECUTING
    IN 31-BIT ADDRESSING MODE.
  09.31.42 STC00104  BBO00340W 31-BIT MODE IS DEPRECATED FROM THE APPLICATION SERVER ON THE
    Z/OS OPERATING SYSTEM. CONSIDER USING 64-BIT MODE FOR BBOS001 AS AN ALTERNATIVE.
  ```

Changing the values of variables referenced in BBOM0001I messages

BBOM0001I messages are issued during server startup, and indicate the product configuration settings. Unless otherwise indicated, these variables apply to all application servers, including the deployment managers, node agents, and JMS servers.

Before you begin

Not all of these settings can be changed. However, the settings that you can change must be changed using the administrative console. Any changes that you make directly to the was.env file for a specific server are discarded the next time that you use the administrative console to make a configuration change.

Use the following table to determine which product variable, custom property or administrative console field must be updated to change the value of a specific internal variable.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields. The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

Internal Variable Name	How to Change Indicated Value	Comments
adjunct_region_libpath (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Adjunct > Environment entries > LIBPATH**, and specify a different value.	Specifies the libpath for the adjunct process' JVM.
cell_name	User cannot change.	Initially specified during installation and customization.
cell_short_name	User cannot change.	Initially specified during installation and customization.
client_protocol_resolve_ name	User cannot change.	No longer used. Message will not appear for V5.1 and higher.
client_protocol_ resolve_ port	User cannot change.	No longer used. Message will not appear for V5.1 and higher.
client_ras_logstream_ name	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `client_ras_logstreamname` property and specify a different value.	See the information about z/OS custom properties for the application server.
com_ibm_security_SAF_ EJBROLE_Audit_ Messages_ Suppress	In the administrative console, click **Security > Security administration > Applications > Infrastructure**. In the Additional Properties section, click **Custom properties > New**. Add the `com_ibm_security_SAF_EJBROLE_Audit_Messages _Suppress` property, and specify a different value.	
com_ibm_DAEMON_claim _ssl_sys_v3_timeout	In the administrative console, click**System Administration > Node groups > Sysplex node group > z/OS location service**.	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_ claimClientAuthentication	In the administrative console, click**System Administration > Node groups > Sysplex node group > z/OS location service**.	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_ claimKeyringName	In the administrative console, click**System Administration > Node groups > Sysplex node group > z/OS location service**.	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_ claimSecurityCipherSuite List	In the administrative console, click**System Administration > Node groups > Sysplex node group > z/OS location service**.	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_ claimSecurityLevel	In the administrative console, click**System Administration > Node groups > Sysplex node group > z/OS location service**.	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_HTTP_claim_ ssl_sys_v2_timeout	User cannot change.	**depfeat:** This variable has been deprecated.
com_ibm_HTTP_claim_ ssl_sys_v3_timeout	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name* > **Web container settings > Web container > HTTP transports >** *,ssl_transport*.	Transport option only appears if you have an HTTP transport defined for your system.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued).   The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

com_ibm_HTTP_claim_sslEnabled	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Web container settings > Web container > HTTP transports**,*ssl_transport*.	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_HTTP_claimClientAuthentication	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Web container settings > Web container > HTTP transports**.	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_HTTP_claimKeyringName	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Web container settings > Web container > HTTP transports**.	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_HTTP_claimSecurityCipherSuiteList	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Web container settings > Web container > HTTP transports**.	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_HTTP_claimSecurityLevel	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Web container settings > Web container > HTTP transports**.	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_Server_Security_Enabled	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Security > Server security**.	Select an existing alias or create a new SSL Configuration Repertoire. This setting overrides the setting specified using the **Security > Global Security > enabled/disabled** field in the administrative console.
control_region_classpath	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Java Virtual Machine**, and specify the classpath to be appended.	Specifies the classpath used by the conroller's JVM.
control_region_http_queue_timeout_percent	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_http_queue_timeout_percent` property and specify a different value.	Indicates the percentage of the HTTP dispatch time limit that should be used as the maximum amount of time that an HTTP request can spend on the workload management (WLM) queue. See the information about z/OS custom properties for the application server.
control_region_https_queue_timeout_percent	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_https_queue_timeout_percent` property and specify a different value.	Indicates the percentage of the HTTPS dispatch time limit that should be used as the maximum amount of time that an HTTPS request can spend on the workload management (WLM) queue. See the information about z/OS custom properties for the application server.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued). The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

control_region_iiop_ queue_timeout_percent	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_iiop_queue_timeout_percent` property and specify a different value.	Indicates the percentage of the IIOP dispatch time limit that should be used as the maximum amount of time that an IIOP request can spend on the workload management (WLM) queue. See the information about z/OS custom properties for the application server.
control_region_jvm_ localrefs		Should only be used under the direction of IBM support.
control_region_jvm_ logfile	User cannot change.	Specifies the file to which the conroller's JVM will write messages.
control_region_jvm_ properties_file	User cannot change.	Is dynamically created.
control_region_libpath	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, under Server Infrastructure, click **Java and process management > Process definition**, and specify the libpath.	
control_region_mdb_ queue_timeout_percent	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_mdb_queue_timeout_percent` property and specify a different value.	Indicates the percentage of the MDB dispatch time limit that should be used as the maximum amount of time that an MDB request can spend on the workload management (WLM) queue. See the information about z/OS custom properties for the application server.
control_region_mdb_ request_timeout (application server only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_mdb_request_timeout` property and specify a different value.	See the information about z/OS custom properties for the application server.
control_region_sip_ queue_timeout_percent	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_sip_queue_timeout_percent` property and specify a different value.	Indicates the percentage of the SIP dispatch time limit that should be used as the maximum amount of time that a SIP request can spend on the workload management (WLM) queue. See the information about z/OS custom properties for the application server.
control_region_sips_ queue_timeout_percent	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_sips_queue_timeout_percent` property and specify a different value.	Indicates the percentage of the SIPS dispatch time limit that should be used as the maximum amount of time that a SIPS request can spend on the workload management (WLM) queue. See the information about z/OS custom properties for the application server.
control_region_ssl_ thread_pool_size		Should only be changed under the direction of IBM Support personnel.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued). The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

control_region_thread_ pool_size		Should only be changed under the direction of IBM Support personnel.
control_region_thread_ stack_size		Should only be changed under the direction of IBM Support personnel.
control_region_timeout_ delay (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_timeout_delay` property and specify a different value.	See the information about z/OS custom properties for the application server.
control_region_timeout_ dump_action	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_timeout_dump_action` property and specify `SVCDUMP`, `JAVACORE`, `HEAPDUMP`, `TRACEBACK`, `JAVATDUMP`, or `NONE` for the value.	Indicates whether a Java core dump, an SVC dump, or a JVM-initiated TDUMP should be taken whenever a timeout occurs for work that has been dispatched to a servant. See the information about z/OS custom properties for the application server.
control_region_timeout_ dump_action_session	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_timeout_dump_action_session` property and specify `SVCDUMP`, `JAVACORE`, `HEAPDUMP`, `TRACEBACK`, `JAVATDUMP`, or `NONE` for the value.	Indicates whether a Java core dump, an SVC dump, or a JVM-initiated TDUMP should be taken whenever a timeout occurs for an HTTP, HTTPS, SIP, or SIPS request that has been dispatched to a servant. See the information about z/OS custom properties for the application server.
control_region_timeout_ save_last_servant	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `control_region_timeout_save_last_servant` property and specify a different value.	Indicates whether the last available servant should be abended if a timeout situation occurs on that servant, or the last available servant should remain active until a new servant is initialized. See the information about z/OS custom properties for the application server.
control_region_use _java_g	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, under Server Infrastructure, click **Java and process management > Process definition > Java Virtual Machine**, and select **Debug Mode**.	Indicates if conroller's JVM should use the debug JVM (java_g). Should only be changed under the direction of IBM Support personnel.
control_region_wlm_ dispatch_timeout (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, under Container Settings, click **Container Services > ORB Service** and specify a new value in the **WLM timeout** field.	
daemon_group_name	User cannot change.	

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued). The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

daemon_protocol_iiop _listenIPAddress	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `daemon_protocol_iiop_listenIPAddress` property, and specify either * to bind all, or the IP name to specify bind-specific support.	Allows you to restrict the IP addresses to which the location service daemon binds. You set this variable in your cell-level variables.xml file.
daemon_start_ command	User cannot change.	
daemon_start_ command_args	User cannot change.	
daemon_wlmable	User cannot change.	
daemonInstanceName	User cannot change.	
daemonName	User cannot change.	
default_internal_work_ transaction_class	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `default_internal_work_transaction_class` property, and specify a different value.	See the information about z/OS custom properties for the application server.
nls_language	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `nls_languag` property, and specify a different value.	See the information about z/OS custom properties for the application server.
node_name	User cannot change.	
node_short_name	User cannot change.	
nonauthenticated_ clients_allowed	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication Protocol** > **zSAS authentication**.	
private_bborlog_include _extra_server_info	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `private_bborlog_include_extra_server_info` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_accept_http_ work_ after_min_srs	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_accept_http_work_after_min_srs` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_accept_iiop_ work_after_min_srs	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_accept_iiop_work_after_min_srs` property, and specify a different value.	See the information about z/OS custom properties for the application server.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued). The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

protocol_bboc_log_ response_failure	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_bboc_log_response_failure` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_bboc_log_ return_exception	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_bboc_log_return_exception` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_giop_ level_highest	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_giop_level_highest` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_http_ backlog	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_http_backlog` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_http_large_ data_inbound_buffer	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_http_large_data_inbound_buffer` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_http_large_ data_response_buffer	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_http_large_data_response_buffer` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_http_max_ connect_backlog	User cannot update.	No longer used.
protocol_http_max_keep _alive_connections	User cannot update.	No longer used
protocol_http_timeout_ output (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name*, and then, under Web Container Settings, click **Custom properties**. Select the **ConnectionResponseTimeout** property and specify a new value.	See the topic on HTTP transport custom properties for additional information.
protocol_http_timeout_ output_recovery (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_http_timeout_output_recovery` property, and specify a different value.	See the information about z/OS custom properties for the application server.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued). The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

protocol_http_ transactionClass	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_http_transactionClass` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_http_transport_ class_mapping_file	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Web container settings > Web container > z/OS additional Settings**.	**depfeat:** This variable is deprecated. When possible, use a workload classification document file instead of this variable. See the information about classifying z/OS workload for more information.
protocol_https_backlog	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_http_backlog` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_https_max_ connect_backlog	User cannot change.	No longer used.
protocol_https_max_ keep_alive_connections	User cannot change.	No longer used.
protocol_https_ timeout_output (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Web container settings > Web container > HTTP transports >** `ssl_transport`	See the topic about HTTP transport custom properties for additional information.
protocol_https_timeout_ output_recovery (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_https_timeout_ output_recovery` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_https_ transactionClass	User cannot change. The value specified for the protocol_http_transport_class_mapping_file variable is also used for this variable.	
protocol_iiop_backlog _ssl	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_iiop_backlog_ssl` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_iiop_backlog	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_iiop_backlog` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_iiop_daemon_ listenIPAddress	In the administrative console, click **System Administration > Node groups > Sysplex node group > z/OS Location Service**. Specify the new IP address.	

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued). The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

protocol_iiop_daemon_ port	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service**.	
protocol_iiop_daemon_ port_ssl	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service**.	
protocol_iiop_no_ local_copies	In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, under Container Settings, click **Container Services > ORB Service**.	
protocol_iiop_propagate_ unknown_service_ctxs	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_iiop_propagate_unknown_service_ctxs` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_sip_timeout_ output_recovery (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_sip_timeout_output_recovery` property, and specify a different value.	See the information about z/OS custom properties for the application server.
protocol_sips_timeout_ output_recovery (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `protocol_sips_timeout_output_recovery` property, and specify a different value.	See the information about z/OS custom properties for the application server.
ras_debugEnabled		Should only be changed under the direction of IBM Support personnel.
ras_default_msg_dd	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `ras_default_msg_dd` property, and specify a different value.	
ras_dumpoptions_ dumptype		Should only be changed under the direction of IBM Support personnel.
ras_hardcopy_ msg_dd	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `ras_hardcopy_msg_dd` property, and specify a different value.	
ras_log_ logstreamName	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `ras_log_logstreamName` property, and specify a different value.	
ras_minorcode_ action		Should only be changed under the direction of IBM Support personnel.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued).   The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

ras_stderr_ff_interval	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `ras_stderr_ff_interval` property, and specify a different value.	
ras_stdout_ff_interval	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `ras_stdout_ff_interval` property, and specify a different value.	
ras_time_local	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `ras_time_local` property, and specify a different value.	See the information about z/OS custom properties for the application server.
ras_trace_basic		Should only be changed under the direction of IBM Support personnel.
ras_trace_ctraceParms		Should only be changed under the direction of IBM Support personnel.
ras_trace_ defaultTracingLevel		Should only be changed under the direction of IBM Support personnel.
ras_trace_detail		Should only be changed under the direction of IBM Support personnel.
ras_trace_exclude_ specific		Should only be changed under the direction of IBM Support personnel.
ras_trace_ minorCodeTraceBacks		Should only be changed under the direction of IBM Support personnel.
ras_trace_ outputLocation	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `ras_trace_outputLocation` property, and specify a different value.	
ras_trace_specific		Should only be changed under the direction of IBM Support personnel.
ras_trace_BufferCount	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `ras_trace_BufferCount` property, and specify a different value.	
ras_trace_BufferSize	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `ras_trace_BufferSize` property, and specify a different value.	

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued).   The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

read_license_ agreement	User cannot change.	Indicates that the server's initialization code will verify license agreement. The default is 1. If this variable is not set to 1, the server will not initialize.
security_SMF_record_ first_auth_user	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the pull-down list of available nodes and cells, and then click **New**. Add the **security_SMF_record_first_auth_user** property, and set it equal to 1 to fill the SM120CRE field with the ID of the first authenticated user. By default, the property is set to 0, and the SM120CRE field is filled with the ID under which the server activity began. This is often the guest ID.	If no authentication for a request is required, then the SM120CRE field will NOT contain an authenticated user ID for that request. Instead, it will contain an unauthenticated ID, typically the guest account ID or the WebSphere server ID.
security_sslKeyring	User cannot change.	No longer used.
security_zOS_ domainName	User cannot change.	Set during customization.
security_zOS_ domainType	User cannot change.	Set during customization.
security_zSAS_ssl_ repertoire	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Server security > zSAS Transport > SSL Setttings**. Select a different repertoire.	
security_ EnableRunAsIdentity	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `security_EnableRunAsIdentity` property, and specify a different value.	See the information about z/OS custom properties for the application server.
security_sslType1	User cannot change.	No longer used.
server_configured_ system_name	User cannot change.	Specifies the name of the system to which the server instance was originally configured.
server_generic_short_ name	If the server is clustered, this value is the cluster's short name. If the server is not clustered, this value is the value specified on the server custom property, **ClusterTransitionName**. Either way, this value can be changed using the administrative console.	
server_generic_uuid	User cannot change.	Specifies the unique identifier for this server
server_region_classpath (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Java Virtual Machine**, and, in the **Classpath** field, specify the classpath to be appended.	Specifies the classpath used by the JVM for the servant.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued).   The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

server_region_dpm_ dump_action	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_dpm_ dump_action` property, and specify a different value.	See the information about z/OS custom properties for the application server.
server_region_ dynapplenv_jclparms (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, under Server Infrastructure, click **Java and process management > Process definition > Servant > Start command arguments**, Specify the new parameters.	When dynamic applenv is being used (instead of the same content existing in static definition of WLM panels), this variable specifies the JCL parameters provided to the servant region when WLM starts this servant region.
server_region_ dynapplenv_jclproc (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, under Server Infrastructure, click **Java and process management > Process definition > servant > Start command**, and then specify the new JCL procedure.	When dynamic applenv is used, this variable specifies the name of the JCL procedure for a servant region when WLM starts this servant region.
server_region_jvm_ localrefs (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_jvm_localrefs` property, and specify a different value.	See the information about z/OS custom properties for the application server.
server_region_jvm_ logfile (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_jvm_logfile` property, and specify a different value.	Specifies the HFS file that JNI debug messages are written to.
server_region_jvm_ properties_file (application server and deployment manager only)	User cannot change	File is dynamically created.
server_region_libpath (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, under Server Infrastructure, click **Java and process management > Process definition > Servant > Environment entries**, and then specify a new value for the **LIBPATH** property.	Specifies the libpath for the servant region's JVM
server_region_recycle_ count (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_recycle_count` property, and specify a different value.	See the information about z/OS custom properties for the application server.
server_region_http_ stalled_thread_dump_ action (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_http_stalled_thread_dump_action` property, and specify a different value.	See the information about z/OS custom properties for the application server.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued).   The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

server_region_https_ stalled_thread_dump_ action (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_https_stalled_thread_dump_action` property, and specify a different value.	See the information about z/OS custom properties for the application server.
server_region_iiop_ stalled_thread_dump_ action (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_iiop_stalled_thread_dump_action` property, and specify a different value.	See the information about z/OS custom properties for the application server.
server_region_mdb_ stalled_thread_dump_ action (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_mdb_stalled_thread_dump_action` property, and specify a different value.	See the information about z/OS custom properties for the application server.
server_region_sip_ stalled_thread_dump_ action (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_sip_ stalled_thread_dump_ action` property, and specify a different value.	See the information about z/OS custom properties for the application server.
server_region_sips_ stalled_thread_dump_ action (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_sips_stalled_thread_dump_action` property, and specify a different value.	See the information about z/OS custom properties for the application server.
server_region_stalled_ thread_threshold_ percent (application server and deployment manager only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_region_stalled_thread_threshold_percent` property, and specify a different value.	See the information about z/OS custom properties for the application server.
server_region_thread_ stack_size (application server and deployment manager only)		Should only be changed under the direction of IBM Support personnel.
server_region_use_ java_g (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Java virtual machine**, and select **Debug Mode**.	Indicates if the servant region's JVM should use the debug JVM (java_g). Should only be used under the direction of IBM support.
server_region_workload _profile (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name*. Then, under Container Settings, click **Container Services > ORB Service**.	

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued).   The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

server_SMF_container_ activity_enabled (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Control > Environment entries**. Add the `server_SMF_container_activity_enabled` property and specify a different value.	See the topic about using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_container_ interval_enabled (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Control > Environment entries**. Add the `server_SMF_container_interval_enabled` property and specify a different value.	See the topic about using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_interval_ length (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Control > Environment entries**. Add the `server_SMF_interval_length` property and specify a different value.	See the topic about using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_request _activity_enabled	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Control > Environment entries**. Add the `server_SMF_request_activity_enabled` property and specify a different value.	See the topic about using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_request _activity_enabled _CPU_ detail	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Control > Environment entries**. Add the `server_SMF_request_activity_enabled_CPU_detail` property and specify a different value.	See the topic about using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_request _activity_enabled_ security	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Control > Environment entries**. Add the `server_SMF_request _activity_enabled_security` property and specify a different value.	See the topic about using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_request _activity_enabled _ timestamps	In the administrative console, click **Servers > Server Types > WebSphere application servers** > *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Control > Environment entries**. Add the `server_SMF_request_activity_enabled_timestamps` property and specify a different value.	See the topic about using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued). The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

server_SMF_server_ activity_enabled (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > server_name**. Then, under Server Infrastructure, click **Java and process management > Process definition > Control > Environment entries**. Add the `server_SMF_server_activity_enabled` property and specify a different value.	See the topic about using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information..
server_SMF_server_ interval_enabled (application server and deployment manager only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > server_name**. Then, under Server Infrastructure, click **Java and process management > Process definition > Control > Environment entries**. Add the `server_SMF_server_interval_enabled` property and specify a different value.	See the topic about using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_web_ container_activity_ enabled	User cannot change.	No longer used.
server_SMF_web_ container_interval_ enabled	User cannot change.	No longer used.
serverRegionid	User cannot change.	No longer used.
server_specific_name	User cannot change.	
server_specific_ short_name	In the administrative console, click **Servers > Server Types > WebSphere application servers > server_name**, and specify a new value in the **Short name** field.	
server_specific_uuid	User cannot change.	Specifies the unique identifier for this server
server_start_wait_ for_initialization_Timeout	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_start_wait_for_initialization_Timeout` property, and specify a different value.	See the information about z/OS custom properties for the application server.
server_wto_on_write_error	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `server_wto_on_write_error` property, and specify a different value.	Indicates whether the error message BBOO0384I ERROR OCCURRED WRITING TO {0} is written to the SYSLOG when an error occurs while writing to SYSPRINT or SYSOUT. See the information about z/OS custom properties for the application server.
suppress_hung_thread_abend (application server only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `suppress_hung_thread_abend` property, and specify a different value.	See the information about z/OS custom properties for the application server.

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued).   The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

suppress_hung_thread_dump (application server only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `suppress_hung_thread_dump` property, and specify a different value.	See the information about z/OS custom properties for the application server.
transaction_ defaultTimeout (application server only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name*. Then, under Container Settings, click **Container Services > Transaction Service** and specify a new value in the **Transaction lifetime timeout** field.	The maximum duration, in seconds, for transactions on this application server. Any transaction that is not requested to complete before this timeout will be rolled back. Default is 120.
transaction_ maximumTimeout (application server only)	In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name*. Then, under Container Settings, click **Container Services > Transaction Service** and specify a new value in the **Maximum Transaction Timeout** field.	The maximum duration, in seconds, that transactions propagated into the server or transactions started by BMT components from within the server will be allowed to execute. Any transaction that is not requested to complete before this timeout will be rolled back. Default is 300.
transaction_ recoveryTimeout (application server only)	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `transaction_recoveryTimeout` property and specify a different value.	See the information about z/OS custom properties for the application server.
was_env_file	User cannot change.	File is dynamically created.
wlm_dynapplenv_ single_server (application server and deployment manager only)	For an application server, In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name*. Then, under Server Infrastructure, click **Java and process management > Server instance**. For a deployment manager, in the administrative console, click **System administration > Deployment manager**. Then, under Server Infrastructure, click **Java and process management > Server instance**.	
wlm_ maximumSRCount (application server and deployment manager only)	For an application server, In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name*. Then, under Server Infrastructure, click **Java and process management > Server instance**. For a deployment manager, in the administrative console, click **System administration > Deployment manager**. Then, under Server Infrastructure, click **Java and process management > Server instance**	

Table 43. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued). The following table maps internal variable references in BBOM0001I messages to external WebSphere variables, custom properties, or administrative console fields.

wlm_ minimumSRCount (application server and deployment manager only)	For an application server, In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, under Server Infrastructure, click **Java and process management > Server instance**. For a deployment manager, in the administrative console, click **System administration > Deployment manager**. Then, under Server Infrastructure, click **Java and process management > Server instance**	
wsadmin_dumpthreads_ enable_heapdump	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `wsadmin_dumpthreads_enable_heapdump` property, and specify either 0 or 1.	Indicates whether a Java heap dump is generated, in addition to a Java core dump when you issue the wsadmin dumpThreads command. See the information about z/OS custom properties for the application server.
wsadmin_dumpthreads_ enable_javatdump	In the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available nodes and cells, and then click **New**. Add the `wsadmin_dumpthreads_enable_javatdump` property, and specify either 0 or 1.	Indicates whether a system TDUMP is generated, in addition to a Java core dump when you issue the wsadmin dumpThreads command. See the information about z/OS custom properties for the application server.

About this task

After you determine which WebSphere variable, custom property or administrative console field needs to be updated, complete the following procedure to change the value of that setting.

Procedure

1. Navigate to the appropriate administrative console panel. The "How to change indicated value" column describes how to navigate within the administrative console to the appropriate panel for changing a specific internal variable setting. For example, to change the setting of the wlm_maximumSRCount variable for an application server, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, under Server Infrastructure, click **Java and process management > Server instance**.

2. Update the variable, custom property, or administrative console field with the new value.

3. Click **Apply**, and then click **SAVE** to save your changes.

Environment entries collection

Use this page to view and manage arbitrary name-value pairs of data, where the name is a environment entry key and the value is a string value that can be used to set internal system configuration environment entries.

To view this page, in the administrative console click **Servers** > **Server Types** > **WebSphere application servers** > ***server_name***, and then under Server Infrastructure, click **Java and process management** > **Environment entries**.

Name
Specifies the name (or key) for the environment entry. The name is a string that is used to set an internal system configuration environment entry.

Each environment entry name must be unique. If the same name is used for multiple environment entries, the value specified for the first environment entry that has that name is used.

Do not start your environment entry names with `was.` because this prefix is reserved for environment entries that are predefined for WebSphere Application Server.

Value
Specifies the value paired with the specified name.

Description
Provides information about the name-value pair.

Environment entries settings
Use this page to configure arbitrary name-value pairs of data, where the name is an environment entry key and the value is a string value that can be used to set internal system configuration environment entries. Defining a new environment entry enables you to configure a setting beyond that which is available in the administrative console.

To view this page, in the administrative console click **Servers > Server Types > WebSphere application servers >** *server_name*. Under Server Infrastructure, click **Java and process management > Environment entries**. Then do one of the following:
* Click **New** to create a new environment entry.
* Click the name of an existing environment entry to change its settings,
* Select an existing environment entry and click **Delete** to delete that entry.

Name:

Specifies the name (or key) for the environment entry.

Each environment entry name must be unique. If the same name is used for multiple environment entries, the value specified for the first environment entry that has that name is used.

Do not start an environment entry name with `was.` because this prefix is reserved for environment entries that are predefined in WebSphere Application Server.

Data type	String

Value:

Specifies the value paired with the specified name.

Data type	String

Description:

Provides information about the name and value pair.

Data type	String

Updating resources for an application server

Properly updating resources ensures that all transactional work completes while the original versions of the resources are still available. If resources are not properly updated, data builds up in the transaction partner log. Eventually high CPU usage is observed in the controller.

Before you begin

Before you start to update a resource, verify that all of the transactions being handled by that resource have completed..

About this task

In severe cases, when high CPU usage occurs in the controller because resources are improperly updated, the partner logs becomes full, and the application server becomes unusable. When the partner logs become full, the following error message appears in the servant log:

```
BBOO0220E: WTRN0000E: An internal error occurred in method logData in class
com.ibm.ws.Transaction.JTA.PartnerLog
```

If before updating a resource, you ensure that there is no work pending that involves this resource, data will not build up in the transaction partner log. However, data accumulates in the transaction partner logs if there is a change in resource, there is a change in the configuration for a resource, or a resource is deleted, before all of the transactional work that the resource is handling completes. This situation occurs because after the resource update occurs, the old version of the resource is no longer available for recovery when the server is restarted.

If the recovery process does not completes when you restart the server, periodically, the product attempts to recover those transactions. To determine if there are transactions that have pending resolutions, look for the following message in the controller log:

```
BBOT0009I: TRANSACTION SERVICE RESTART UR STATUS COUNTS FOR SERVER {0}: IN-BACKOUT={1},
IN-DOUBT={2}, IN-COMMIT={3}
```

Procedure

1. Make sure the resources you want to change are available.
2. Stop the application server.
3. Restart the server in recovery mode

   Restarting the server in recovery mode ensures all transactions are resolved, and the transaction partner logs are clean for next server restart. The server automatically shuts down after the recovery process completes.

   See the topic *Restarting an application server in recovery mode* for a description of how to perform this step.

   If the server you started in recovery mode does not shutdown after a reasonable amount of time, you might already have old resource entries in the partner logs that cannot be recovered. You should contact IBM Support for assistance in determining the issue, and the proper course of action.
4. Start the administrative console to make your resource configuration changes.

   You need to use the administrative console to make the resources changes.

   In a stand-alone environment, you must start the single server that is defined for this environment before you can access the administrative console and make the changes to the resources. However after you start this server, you must ensure that no one attempts to use any of the resources you are changing. If that resource is accessed even once, the resource is be put into the recovery log, and any recovery attempt the next time the server starts will fail if the resource cannot be reached.

   In a Network Deployment environment, you can start any of the application servers except the application server that uses the resources you are changing.
5. Make your resources changes to the server configuration
6. Save and synchronize your changes.
7. If you are running in a Network Deployment environment, start the application server that uses the resources you changed.

Results

The application server is now using the changed resources with no impact to the transaction partner logs.

Starting an application server

When you start an application server, a new server process starts. This new server process is based on the process definition settings of the current server configuration.

Before you begin

Before you start an application server, verify that all of the application required resources are available. You must also start all prerequisite subsystems.

If you want server components to dynamically start as they are needed by the installed applications, verify that the **Start components as needed** option is selected in the configuration settings for the application server before you start the application server. Selecting this option can improve startup time, and reduce the memory footprint of the application server. Starting components as they are needed is most effective if all of the applications that are deployed on the server are of the same type. For example, using this option works better if all of your applications are web applications that use servlets, and JavaServer Pages (JSP). This option works less effectively if your applications use servlets, JSPs, and Enterprise JavaBeans (EJB).

gotcha: To ensure compatibility with other WebSphere products, the default setting for this option is cleared. Before selecting this option, verify that any other WebSphere products, that you are running with this product, support this function.

About this task

The node agent for the node on which an application server resides must be running before you can start the application server.

This procedure for starting a server also typically applies to restarting a server. The one exception might be if a server fails and you want the recovery functions to complete their processing before new work being started on that server. In this situation, you must restart the server in recovery mode.

After you create an application server definition, you can start, stop, or manage the new server using the administrative console, or you can use commands to complete these tasks for the new server.

After you start an application server, other processes might not immediately discover the running application server. Application servers are discovered by the node agent. However, node agents are discovered by the deployment manager. Even though node agents typically discover local application servers quickly, it might take a deployment manager up to 60 seconds to discover a node agent.

If you are using clusters, the **Initial State** property of the application server subcomponent is not intended to be used to control the state of individual servers in the cluster at the time the cluster is started. This property is intended only as a way to control the state of the subcomponent of a server. You should use the Server options on the administrative console, or the startServer and stopServer command-line commands to start and stop the individual servers of a cluster

There are several options available for starting an application server.

Procedure

- You can use the administrative console:
    1. Click **Servers** > **Server Types** > **WebSphere application servers** to determine the node agent on which the application server that you are starting resides.
    2. Click **System administration** > **Node agents**, and verify that the node agent is running.

If the node agent is not running, issue the `startNode` command. After a node agent completely stops running and remains stopped, you cannot remotely start the node agent from the Node Agents page. You must issue the `startNode` command to start the node agent on the node where it runs.

3. Click **Servers** > **Server Types** > **WebSphere application servers** again and select the application server that you want to start.

4. Click **Start**. You can view the status and any messages or logs to make sure the application server starts.

- You can issue a start command from the MVS console. Issue the following MVS console command, all in uppercase and on a single line:

1. To start a server, issue the following command all in uppercase and on a single line:

   ```
   S procname,JOBNAME=server_shortname,ENV=cell_shortname.node_shortname.server_shortname
   ```

   where:

   procname
   > Is the JCL procedure name in the proclib that is used to start the server.

   servername
   > Is the short name of the server which is also used as the process job name.

   cellname.nodename.servername
   > This element of the ENV parameter is a concatenation of the cell short name, the node short name, and the server short name.

   For example:

   ```
   S BBO7ACR,JOBNAME=BBOS001,ENV=BBOBASE.BBONODE.BBOS001
   ```

   transition: If you are migrating from a previous version of the product and want to be able to restart the server in recovery mode, make sure that the ENV parameter for this procedure includes either the REC=N or the REC=Y element. If the ENV parameter includes the REC=N element, the setting is automatically changed to REC=Y if you specify `-recovery` when you restart the server. The REC=N element is automatically included on the ENV parameter if you did not migrate from a previous version of the product. The following example illustrates what your updated PROC statement might look like:

   ```
   //BBO7ACR  PROC ENV=,PARMS=' ',REC=N,Z=BBO7ACRZ
   ```

   The following messages indicate that the controller is running:

   ```
   $HASP100 BBO7ACR  ON STCINRDR
   $HASP373 BBO7ACR  STARTED
   BBOO0001I WEBSPHERE FOR Z/OS CONTROL PROCESS BBODMNB/SY1/BBOC001/BBOS001
             IS STARTING.
   IRR812I PROFILE BBO*.* (G) IN THE STARTED CLASS WAS USED TO START BBOS001S
           WITH JOBNAME BBOS001S.
   $HASP100 BBOS001S ON STCINRDR
   $HASP373 BBOS001S STARTED
   +BBOO0004I WEBSPHERE FOR Z/OS SERVANT PROCESS BBODMNB/SY1/BBOC001/BBOS001
             IS STARTING.
   +BBOO0020I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS SERVANT PROCESS
             BBOS001.
   BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL PROCESS
             BBOS001.
   ```

- You can issue a startServer command.

   Read the topic on the startServer command for information about the command, including such information as running the command and defining the file name for the start server log. Read the topic on using command-line tools for information such as determining from what directory to run the startServer command.

   You can check that the server has successfully started by checking the start server log. If the server has started successfully, the last two lines of the start server log look like the following example:

```
Server launched.  Waiting for initialization status.
Server server1 open for e-business; process id is 1932.
```

Results

The specified server starts. To verify that the server is in start state, in the administrative console, click **Servers** > **Server Types** > **WebSphere application servers**.

What to do next

After the server starts, deploy the applications that you want to run on this server.

If you must start an application server with standard Java debugging enabled:
1. In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers**.
2. Click the name of the application server with the processes that you want to trace and debug.
3. Under Server Infrastructure, click **Java and process management** > **Process definition**.
4. Select **Control**.
5. Select **Java virtual machine**.
6. On the Java virtual machine page, select the **Debug mode** option to start the standard Java debugger. Set **Debug mode** arguments, if they are needed.
7. Click **OK**.
8. Save the changes to a configuration file
9. Stop the application server.
10. Start the application server again as previously described.

Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root
> Refers to the top directory for a WebSphere Application Server node.
>
> The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.
>
> The default varies based on node type. Common defaults are `configuration_root`/AppServer and `configuration_root`/DeploymentManager.

configuration_root
> Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.
>
> The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.
>
> The default is `/wasv8config/`*cell_name*`/`*node_name*.

plug-ins_root
> Refers to the installation root directory for Web Server Plug-ins.

profile_root
> Refers to the home directory for a particular instantiated WebSphere Application Server profile.
>
> Corresponding product variables are *server.root* and *user.install.root*.

In general, this is the same as *app_server_root*/profiles/*profile_name*. On z/OS, this will always be *app_server_root*/profiles/default because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root
Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.

The corresponding product variable is *smpe.install.root*.

The default is /usr/lpp/zWebSphere/V8R0.

Restarting an application server in recovery mode

When an application server instance with active transactions in progress restarts after a failure, the transaction service uses recovery logs to complete the recovery process. These logs, which each transactional resource maintains, are used to rerun any InDoubt transactions and return the overall system to a self-consistent state.

Before you begin

If you are migrating from a previous version of the product, make sure that the REC parameter is included on the JCL procedure statement for the controller as either REC=N or REC=Y. If the JCL procedure does not specify either REC=N or REC=Y, the server does not restart in recovery mode even if you specify the -recovery option.

If the JCL procedures includes REC=N, the setting automatically changes to REC=Y if you specify -recovery when you restart the server. REC=N is automatically included on the JCL procedure if you did not migrate from a previous version of the product. Following is an example of what your updated PROC statement might look like:

```
//BBO6ACR  PROC PARMS=' ',REC=N,Z=BBO6ACRZ
```

About this task

When you restart an application server in recovery mode:
* Transactional resources complete the actions in their recovery logs and then shut down. This action frees up any resource locks that the application server held prior to the failure.
* During the recovery period, only the subset of application server functions that are necessary for transactional recovery to proceed are available.
* The application server does not accept new work during the recovery process.
* The application server shuts down when the recovery is complete.

This recovery process begins as soon as all of the necessary subsystems within the application server are available. If the application server is not restarted in recovery mode, the application server can start accepting new work as soon as the server is ready, which might occur before the recovery work has completed.

Normally, this process is not a problem. However, situations exist when your operating procedures might not be compatible with supporting recovery work and new work simultaneously. For example, you might have a high availability environment where the work handled by the application server that failed is immediately moved to another application server. This backup application server then exclusively processes the work from the application server that failed until recovery has completed on the failed application server and the two application servers can be re-synchronized. In this situation, you might want the failing application server to only perform its transactional recovery process and then shut down. You might not want this application server to start accepting new work while the recovery process is taking place.

To prevent the assignment of new work to an application server that is going through its transaction recovery process, restart the application server in recovery mode.

When you restart a failed application server, the node agent for the node on which the failed application server resides must be running before you can restart that application server.

gotcha: When an application server stops as part of normal shutdown processing, message `WSVR0024I: Server xxxxxxxx PROCESS xxxxxxxx stopped` is sent to the system log file. If the server user Ids have ALTER access to the appropriate MVSADMIN.* profiles in the facility class, the resource manager registration entry that is associated with the application server for this instance of the application server is removed from the RRS logs. However, if the server user Ids have do not have ALTER access to the appropriate MVSADMIN.* profiles in the facility class, the resource manager registration entry that is associated with the application server for this instance of the application server is not removed from the RRS logs.

If the resource manager registration entry was deleted from the RRS logs, on a subsequent application server start, a cold start is performed. However, you cannot perform a cold start with RRS if you are starting the application server in recovery mode.

With this service release, you can cold start the server in a recovery mode only on the system where the server was configured.

If you want to be able restart an application server in recovery mode, you must perform the following steps before a failure occurs, and then restart the application server to enable your configuration changes:

Procedure

- If the server is monitored by a node agent, you must clear the Automatic restart option for that server. Clearing this option prevents the node agent from automatically restarting the server in normal mode, before you have a chance to start it in recovery mode.
  1. In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name*.
  2. In the Server Infrastructure section, click **Java and process management** > **Monitoring Policy**.
  3. Clear the Automatic restart option.
- If a catastrophic failure occurs that leaves InDoubt transactions, issue the **startServer** *server_name* **-recovery** command from the command line. This command restarts the server in recovery mode. You must issue the command from the *profile_root*/`bin` directory for the profile with which the server is associated.

Results

The application server restarts in recovery mode, performs transactional recovery, and shuts down. Any resource locks that the application server held prior to the failure are released.

What to do next

Configure the integrated high availability support for the transaction service subcomponent for peer recovery of transactions.

InFlight work and presumed abort mode

Presumed abort mode is activated when a failure occurs before a distributed transaction starts to commit.

If you have a distributed transaction that spans several servers, transactional locks may be held by resource managers involved in that work. When a failure occurs before that distributed transaction has started to commit, the product and the resource managers go into presumed abort mode. In this mode, the resource managers rolls back the transaction.

- The effect of a server failure or communications failure will vary depending on which server is running the work at the time of failure.
- An OTS timeout may be required to rollback the subordinate branches of the distributed transaction tree.

Example: A common case of this is when you have a server B web client that is driving a session bean in the same server. That session bean has executed work against entity beans in servers C and D. All of the servers are involved in the same distributed, global transaction. Suddenly, server B fails while the session bean is InFlight (meaning it hadn't started to commit yet). Servers C and D are waiting for more work or the start of the two-phase commit protocol, but, while in this state, the transactional locks may still be held by the resource managers. So, the server roles are as follows:
- Server A: Servlet/JavaServer Page executed
- Server B: Session bean accessed
- Server C: Entity bean accessed
- Server D: Entity bean accessed

After the timeout occurs, because the session bean is InFlight at the time of the failure, the product rolls back the transaction branch.

When local resource managers are involved, RRS ensures that they are called to perform presumed abort processing. When doing recovery, RRS works with the resource managers to ensure that the recovery is done properly. When a failure occurs while work is InFlight, RRS directs the resource managers involved in the local UR to rollback.

The product always assumes that there is recovery to do. Every time a server comes up, it does something different depending in which mode it is running:
- If the server is running in restart/recovery mode, the product checks to see whether there is any recovery required. If recovery is required, the product attempts to complete the recovery and either succeeds or terminates.
- If the server is running normally, the restart/recovery transaction does not have to complete before the server takes on new work. After the server determines what the restart work is, it begins to take in new work items. Processing of the restart/recovery transaction continues along with the processing of new work items.

IMS Connect considerations following server recovery

After InDoubt and InFlight work completes, the product server shuts down. A new application server configured for that system is then started up to accept new work. Special considerations must be taken if you are using IMS Connect after recovering to an alternate system.

After server recovery is completed, IMS Connect starts, but is not usable without some manual intervention. On the current IMS Connect WTOR perform the following commands `nn,viewhws` followed by `nn,viewhwsnn,opends` *XXX* where *XXX* is the IMS subsystem name displayed in the result of the `nn,viewhws` query. The IMS datastore needs to reflect 'active' status, as is shown in the following example:

```
*17 HWSC0000I *IMS CONNECT READY*  IMSCONN
 R 17,VIEWHWS
 IEE600I REPLY TO 17 IS;VIEWHWS
 HWSC0001I   HWS ID=IMSCONN     Racf=N
 HWSC0001I     Maxsoc=100  Timeout=12000
 HWSC0001I    Datastore=IMS      Status=ACTIVE
 HWSC0001I      Group=IMSGROUP Member=IMSCONN
 HWSC0001I      Target Member=IMSA
 HWSC0001I    Port=9999     Status=ACTIVE
 HWSC0001I      No active Clients
 HWSC0001I    Port=LOCAL    Status=ACTIVE
 HWSC0001I      No active Clients
```

After you complete the required manual intervention, IMS Connect is ready to handle new work on this server.

Detecting and handling problems with runtime components

You must monitor the status of runtime components to ensure that, once started, they remain operational as needed.

Procedure

1. Regularly examine the status of runtime components.

   Browse messages displayed under WebSphere Runtime Messages in the status area at the bottom of the console. The runtime event messages, marked with a red X, provide detailed information on event processing.

2. If an application stops running, examine the status of the application If an application stops running when it should be operational, examine the status of the application on an Applications page and try restarting the application. If messages indicate that a server has stopped running, use the Application servers page to try restarting the server. If a cluster of servers stops running, use the Server Cluster page to try to restart the cluster. If the status of an application server is Unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

3. If the runtime components do not restart, reexamine the messages and read information on problem determination to help you to restart the components.

Stopping an application server

Stopping an application server ends a server process based on the process definition settings in the current application server configuration.

Before you begin

Make sure you understand the impact of stopping a particular server has on your ability to handle work requests, especially if you need to maintain a highly available environment.

About this task

There are times you need to stop an application server. For example, you might have to apply service to an application running on that server, or you might want to change one of the application server's configuration setting. Use one of the following options when you need to stop an application server.

Procedure

- You can issue a stopServer command to stop a single server or the stopManager command to stop the deployment manager.

  Read the stopServer and stopManager topics for information about the commands, including such information as running the commands and defining the log file names.

  Read the topic on using command-line tools for information such as determining from what directory to run the stopServer and stopManager commands.

  You can check that the server or deployment manager has successfully stopped by checking the appropriate log file.

  You should not use the `CANCEL appserver_proc_name` command to stop a server. Every time a server is started, a new `temp` directory is created off of the servant process token, such as *profile_root*/`default/`temp/*node_name*/*server_name*. When the server is cleanly stopped, these `temp` directories are normally removed. However, if the server is frequently not stopped cleanly, which happens if you cancel rather than stop the server, these `temp` directories are not removed and the HFS used for these `temp` directories eventually becomes full. You can also prevent this storage problem from occurring if you precompile your JavaServer pages when you install an application or if you use the JspBatchCompiler function to precompile them before they are invoked.

- You can use the administrative console to stop an application server:

  1. In the administrative console, click **Servers > Server Types > WebSphere application servers**.

2. Select the application server that you want stopped and click **Stop**.
3. Confirm that you want to stop the application server.
4. View the **Status** value and any messages or logs to see whether the application server stops.

Results

The specified server stops as soon as requests assigned to that server finish processing. To verify that the server is in stop state, in the administrative console, click **Servers > Server Types > WebSphere application servers**.

What to do next

If you experience any problems shutting down a server, see the *Troubleshooting and support* PDF.

Automatically rejecting work requests when no servant is available to process these requests

When a controller determines that a servant has terminated, the controller normally cleans up any other work requests that were dispatched in that servant. If that servant is the last servant, new work requests are placed in the request queue until a servant is available. Depending on how long it takes for a servant to become available, these requests might terminate because the time allowed to process a request has expired. To prevent this from happening, you can change the configuration settings for an application server to prevent the controller from accepting new requests.

About this task

Controllers receive application requests on a continual basis and dispatch them to a servant for processing. When a system level problem, such as a database error, occurs, request processing stops and requests pile up in the queues between the controller and the servants. During the time it takes for a servant to become available, requests continue to pile up in the queues until they start to time out. When a timeout occurs, the request that timed out is removed from the queue.

When a new servant is ready to start accepting requests, the next request in the queue might be so close to timing out that the dispatch process for the request cannot complete and the servant again is terminated by timeout processing. Again requests accumulate in the queue until another new servant is ready and potentially the same timeout problem occurs. When this problem keeps reoccurring, it is sometimes referred to as a *bouncing servant* problem. You can handle this problem in one of the following ways:

- You can configure the server to automatically detect a no-server situation and stop accepting requests until the minimum configured number of servants are ready to accept work. This is the simplest approach.
- You can create an automation routine to handle the problem if you are able to detect that you are having a system problem before servants are terminated because of timeouts. This automation routine can issue the f *server*, `pauselisteners` command to prevent requests from being accepted by this server. The routine must then detect when circumstances have changed and issue the f *server*, `resumelisteners` command when the detected system problem is resolved.
- You can configure the server to detect a no-server condition and stop accepting requests, and create the previously discussed automation routine. The automation routine must recognize the different processing that can take place because the server is configured to detect a no-server condition:
  - If the last servant terminates even though the f *server*, `pauselisteners` command is issued, the server starts to reject all requests and issues message BBOO0299I. The server automatically starts to accept requests when the minimum number of servants, for which the server is configured, are ready to accept work. It also issues message BBOO0300I to indicate that requests are again being processed. Therefore, the automation routine must be sensitive to the fact that the server might have resumed accepting requests upon detecting that the minimal number of servants are available.

- If the control_region_confirm_recovery_on_no_srs custom property is specified for the server, the server issues WTOR message BBOO0297A after it detects that the minimal number of servants, for which the server is configured, are ready to process new requests. You must enter a response to this message before the server actually starts to accept work.
- If the automation routine prevents the server from terminating the servants because of timeout processing, it must also recognize when it is safe for the server to resume taking requests and issue the f *server*, resumelisteners command at that point in time. The automation routine can be set up to determine whether or not it needs to issue the f *server*, resumelisteners command based on whether or not the message BBOO0299I is issued. This message indicates that the server ran out of servants and is rejecting requests. This approach is the most complex, but provides the most flexibility.

Complete the following steps if you want to configure the server to handle no-server conditions.

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere application servers**, and select the application server for which you want to automatically detect no-servant conditions.
2. Under Additional Properties, click **Custom properties > New**.
3. Specify control_region_dreg_on_no_srs in the Name field and 1 in the Value field. When this custom property is set to 1, the server rejects all requests targeted for dispatch when it detects that there are no servants ready to process the requests. Setting this property to 0 (zero) turns off this function.
4. Specify control_region_confirm_recovery_on_no_srs in the Name field and either 0 (zero) or 1 in the Value field. If you enter 0 in the Value field, the controller resumes taking requests as soon as the minimum number of servants are ready to receive requests. If you enter 1 in the Value field, the controller issues WTOR message BBOO0297A as soon as it detects that the minimum number of servants for which the server is configured are ready to accept work. The server waits until it receives a response to this message before it actually resumes taking requests.
5. Click **Review**, select **Synchronize changes with Nodes**, and then click **Save** to update the master repository with your changes.

Results

When a controller determines that a servant has terminated, and that servant is the last servant, the controller will not accept new work requests until the minimum number of servants are available to receive requests.

Converting a 7-character server short name to 8 characters

By default, the product assumes you are using a 7-character short name (JOBNAME) for your application servers and deployment managers. If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.

Before you begin

You should consider the following before performing this task:

- The Resource Recovery Services (RRS) log names are based on the server short name. When you change the server short name, you are changing the server's identity to the RRS. This means that the previously existing transaction and partner logs will be abandoned, or will not match the new name, and either of these situations will result in restart problems. To prevent this from happening, ensure that there are no outstanding RRS units of recovery (URs) for your server **before** changing its name. See *z/OS MVS Programming: Resource Recovery* for instructions on using the RRS panels to view information about URs.

  Note that the only safe way to provide an 8-character short name for a server is to do so before it is initially started.

- Converting your 7-character server short name to 8 characters requires you to change the JOBNAME used by the servant's start command. This means that the System Authorization Facility (SAF) started class that previously matched this job may no longer match. Review your SAF STARTED class profile and, if necessary, define a new class.
- Because the JOBNAME appears as part of a start command's arguments, you need to review your COMMNDxx PARMLIB member, as well as any other form of automation you use that issues a start command to start a server.
- Review the start parameters of your Workload Management (WLM) static APPLENV definitions. These are the parameters that are used to start the servant process (server region). If you are using static APPLENVs, the start parm string used by the WLM for your server looks similar to JOBNAME=BBOS001S,ENV=... You will need to decide if you want to keep this JOBNAME or change to the new JOBNAME that you specify in the steps below. The original JOBNAME should be sufficient.

  This consideration does not apply if you are using WLM dynamic APPLENVs.
- Review and update the necessary Resource Access Control Facility (RACF) profiles to support these server short names. See the *Securing applications and their environment* PDF for more information.

About this task

To lengthen a 7-character server short name to 8 characters:

Procedure

1. Change the 7-character server short name to the 8-character name you wish to use:
   a. Navigate to the appropriate application server or deployment manager.

      For an application server, in the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*.

      For a deployment manager, in the administrative console, click **System Administration > Deployment manager**.
   b. In the **Short name** field, replace the 7-character name with the 8-character short name you wish to use.
   c. Click **OK**.
2. Update the start command arguments for the servant to use the new 8-character name. If you are reconfiguring a node agent, you can skip this step because it does not have an associated servant process.
   a. Navigate to the servant process.

      For an application server, in the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Java and process management > Process definition > Servant**.

      For a deployment manager, in the administrative console, click **System Administration > Deployment manager**, and then under Server Infrastructure, click **Java and process management > Process definition**.
   b. In the startCommandArgs field, replace the 7-character name, designated by the JOBNAME argument, with the 8-character name you wish to use. Do not include the S character at the end of the JOBNAME. For example, JOBNAME=P5SVR1D,ENV=P5CELL.P5NODED.P5SVR1D
   c. Click **OK**.
3. Click **Save** to save your configuration changes.

Changing time zone settings

In some application environments, it is important that application server components use the same time zone. You can use the administrative console or system environment variables to ensure that your application components use the correct time zone.

Before you begin

Determine the scope at which you want to set the time zone value. You can set the time zone value such that is applies for an entire cell, for an entire node, or only for a specific server.

Remember that time zone IDs should include an offset and, in almost all cases, a daylight saving time zone name for consistent results. For example, specify EST5EDT for Eastern Standard Time, Daylight Savings Time.

About this task

In general cases, the time zone for application server is inherited from the time zone that is set for the operating system; Java should be inherit the time zone from the operating system, and the application server will use the time zone that is set for each Java Virtual Machine (JVM). If you need to configure a different time zone for a single JVM, you can set the TZ environment variable in the application server, modify the properties file, or specify a command-line parameter when the JVM starts.

You can specify the Unix System Services (USS) TZ variable as an environment variable to set the time stamps for your application logs.

Procedure

- Set the time zone for each of your server processes.

  1. In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Java process management** > **Process definition** > **Environment entries**.

  2. Set a value for the TZ variable.

     – If the TZ variable is included in the list of defined variables, click **TZ**, and then specify a new time zone value in the **Value** field.

     – If the TZ variable is not included in the list of defined variables, click **New**, and then specify TZ in the **Name** field, and the appropriate time zone value in the **Value** field.

     For example, if you specify TZ in the **Name** field, and EST5EDT in the **Value** field, Eastern United States is used as the time zone setting for all of your server processes.

     The following table lists the Unix System Services (USS) TZ variable values for the various time zones in the United States. The same values can be specified for the WebSphere TZ variable.

Table 44. United states time zones. This table lists the United States time zone values that can be specified for the USS TZ variable.

Time zone code	Time zone	CUT offset
CUT0GDT	Coordinated Universal Time	CUT
EST5EDT	Eastern United States	CUT-5
CST6CDT	Central United States	CUT-6
MST7MDT	Mountain United States	CUT-7
PST8PDT	Pacific United States	CUT-8
AST9ADT	Alaska	CUT-9
HST10HDT	Hawaii	CUT-10

     See the topic *Time zone codes for the TZ environment variable* in your z/OS Information Center for a complete list of the Unix System Services (USS) TZ variable values.

  3. Click **Apply**, and then click **Save** to save your changes.

  4. Stop and restart all of the affected application server that were running when you made the time zone changes.

- Set the time zone with a command-line property for each JVM. For example, use the following parameter to set the time zone on the Java call:

  `-Duser.timezone=time_zone_code`

Results

Your new time zone setting are in affect for the designated servers.

Time zone IDs that can be specified for the user.timezone property

The following table lists the time zone IDs that you can specify for the user.timezone property.

- The **Time zone ID** column lists time zones, in boldface, and the locations within each time zone.
- The **Raw offset** column lists the difference, in hours and minutes, between Greenwich Mean Time (GMT) and the specified time zone.
- The **DST offset** column lists the offset, in minutes, for Daylight Savings Time (DST). If the field is blank, the time zone does not use DST.
- The **Display name** column lists the names of the time zones.
- The **QTIMZON variable** column only applies to the IBM i operating system. The **QTIMZON variable** column lists the corresponding value for the QTIMZON system variable. If multiple values are specified in this column, either value is acceptable.

Important: The United States and Canada are making changes to the Daylight Saving Time start and end dates. The Technote Changes to Daylight Saving Time will affect IBM WebSphere Application Server and its associated Operating Systems, that is available on the Support website, provides the latest information on service updates that are being made to support these changes.

Table 45. Time zone IDs for the user.timezone property. The following table lists the time zone IDs that you can specify for the user.timezone property.

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Etc/GMT+12	-12 : 00		GMT-12:00	
Etc/GMT+11	-11 : 00		GMT-11:00	
MIT	-11 : 00		West Samoa Time	
Pacific/Apia	-11 : 00		West Samoa Time	QN1100UTCS
Pacific/Midway	-11 : 00		Samoa Standard Time	
Pacific/Niue	-11 : 00		Niue Time	
Pacific/Pago_Pago	-11 : 00		Samoa Standard Time	
Pacific/Samoa	-11 : 00		Samoa Standard Time	
US/Samoa	-11 : 00		Samoa Standard Time	
America/Adak	-10 : 00	60	Hawaii-Aleutian Standard Time	QN1000HAST
America/Atka	-10 : 00	60	Hawaii-Aleutian Standard Time	
Etc/GMT+10	-10 : 00		GMT-10:00	
HST	-10 : 00		Hawaii Standard Time	
Pacific/Fakaofo	-10 : 00		Tokelau Time	
Pacific/Honolulu	-10 : 00		Hawaii Standard Time	QN1000UTCS
Pacific/Johnston	-10 : 00		Hawaii Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Pacific/Rarotonga	-10 : 00		Cook Is. Time	
Pacific/Tahiti	-10 : 00		Tahiti Time	
SystemV/HST10	-10 : 00		Hawaii Standard Time	
US/Aleutian	-10 : 00	60	Hawaii-Aleutian Standard Time	
US/Hawaii	-10 : 00		Hawaii Standard Time	
Pacific/Marquesas	-9 : 30		Marquesas Time	
AST	-9 : 00	60	Alaska Standard Time	QN0900AST
America/Anchorage	-9 : 00	60	Alaska Standard Time	
America/Juneau	-9 : 00	60	Alaska Standard Time	
America/Nome	-9 : 00	60	Alaska Standard Time	
America/Yakutat	-9 : 00	60	Alaska Standard Time	
Etc/GMT+9	-9 : 00		GMT-09:00	
Pacific/Gambier	-9 : 00		Gambier Time	QN0900UTCS
SystemV/YST9	-9 : 00	60	Alaska Standard Time	
US/Alaska	-9 : 00	60	Alaska Standard Time	
America/Dawson	-8 : 00	60	Pacific Standard Time	
America/Ensenada	-8 : 00	60	Pacific Standard Time	
America/Los_Angeles	-8 : 00	60	Pacific Standard Time	
America/Tiajuana	-8 : 00	60	Pacific Standard Time	
America/Vancouver	-8 : 00	60	Pacific Standard Time	
America/Whitehorse	-8 : 00	60	Pacific Standard Time	
Canada/Pacific	-8 : 00	60	Pacific Standard Time	
Canada/Yukon	-8 : 00	60	Pacific Standard Time	
Etc/GMT+8	-8 : 00		GMT-08:00	
Mexico/BajaNorte	-8 : 00	60	Pacific Standard Time	
PST	-8 : 00	60	Pacific Standard Time	QN0800PST, QN0800U
PST8PDT	-8 : 00	60	Pacific Standard Time	
Pacific/Pitcairn	-8 : 00		Pitcairn Standard Time	QN0800UTCS
SystemV/PST8	-8 : 00		Pitcairn Standard Time	
SystemV/PST8PDT	-8 : 00	60	Pacific Standard Time	
US/Pacific	-8 : 00	60	Pacific Standard Time	
US/Pacific-New	-8 : 00	60	Pacific Standard Time	
America/Boise	-7 : 00	60	Mountain Standard Time	
America/Cambridge_Bay	-7 : 00	60	Mountain Standard Time	
America/Chihuahua	-7 : 00	60	Mountain Standard Time	
America/Dawson_Creek	-7 : 00		Mountain Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
America/Denver	-7 : 00	60	Mountain Standard Time	
America/Edmonton	-7 : 00	60	Mountain Standard Time	
America/Hermosillo	-7 : 00		Mountain Standard Time	
America/Inuvik	-7 : 00	60	Mountain Standard Time	
America/Mazatlan	-7 : 00	60	Mountain Standard Time	
America/Phoenix	-7 : 00		Mountain Standard Time	QN0700MST2, QN0700UTCS
America/Shiprock	-7 : 00	60	Mountain Standard Time	
America/Yellowknife	-7 : 00	60	Mountain Standard Time	
Canada/Mountain	-7 : 00	60	Mountain Standard Time	
Etc/GMT+7	-7 : 00		GMT-07:00	
MST	-7 : 00	60	Mountain Standard Time	QN0700MST, QN0700T
MST7MDT	-7 : 00	60	Mountain Standard Time	
Mexico/BajaSur	-7 : 00	60	Mountain Standard Time	
Navajo	-7 : 00	60	Mountain Standard Time	
PNT	-7 : 00	60	Mountain Standard Time	
SystemV/MST7	-7 : 00		Mountain Standard Time	
SystemV/MST7MDT	-7 : 00	60	Mountain Standard Time	
UA/Arizona	-7 : 00		Mountain Standard Time	
US/Mountain	-7 : 00	60	Mountain Standard Time	
America/Belize	-6 : 00		Central Standard Time	
America/Cancun	-6 : 00	60	Central Standard Time	
America/Chicago	-6 : 00	60	Central Standard Time	
America/Costa_Rica	-6 : 00		Central Standard Time	QN0600UTCS
America/El_Salvador	-6 : 00		Central Standard Time	
America/Guatemala	-6 : 00		Central Standard Time	
America/Managua	-6 : 00		Central Standard Time	
America/Menominee	-6 : 00	60	Central Standard Time	
America/Merida	-6 : 00	60	Central Standard Time	
America/Mexico_City	-6 : 00	60	Central Standard Time	
America/Monterrey	-6 : 00	60	Central Standard Time	
America/North_Dakota/Center	-6 : 00	60	Central Standard Time	
America/Rainy_River	-6 : 00	60	Central Standard Time	
America/Rankin_Inlet	-6 : 00	60	Central Standard Time	
America/Regina	-6 : 00		Central Standard Time	
America/Swift_Current	-6 : 00		Central Standard Time	
America/Tegucigalpa	-6 : 00		Central Standard Time	

Table 45. Time zone IDs for the user.timezone property (continued). The following table lists the time zone IDs that you can specify for the user.timezone property.

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
America/Winnipeg	-6 : 00	60	Central Standard Time	
CST	-6 : 00	60	Central Standard Time	QN0600CST, QN600S
CST6CDT	-6 : 00	60	Central Standard Time	
Canada/Central	-6 : 00	60	Central Standard Time	
Canada/East-Saskatchewan	-6 : 00		Central Standard Time	
Canada/Saskatchewan	-6 : 00		Central Standard Time	
Chile/EasterIsland	-6 : 00	60	Easter Is.Time	
Etc/GMT+6	-6 : 00		GMT-06:00	
Mexico/General	-6 : 00	60	Central Standard Time	
Pacific/Easter	-6 : 00	60	Easter Is. Time	
Pacific/Galapagos	-6 : 00		Galapagos Time	
Pacific/Easter	-6 : 00	60	Easter Is. Time	
Pacific/Galapagos	-6 : 00		Galapagos Time	
SystemV/CST6	-6 : 00		Central Standard Time	
SystemV/CST6CDT	-6 : 00	60	Central Standard Time	
US/Central	-6 : 00	60	Central Standard Time	
America/Bogota	-5 : 00		Colombia Time	
America/Cayman	-5 : 00		Eastern Standard Time	
America/Detroit	-5 : 00	60	Eastern Standard Time	
America/Eirunepe	-5 : 00		Acre Time	
America/Fort_Wayne	-5 : 00		Eastern Standard Time	
America/Grand_Turk	-5 : 00	60	Eastern Standard Time	
America/Guayaquil	-5 : 00		Ecuador Time	
America/Havana	-5 : 00	60	Central Standard Time	
America/Indiana/Indianapolis	-5 : 00		Eastern Standard Time	
America/Indiana/Knox	-5 : 00		Eastern Standard Time	
America/Indiana/Marengo	-5 : 00		Eastern Standard Time	
America/Indiana/Vevay	-5 : 00		Eastern Standard Time	
America/Indianapolis	-5 : 00		Eastern Standard Time	QN0500UTCS
America/Iqaluit	-5 : 00	60	Eastern Standard Time	
America/Jamaica	-5 : 00		Eastern Standard Time	
America/Kentucky/Louisville	-5 : 00	60	Eastern Standard Time	
America/Kentucky/Monticello	-5 : 00	60	Eastern Standard Time	
America/Knox_IN	-5 : 00		Eastern Standard Time	
America/Lima	-5 : 00		Peru Time	
America/Louisville	-5 : 00	60	Eastern Standard Time	
America/Montreal	-5 : 00	60	Eastern Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
America/Nassau	-5 : 00	60	Eastern Standard Time	
America/New_York	-5 : 00	60	Eastern Standard Time	
America/Nipigon	-5 : 00	60	Eastern Standard Time	
America/Panama	-5 : 00		Eastern Standard Time	
America/Pangnirtung	-5 : 00	60	Eastern Standard Time	
America/Port-au-Prince	-5 : 00		Eastern Standard Time	
America/Porto_Acre	-5 : 00		Acre Time	
America/Rio_Branco	-5 : 00		Acre Time	
America/Thunder_Bay	-5 : 00	60	Eastern Standard Time	
Brazil/Acre	-5 : 00		Acre Time	
Canada/Eastern	-5 : 00	60	Eastern Standard Time	
Cuba	-5 : 00	60	Central Standard Time	
EST	-5 : 00	60	Eastern Standard Time	QN0500EST
EST5EDT	-5 : 00	60	Eastern Standard Time	
Etc/GMT+5	-5 : 00		GMT-05:00	
IET	-5 : 00		Eastern Standard Time	QN0500EST2
Jamaica	-5 : 00		Eastern Standard Time	
SystemV/EST5	-5 : 00		Eastern Standard Time	
SystemV/EST5EDT	-5 : 00	60	Eastern Standard Time	
US/East-Indiana	-5 : 00		Eastern Standard Time	
US/Eastern	-5 : 00	60	Eastern Standard Time	
US/Indiana-Starke	-5 : 00		Eastern Standard Time	
US/Michigan	-5 : 00	60	Eastern Standard Time	
America/Anguilla	-4 : 00		Atlantic Standard Time	
America/Antigua	-4 : 00		Atlantic Standard Time	
America/Aruba	-4 : 00		Atlantic Standard Time	
America/Asuncion	-4 : 00	60	Paraguay Time	
America/Barbados	-4 : 00		Atlantic Standard Time	
America/Boa_Vista	-4 : 00		Amazon Standard Time	
America/Caracas	-4 : 00		Venezuela Time	QN0400UTC2
America/Cuiaba	-4 : 00	60	Amazon Standard Time	
America/Curacao	-4 : 00		Atlantic Standard Time	
America/Dominica	-4 : 00		Atlantic Standard Time	
America/Glace_Bay	-4 : 00	60	Atlantic Standard Time	
America/Goose_Bay	-4 : 00	60	Atlantic Standard Time	
America/Grenada	-4 : 00		Atlantic Standard Time	
America/Guadeloupe	-4 : 00		Atlantic Standard Time	
America/Guyana	-4 : 00		Guyana Time	

Table 45. Time zone IDs for the user.timezone property (continued). The following table lists the time zone IDs that you can specify for the user.timezone property.

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
America/Halifax	-4 : 00	60	Atlantic Standard Time	
America/La_Paz	-4 : 00		Bolivia Time	
America/Manaus	-4 : 00		Amazon Standard Time	
America/Martinique	-4 : 00		Atlantic Standard Time	
America/Montserrat	-4 : 00		Atlantic Standard Time	
America/Port_of_Spain	-4 : 00		Atlantic Standard Time	
America/Porto_Velho	-4 : 00		Amazon Standard Time	
America/Puerto_Rico	-4 : 00		Atlantic Standard Time	QN0400UTCS
America/Santiago	-4 : 00	60	Chile Time	
America/Santo_Domingo	-4 : 00		Atlantic Standard Time	
America/St_Kitts	-4 : 00		Atlantic Standard Time	
America/St_Lucia	-4 : 00		Atlantic Standard Time	
America/St_Thomas	-4 : 00		Atlantic Standard Time	
America/St_Vincent	-4 : 00		Atlantic Standard Time	
America/Thule	-4 : 00	60	Atlantic Standard Time	
America/Tortola	-4 : 00		Atlantic Standard Time	
America/Virgin	-4 : 00		Atlantic Standard Time	
Antarctica/Palmer	-4 : 00	60	Chile Time	
Atlantic/Bermuda	-4 : 00	60	Atlantic Standard Time	QN0400AST
Atlantic/Stanley	-4 : 00	60	Falkland Is. Time	
Brazil/West	-4 : 00		Amazon Standard Time	
Canada/Atlantic	-4 : 00	60	Atlantic Standard Time	
Chile/Continental	-4 : 00	60	Chile Time	
Etc/GMT+4	-4 : 00		GMT-04:00	
PRT	-4 : 00		Atlantic Standard Time	
SystemV/AST4	-4 : 00		Atlantic Standard Time	
SystemV/AST4ADT	-4 : 00	60	Atlantic Standard Time	
America/St_Johns	-3 : 30	60	Newfoundland Standard Time	
CNT	-3 : 30	60	Newfoundland Standard Time	QN0330NST
Canada/Newfoundland	-3 : 30	60	Newfoundland Standard Time	
AGT	-3 : 00		Argentine Time	
America/Araguaina	-3 : 00	60	Brazil Time	
America/Belem	-3 : 00		Brazil Time	
America/Buenos_Aires	-3 : 00		Argentine Time	QN0300UTCS
America/Catamarca	-3 : 00		Argentine Time	
America/Cayenne	-3 : 00		French Guiana Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
America/Cordoba	-3 : 00		Argentine Time	
America/Fortaleza	-3 : 00		Brazil Time	
America/Godthab	-3 : 00	60	Western Greenland Time	
America/Jujuy	-3 : 00		Argentine Time	
America/Maceio	-3 : 00		Brazil Time	
America/Mendoza	-3 : 00		Argentine Time	
America/Miquelon	-3 : 00	60	Pierre & Miquelon Standard Time	
America/Montevideo	-3 : 00		Uruguay Time	
America/Paramaribo	-3 : 00		Suriname Time	
America/Recife	-3 : 00		Brazil Time	
America/Rosario	-3 : 00		Argentine Time	
America/Sao_Paulo	-3 : 00	60	Brazil Time	
Antarctica/Rothera	-3 : 00		Rothera Time	
BET	-3 : 00	60	Brazil Time	QN0300UTC2
Brazil/East	-3 : 00	60	Brazil Time	
Etc/GMT+3	-3 : 00		GMT-03:00	
America/Noronha	-2 : 00		Fernando de Noronha Time	QN0200UTCS
Atlantic/South_Georgia	-2 : 00		South Georgia Standard Time	
Brazil/DeNoronha	-2 : 00		Fernando de Noronha Time	
Etc/GMT+2	-2 : 00		GMT-02:00	
America/Scoresbysund	-1 : 00	60	Eastern Greenland Time	
Atlantic/Azores	-1 : 00	60	Azores Time	
Atlantic/Cape_Verde	-1 : 00		Cape Verde Time	QN0100UTCS
Etc/GMT+1	-1 : 00		GMT-01:00	
Africa/Abidjan	0 : 00		Greenwich Mean Time	
Africa/Accra	0 : 00		Greenwich Mean Time	
Africa/Bamako	0 : 00		Greenwich Mean Time	
Africa/Banjul	0 : 00		Greenwich Mean Time	
Africa/Bissau	0 : 00		Greenwich Mean Time	
Africa/Casablanca	0 : 00		Western European Time	
Africa/Conakry	0 : 00		Greenwich Mean Time	
Africa/Dakar	0 : 00		Greenwich Mean Time	
Africa/El_Aaiun	0 : 00		Western European Time	
Africa/Freetown	0 : 00		Greenwich Mean Time	
Africa/Lome	0 : 00		Greenwich Mean Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Africa/Monrovia	0 : 00		Greenwich Mean Time	
Africa/Nouakchott	0 : 00		Greenwich Mean Time	
Africa/Ouagadougou	0 : 00		Greenwich Mean Time	
Africa/Sao_Tome	0 : 00		Greenwich Mean Time	
Africa/Timbuktu	0 : 00		Greenwich Mean Time	
America/Danmarkshavn	0 : 00		Greenwich Mean Time	
Atlantic/Canary	0 : 00	60	Western European Time	
Atlantic/Faeroe	0 : 00	60	Western European Time	
Atlantic/Madeira	0 : 00	60	Western European Time	
Atlantic/Reykjavik	0 : 00		Greenwich Mean Time	
Atlantic/St_Helena	0 : 00		Greenwich Mean Time	
Eire	0 : 00	60	Greenwich Mean Time	
Etc/GMT	0 : 00		GMT+00:00	
Etc/GMT+0	0 : 00		GMT+00:00	
Etc/GMT-0	0 : 00		GMT+00:00	
Etc/GMT0	0 : 00		GMT+00:00	
Etc/Greenwich	0 : 00		Greenwich Mean Time	
Etc/UCT	0 : 00		Coordinated Universal Time	
Etc/UTC	0 : 00		Coordinated Universal Time	
Etc/Universal	0 : 00		Coordinated Universal Time	
Etc/Zulu	0 : 00		Coordinated Universal Time	
Europe/Belfast	0 : 00	60	Greenwich Mean Time	
Europe/Dublin	0 : 00	60	Greenwich Mean Time	
Europe/Lisbon	0 : 00	60	Western European Time	
Europe/London	0 : 00	60	Greenwich Mean Time	Q0000GMT2
GB	0 : 00	60	Greenwich Mean Time	
GB-Eire	0 : 00	60	Greenwich Mean Time	
GMT	0 : 00		Greenwich Mean Time	Q0000GMT
GMT0	0 : 00		GMT+00:00	
Greenwich	0 : 00		Greenwich Mean Time	
Iceland	0 : 00		Greenwich Mean Time	
Portugal	0 : 00	60	Western European Time	
UCT	0 : 00		Coordinated Universal Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
UTC	0 : 00		Coordinated Universal Time	Q0000UTC
Universal	0 : 00		Coordinated Universal Time	
WET	0 : 00	60	Western European Time	
Zulu	0 : 00		Coordinated Universal Time	
Africa/Algiers	1 : 00		Central European Time	QP0100CET, QP0100UTCS
Africa/Bangui	1 : 00		Western African Time	
Africa/Brazzaville	1 : 00		Western African Time	
Africa/Ceuta	1 : 00	60	Central European Time	
Africa/Douala	1 : 00		Western African Time	
Africa/Kinshasa	1 : 00		Western African Time	
Africa/Lagos	1 : 00		Western African Time	
Africa/Libreville	1 : 00		Western African Time	
Africa/Luanda	1 : 00		Western African Time	
Africa/Malabo	1 : 00		Western African Time	
Africa/Ndjamena	1 : 00		Western African Time	
Africa/Niamey	1 : 00		Western African Time	
Africa/Porto-Novo	1 : 00		Western African Time	
Africa/Tunis	1 : 00		Central European Time	
Africa/Windhoek	1 : 00	60	Western African Time	
Arctic/Longyearbyen	1 : 00	60	Central European Time	
Atlantic/Jan_Mayen	1 : 00	60	Eastern Greenland Time	
CET	1 : 00	60	Central European Time	
ECT	1 : 00	60	Central European Time	QP0100CET3
Etc/GMT-1	1 : 00		GMT+01:00	
Europe/Amsterdam	1 : 00	60	Central European Time	
Europe/Andorra	1 : 00	60	Central European Time	
Europe/Belgrade	1 : 00	60	Central European Time	
Europe/Berlin	1 : 00	60	Central European Time	
Europe/Bratislava	1 : 00	60	Central European Time	
Europe/Brussels	1 : 00	60	Central European Time	
Europe/Budapest	1 : 00	60	Central European Time	
Europe/Copenhagen	1 : 00	60	Central European Time	
Europe/Gibraltar	1 : 00	60	Central European Time	
Europe/Ljubljana	1 : 00	60	Central European Time	
Europe/Luxembourg	1 : 00	60	Central European Time	

Table 45. Time zone IDs for the user.timezone property (continued). The following table lists the time zone IDs that you can specify for the user.timezone property.

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Europe/Madrid	1 : 00	60	Central European Time	
Europe/Malta	1 : 00	60	Central European Time	
Europe/Monaco	1 : 00	60	Central European Time	
Europe/Oslo	1 : 00	60	Central European Time	
Europe/Paris	1 : 00	60	Central European Time	
Europe/Prague	1 : 00	60	Central European Time	
Europe/Rome	1 : 00	60	Central European Time	
Europe/San_Marino	1 : 00	60	Central European Time	
Europe/Sarajevo	1 : 00	60	Central European Time	
Europe/Skopje	1 : 00	60	Central European Time	
Europe/Stockholm	1 : 00	60	Central European Time	
Europe/Tirane	1 : 00	60	Central European Time	
Europe/Vaduz	1 : 00	60	Central European Time	
Europe/Vatican	1 : 00	60	Central European Time	
Europe/Vienna	1 : 00	60	Central European Time	
Europe/Warsaw	1 : 00	60	Central European Time	
Europe/Zagreb	1 : 00	60	Central European Time	
Europe/Zurich	1 : 00	60	Central European Time	QP0100CET2
MET	1 : 00	60	Middle Europe Time	
Poland	1 : 00	60	Central European Time	
ART	2 : 00	60	Eastern European Time	
Africa/Blantyre	2 : 00		Central African Time	
Africa/Bujumbura	2 : 00		Central African Time	
Africa/Cairo	2 : 00	60	Eastern European Time	
Africa/Gaborone	2 : 00		Central African Time	
Africa/Harare	2 : 00		Central African Time	
Africa/Johannesburg	2 : 00		South Africa Standard Time	QP0200SAST
Africa/Kigali	2 : 00		Central African Time	
Africa/Lubumbashi	2 : 00		Central African Time	
Africa/Lusaka	2 : 00		Central African Time	
Africa/Maputo	2 : 00		Central African Time	
Africa/Maseru	2 : 00		South Africa Standard Time	
Africa/Mbabane	2 : 00		South Africa Standard Time	
Africa/Tripoli	2 : 00		Eastern European Time	
Asia/Amman	2 : 00	60	Eastern European Time	
Asia/Beirut	2 : 00	60	Eastern European Time	

The following table lists the time zone IDs that you can specify for the user.timezone property.

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Asia/Damascus	2 : 00	60	Eastern European Time	
Asia/Gaza	2 : 00	60	Eastern European Time	
Asia/Istanbul	2 : 00	60	Eastern European Time	
Asia/Jerusalem	2 : 00	60	Israel Standard Time	
Asia/Nicosia	2 : 00	60	Eastern European Time	
Asia/Tel_Aviv	2 : 00	60	Israel Standard Time	
CAT	2 : 00		Central African Time	
EET	2 : 00	60	Eastern European Time	QP0200EET
Egypt	2 : 00	60	Eastern European Time	
Etc/GMT-2	2 : 00		GMT+02:00	
Europe/Athens	2 : 00	60	Eastern European Time	
Europe/Bucharest	2 : 00	60	Eastern European Time	
Europe/Chisinau	2 : 00	60	Eastern European Time	
Europe/Helsinki	2 : 00	60	Eastern European Time	
Europe/Istanbul	2 : 00	60	Eastern European Time	
Europe/Kaliningrad	2 : 00	60	Eastern European Time	
Europe/Kiev	2 : 00	60	Eastern European Time	
Europe/Minsk	2 : 00	60	Eastern European Time	
Europe/Nicosia	2 : 00	60	Eastern European Time	
Europe/Riga	2 : 00	60	Eastern European Time	
Europe/Simferopol	2 : 00	60	Eastern European Time	
Europe/Sofia	2 : 00	60	Eastern European Time	
Europe/Tallinn	2 : 00	60	Eastern European Time	QP0200EET2, QP0200UTCS
Europe/Tiraspol	2 : 00	60	Eastern European Time	
Europe/Uzhgorod	2 : 00	60	Eastern European Time	
Europe/Vilnius	2 : 00	60	Eastern European Time	
Europe/Zaporozhye	2 : 00	60	Eastern European Time	
Israel	2 : 00	60	Israel Standard Time	
Libya	2 : 00		Eastern European Time	
Turkey	2 : 00	60	Eastern European Time	
Africa/Addis_Ababa	3 : 00		Eastern African Time	QP0300UTCS
Africa/Asmera	3 : 00		Eastern African Time	
Africa/Dar_es_Salaam	3 : 00		Eastern African Time	
Africa/Djibouti	3 : 00		Eastern African Time	
Africa/Kampala	3 : 00		Eastern African Time	
Africa/Khartoum	3 : 00		Eastern African Time	
Africa/Mogadishu	3 : 00		Eastern African Time	

Table 45. Time zone IDs for the user.timezone property (continued). The following table lists the time zone IDs that you can specify for the user.timezone property.

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Africa/Nairobi	3 : 00		Eastern African Time	
Antarctica/Syowa	3 : 00		Syowa Time	
Asia/Aden	3 : 00		Arabia Standard Time	
Asia/Baghdad	3 : 00	60	Arabia Standard Time	
Asia/Bahrain	3 : 00		Arabia Standard Time	
Asia/Kuwait	3 : 00		Arabia Standard Time	
Asia/Qatar	3 : 00		Arabia Standard Time	
Asia/Riyadh	3 : 00		Arabia Standard Time	
EAT	3 : 00		Eastern African Time	
Etc/GMT-3	3 : 00		GMT+03:00	
Europe/Moscow	3 : 00	60	Moscow Standard Time	
Indian/Antananarivo	3 : 00		Eastern African Time	
Indian/Comoro	3 : 00		Eastern African Time	
Indian/Mayotte	3 : 00		Eastern African Time	
W-SU	3 : 00	60	Moscow Standard Time	
Asia/Riyadh87	3 : 07		GMT+03:07	
Asia/Riyadh88	3 : 07		GMT+03:07	
Asia/Riyadh89	3 : 07		GMT+03:07	
Mideast/Riyadh87	3 : 07		GMT+03:07	
Mideast/Riyadh88	3 : 07		GMT+03:07	
Mideast/Riyadh89	3 : 07		GMT+03:07	
Asia/Tehran	3 : 30	60	Iran Standard Time	
Iran	3 : 30	60	Iran Standard Time	
Asia/Aqtau	4 : 00	60	Aqtau Time	QP0400UTC2
Asia/Baku	4 : 00	60	Azerbaijan Time	
Asia/Dubai	4 : 00		Gulf Standard Time	QP0400UTCS
Asia/Muscat	4 : 00		Gulf Standard Time	
Asia/Oral	4 : 00	60	Oral Time	
Asia/Tbilisi	4 : 00	60	Georgia Time	
Asia/Yerevan	4 : 00	60	Armenia Time	
Etc/GMT-4	4 : 00		GMT+04:00	
Europe/Samara	4 : 00	60	Samara Time	
Indian/Mahe	4 : 00		Seychelles Time	
Indian/Mauritius	4 : 00		Mauritius Time	
Indian/Reunion	4 : 00		Reunion Time	
NET	4 : 00	60	Armenia Time	
Asia/Kabul	4 : 30		Afghanistan Time	
Asia/Aqtobe	5 : 00	60	Aqtobe Time	QP0500UTC2

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Asia/Ashgabat	5 : 00		Turkmenistan Time	
Asia/Ashkhabad	5 : 00		Turkmenistan Time	
Asia/Bishkek	5 : 00	60	Kirgizstan Time	
Asia/Dushanbe	5 : 00		Tajikistan Time	
Asia/Karachi	5 : 00		Pakistan Time	QP0500UTCS
Asia/Samarkand	5 : 00		Turkmenistan Time	
Asia/Tashkent	5 : 00		Uzbekistan Time	
Asia/Yekaterinburg	5 : 00	60	Yekaterinburg Time	
Etc/GMT-5	5 : 00		GMT+05:00	
Indian/Kerguelen	5 : 00		French Southern & Antarctic Lands Time	
Indian/Maldives	5 : 00		Maldives Time	
PLT	5 : 00		Pakistan Time	
Asia/Calcutta	5 : 30		India Standard Time	
IST	5 : 30		India Standard Time	QP0530IST
Asia/Katmandu	5 : 45		Nepal Time	
Antarctica/Mawson	6 : 00		Mawson Time	
Antarctica/Vostok	6 : 00		Vostok Time	
Asia/Almaty	6 : 00	60	Alma-Ata Time	QP0600UTC2
Asia/Colombo	6 : 00		Sri Lanka Time	
Asia/Dacca	6 : 00		Bangladesh Time	
Asia/Dhaka	6 : 00		Bangladesh Time	QP0600UTCS
Asia/Novosibirsk	6 : 00	60	Novosibirsk Time	
Asia/Omsk	6 : 00	60	Omsk Time	
Asia/Qyzylorda	6 : 00	60	Qyzylorda Time	
Asia/Thimbu	6 : 00		Bhutan Time	
Asia/Thimphu	6 : 00		Bhutan Time	
BST	6 : 00		Bangladesh Time	
Etc/GMT-6	6 : 00		GMT+06:00	
Indian/Chagos	6 : 00		Indian Ocean Territory Time	
Asia/Rangoon	6 : 30		Myanmar Time	
Indian/Cocos	6 : 30		Cocos Islands Time	
Antarctica/Davis	7 : 00		Davis Time	
Asia/Bangkok	7 : 00		Indochina Time	
Asia/Hovd	7 : 00		Hovd Time	
Asia/Jakarta	7 : 00		West Indonesia Time	QP0700WIB
Asia/Krasnoyarsk	7 : 00	60	Krasnoyarsk Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Asia/Phnom_Penh	7 : 00		Indochina Time	
Asia/Pontianak	7 : 00		West Indonesia Time	
Asia/Saigon	7 : 00		Indochina Time	QP0700UTCS
Asia/Vientiane	7 : 00		Indochina Time	
Etc/GMT-7	7 : 00		GMT+07:00	
Indian/Christmas	7 : 00		Christmas Island Time	
VST	7 : 00		Indochina Time	
Antarctica/Casey	8 : 00		Western Standard Time (Australia)	
Asia/Brunei	8 : 00		Brunei Time	
Asia/Chongqing	8 : 00		China Standard Time	
Asia/Chungking	8 : 00		China Standard Time	
Asia/Harbin	8 : 00		China Standard Time	
Asia/Hong_Kong	8 : 00		Hong Kong Time	QP0800JIST, QP0800UTCS
Asia/Irkutsk	8 : 00	60	Irkutsk Time	
Asia/Kashgar	8 : 00		China Standard Time	
Asia/Kuala_Lumpur	8 : 00		Malaysia Time	
Asia/Kuching	8 : 00		Malaysia Time	
Asia/Macao	8 : 00		China Standard Time	
Asia/Macau	8 : 00		China Standard Time	
Asia/Makassar	8 : 00		Central Indonesia Time	
Asia/Manila	8 : 00		Philippines Time	
Asia/Shanghai	8 : 00		China Standard Time	
Asia/Singapore	8 : 00		Singapore Time	
Asia/Taipei	8 : 00		China Standard Time	
Asia/Ujung_Pandang	8 : 00		Central Indonesia Time	QP0800WITA
Asia/Ulaanbaatar	8 : 00		Ulaanbaatar Time	
Asia/Ulan_Bator	8 : 00		Ulaanbaatar Time	
Asia/Urumqi	8 : 00		China Standard Time	
Australia/Perth	8 : 00		Western Standard Time (Australia)	QP0800AWST
Australia/West	8 : 00		Western Standard Time (Australia)	
CTT	8 : 00		China Standard Time	QP0800BST
Etc/GMT-8	8 : 00		GMT+08:00	
Hongkong	8 : 00		Hong Kong Time	
PRC	8 : 00		China Standard Time	
Singapore	8 : 00		Singapore Time	

Table 45. Time zone IDs for the user.timezone property (continued). The following table lists the time zone IDs that you can specify for the user.timezone property.

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Asia/Choibalsan	9 : 00		Choibalsan Time	
Asia/Dili	9 : 00		East Timor Time	
Asia/Jayapura	9 : 00		East Indonesia Time	QP0900WIT
Asia/Pyongyang	9 : 00		Korea Standard Time	
Asia/Seoul	9 : 00		Korea Standard Time	QP0900KST
Asia/Tokyo	9 : 00		Japan Standard Time	QP0900UTCS
Asia/Yakutsk	9 : 00	60	Yakutsk Time	
Etc/GMT-9	9 : 00		GMT+09:00	
JST	9 : 00		Japan Standard Time	QP0900JST
Japan	9 : 00		Japan Standard Time	
Pacific/Palau	9 : 00		Palau Time	
ROK	9 : 00		Korea Standard Time	
ACT	9 : 30		Central Standard Time (Northern Territory)	
Australia/Adelaide	9 : 30	60	Central Standard Time (South Australia)	QP0930ACST
Australia/Broken_Hill	9 : 30	60	Central Standard Time (South Australia/New South Wales)	
Australia/Darwin	9 : 30		Central Standard Time (Northern Territory)	
Australia/North	9 : 30		Central Standard Time (Northern Territory)	
Australia/South	9 : 30	60	Central Standard Time (South Australia)	
Australia/Yancowinna	9 : 30	60	Central Standard Time (South Australia/New South Wales)	
AET	10 : 00	60	Eastern Standard Time (New South Wales)	QP1000AEST
Antarctica/DumontDUrville	10 : 00		Dumont-d'Urville Time	
Asia/Sakhalin	10 : 00	60	Sakhalin Time	
Asia/Vladivostok	10 : 00	60	Vladivostok Time	
Australia/ACT	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Brisbane	10 : 00		Eastern Standard Time (Queensland)	
Australia/Canberra	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Hobart	10 : 00	60	Eastern Standard Time (Tasmania)	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Australia/Lindeman	10 : 00		Eastern Standard Time (Queensland)	
Australia/Melbourne	10 : 00	60	Eastern Standard Time (Victoria)	
Australia/NSW	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Queensland	10 : 00		Eastern Standard Time (Queensland)	
Australia/Sydney	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Tasmania	10 : 00	60	Eastern Standard Time (Tasmania)	
Australia/Victoria	10 : 00	60	Eastern Standard Time (Victoria)	
Etc/GMT-10	10 : 00		GMT+10:00	
Pacific/Guam	10 : 00		Chamorro Standard Time	QP1000UTCS
Pacific/Port_Moresby	10 : 00		Papua New Guinea Time	
Pacific/Saipan	10 : 00		Chamorro Standard Time	
Pacific/Truk	10 : 00		Truk Time	
Pacific/Yap	10 : 00		Yap Time	
Australia/LHI	10 : 30	30	Load Howe Standard Time	
Australia/Lord_Howe	10 : 30	30	Load Howe Standard Time	
Asia/Magadan	11 : 00	60	Magadan Time	
Etc/GMT-11	11 : 00		GMT+11:00	
Pacific/Efate	11 : 00		Vanuatu Time	
Pacific/Guadalcanal	11 : 00		Solomon Is. Time	QP1100UTCS
Pacific/Kosrae	11 : 00		Kosrae Time	
Pacific/Noumea	11 : 00		New Caledonia Time	
Pacific/Ponape	11 : 00		Ponape Time	
SST	11 : 00		Solomon Is. Time	
Pacific/Norfolk	11 : 30		Norfolk Time	
Antarctica/McMurdo	12 : 00	60	New Zealand Standard Time	
Antarctica/South_Pole	12 : 00	60	New Zealand Standard Time	
Asia/Anadyr	12 : 00	60	Anadyr Time	
Asia/Kamchatka	12 : 00	60	Petropavlovsk-Kamchatski Time	
Etc/GMT-12	12 : 00		GMT+12:00	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (IBM i only)
Kwajalein	12 : 00		Marshall Islands Time	
NST	12 : 00	60	New Zealand Standard Time	QP1200NZST
NZ	12 : 00	60	New Zealand Standard Time	
Pacific/Auckland	12 : 00	60	New Zealand Standard Time	
Pacific/Fiji	12 : 00		Fiji Time	QN1200UTCS, QP1200UTCS
Pacific/Funafuti	12 : 00		Tuvalu Time	
Pacific/Kwajalein	12 : 00		Marshall Islands Time	
Pacific/Majuro	12 : 00		Marshall Islands Time	
Pacific/Nauru	12 : 00		Nauru Time	
Pacific/Tarawa	12 : 00		Gilbert Is. Time	
Pacific/Wake	12 : 00		Wake Time	
Pacific/Wallis	12 : 00		Wallis & Futuna Time	
NZ-CHAT	12 : 45	60	Chatham Standard Time	
Pacific/Chatham	12 : 45	60	Chatham Standard Time	QP1245UTCS
Etc/GMT-13	13 : 00		GMT+13:00	
Pacific/Enderbury	13 : 00		Phoenix Is. Time	
Pacific/Tongatapu	13 : 00		Tonga Time	
Etc/GMT-14	14 : 00		GMT+14:00	
Pacific/Kiritimati	14 : 00		Line Is. Time	

Web module or application server stops processing requests

If an application server process spontaneously closes, or web modules stop responding to new requests, it is important that you quickly determine why this stoppage is occurring. You can use some of the following techniques to determine whether the problem is a web module problem or an application server environment problem.

If an application server process spontaneously closes, or web modules running on the application server stop responding to new requests:
- Try to Isolate the problem by installing the web modules on different servers, if possible.
- If an application server's process spontaneously closes, there will be an SDUMP for you to analyze.
- Use the Tivoli® performance viewer to determine if any of the application server resources, such as the Java heap, or database connections, have reached their maximum capacity. If there is a resource problem, review the application code for a possible cause:
  - If database connections are being assigned to a request but are not being released when the requests finish processing, ensure that the application code performs a **close()** on any opened **Connection** object within a **finally{}** block.
  - If there is a steady increase in servlet engine threads in use, review application **synchronized** code blocks for possible deadlock conditions.

- If there is a steady increase in a JVM heap size, review application code for memory leak opportunities, such as static (class-level) collections, that can cause objects to never get garbage-collected.
- Enable verbose garbage collection on the application server to help you determine if you have a memory leak problems. This feature adds detailed statements about the amount of available and in-use memory to the JVM error log file.

  To enable up verbose garbage collection:

  1. In the administrative console, click **Servers > Server Types > Application Servers >** *server_name*. Then, in the Server Infrastructure section, click **Java and process management > Process definition > Control > Java virtual machine**, and select **Verbose garbage collection**.
  2. Stop and restart the application server.
  3. Periodically, browse the log file for garbage collection statements. Look for statements beginning with "allocation failure". This string indicates that a need for memory allocation has triggered a JVM garbage collection, to release unused memory. Allocation failures are normal and do not necessarily indicate a problem. However, the statements that follow the allocation failure statement show how many bytes are needed and how many are allocated. If these bytes needed statements indicate that the JVM keeps allocating more memory for its own use, or that the JVM is unable to allocate as much memory as it needs, there might be a memory leak.

IBM Support has documents and tools that can save you time gathering information needed to resolve problems as described in Troubleshooting help from IBM. Before opening a problem report, see the Support page:

- http://www.ibm.com/software/webservers/appserv/zos_os390/support/

Setting a time limit for the completion of RMI/IIOP enterprise bean requests

The Request timeout ORB Service setting determines how long a client waits for a response from an outbound RMI/IIOP enterprise bean invocation. This setting is a server-wide setting that applies to each IIOP locate and request message that is sent as a result of an enterprise bean invocation. When the specified time limit expires, the application that invoked the outbound RMI/IIOP enterprise bean receives an org.omg.CORBA.COMM_FAILURE system exception.

Before you begin

For WebSphere Application Server Version 7 and later, listener ports are deprecated. Therefore you should plan to migrate your WebSphere MQ message-driven bean deployment configurations from using listener ports to using activation specifications. However, you should not begin this migration until you are sure the application does not have to work on application servers earlier than WebSphere Application Server Version 7. In some cases you will continue to use the WebSphere MQ message-driven bean deployment and listener ports and in other case you will use the WebSphere MQ message-driven bean deployment and activation specifications.

The following properties DO NOT apply to activation specification driven message-driven bean deployment. That is, the properties require you to configure them to use the WebSphere MQ message-driven bean deployment and listener ports:

- control_region_mdb_request_timeout
- control_region_mdb_queue_timeout_percent
- server_region_mdb_stalled_thread_dump_action

The follow properties DO apply to activation specification driven message-bean deployment. That is, these properties require you to configure them to use the WebSphere MQ message-driven bean deployment and activation specifications.

- control_region_wlm_dispatch_timeout

- control_region_iiop_queue_timeout_percent
- server_region_iiop_stalled_thread_dump_action

As you follow the instructions to configure these properties, remember what properties apply to listener ports versus activation specifications.

Before you begin, you should:

- Determine your settings for all of the dispatch timers. There are separate dispatch timers for message-driven beans (MDBs), (control_region_mdb_request_timeout), HTTP requests (protocol_http_timeout_output), HTTPS requests (protocol_https_timeout_output), SIP requests (protocol_sip_timeout_output), SIPS requests (protocol_sips_timeout_output), and IIOP requests (control_region_wlm_dispatch_timeout). Because enterprise bean invocations can occur while an application is running under an MDB, a servlet or another enterprise bean, you must make sure that the interval setting for the RMI/IIOP outbound timer is shorter than the interval setting for any of these dispatch timers.
- Understand that, from the perspective of the invoker, remote enterprise bean invocations are synchronous. Therefore, while the invoker waits for a response from the enterprise bean, the invoking thread is blocked. When the RMI/IIOP timer expires, the invoking thread is interrupted and returns to the invoker with a system exception response. If a response from the invoked EJB arrives AFTER the RMI/IIOP timer expires, it is ignored.
- Understand the relationship between the RMI/IIOP outbound timer and transactions. When the RMI/IIOP outbound timer expires and a system exception is returned to the invoker, the EJB container immediately puts any existing global transaction into rollback-only state. However, the invoker still returns any response from the target enterprise bean even though the target enterprise bean's transaction is marked for rollback.

About this task

If the timer expires, a message similar to the following message is issued:

```
BBOO0325W An IIOP request for Class Name 'com.ejb.test.hello.second.EJSRemoteStatelessSayHelloSecond_686a0ff2'
and Method Name 'sayHelloTwo', to 'jobname=BBOS002  asid=0031', has timed out.
SessionHandle=0000000026D9F0480000000A008004FF, Request ID=00000004
```

This message indicates the class and method of the target enterprise bean. If the target enterprise bean was invoked over TCP/IP, the "to" section of the message contains the hostname and the port of the target server. If the target enterprise bean is invoked through optimized local communication, the "to" section of the message contains the target jobname and asid, as shown in the preceding example.

When this message is issued, a corresponding exception trace is generated that includes an entry similar to the following entry:

```
/bbooejsb.cpp+3395 ... BBOO0011W The function ORBEJSBridge::invoke_request(JNIEnv *, bboojorb *,
char *, CORBA::Boolean, CORBA::Request *&, void *)+3395 received CORBA system exception CORBA::COMM_FAILURE.
Error code is C9C26A48
```

The minor code C9C26A48 in this trace entry indicates that the wait time for the RMI/IIOP outbound timer has ended.

If a response to the request is received after the request times out and is no longer on the queue, the following message is issued:

```
BBOO0328I: No Request found for inbound GIOP Response,
          SessionHandle=<hstring>, RequestID=<hstring>.
```

A request or reply is uniquely identified by the session handle and the request ID, and can be used to determine if an earlier BBOO0325W message was received for this request.

To change the length of time that the client waits for a response from an invoked enterprise bean:

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Container service > ORB Service**.
2. Specify, in seconds, an appropriate timer setting in the Request timeout field. The default value is 180 seconds. Example: Specifying a value of 2 in the Request timeout field sets the wait time for the timer to 2 seconds.

   It is recommended that you specify a value that is significantly shorter than the time specified for the dispatch timers. This type of setting enables the invoking application to compensate for nonresponsive enterprise beans before the wait time for the dispatch timers ends. If the wait time for the dispatch timers ends, an EC3 ABEND might occur.

What to do next

Because the org.omg.CORBA.COMM_FAILURE exception is a system exception, the application invoking the enterprise bean is not required to compensate for or retry an expired enterprise bean invocation. However, in certain situations, such as when atomic transactions are not in use by the application, you might want the application to compensate for or retry an expired enterprise bean invocation.

For an application to compensate for or retry an expired enterprise bean invocation, the application must:
- Be running outside of the current global transaction, and
- Catch the org.omg.CORBA.COMM_FAILURE exception.

The following example, which is a snippet of code that is running outside an atomic transaction, illustrates the steps an application must perform if you want that application to compensate for or retry an expired enterprise bean invocation:

```
// This method runs outside a global transaction.
public Data callingMethod() throws ... {
    try{
            InitialContext con = new InitialContext();
            EJBHome home con.lookup(...);
            CalledBean cb = home.create();

    } catch (org.omg.CORBA.COMM_FAILURE cf1){
        // The home create could timeout, so put retry or
        // compensation logic here.

    } catch( CreateException cx){
                throw new ...
    } catch( NamingException nx){
                throw new ...
    } catch(RemoteException ex){
                throw new ...
    }
    try{
        cb.calledMethod(...);

    } catch (org.omg.CORBA.COMM_FAILURE cf2){
// The calledMethod could timeout, so put retry or
// compensation logic here.
    } catch( ... ){
        ...
    }
}

// This method can run in a global transaction.
private void calledMethod(String strKey) throws ... {
    try{
            // business logic here
```

```
        }
    catch ( ... ){
               throw new ...
        }
}
```

Applications that run within the scope of an atomic transaction must suspend that transaction before invoking an enterprise bean if you want that application to be able to compensate, for or retry an expired enterprise bean invocation. Embedding the invocation in a TX_NOTSUPPORTED enterprise bean method is the best way of suspending the current transaction.

Preparing to host applications

Rather than use the default application server provided with the product, you can configure a new server and set of resources.

About this task

The default application server and a set of default resources are available to help you begin quickly. You can choose instead to configure a new server and set of resources. Here is what you need to do in order to set up a runtime environment to support applications.

Procedure

1. Configure an application server.
2. Create a virtual host.
3. Configure a web container. See the *Administering applications and their environment* PDF for more information.
4. Configure an EJB container. See the *Administering applications and their environment* PDF for more information.
5. Create resources for data access. See the *Administering applications and their environment* PDF for more information.
6. Create a JDBC provider and data source. See the *Administering applications and their environment* PDF for more information.
7. Create a URL and URL provider. See the *Administering applications and their environment* PDF for more information.
8. Create a mail session. See the *Administering applications and their environment* PDF for more information.
9. Create resources for session support. See the *Administering applications and their environment* PDF for more information.
10. Configure a Session Manager. See the *Administering applications and their environment* PDF for more information.

What to do next

Test the server and resources.

Configuring an application server, a node, or a cell to use a single network interface

Application servers, by default, are configured to use all of the network interfaces that are available for them to use. You can change this configuration such that an application server only uses a specific network interface. However, you cannot configure it to use a subgroup of interfaces. For example, if you have three ethernet adapters, you cannot configure an application server to use two of the three adapters.

About this task

When an application server is configured to use all network interfaces, if it opens a socket on port 9901 on a machine with two TCP/IP addresses, it opens port 9901 on both IP addresses.

When an application server is configured to use a specific network interface, it only communicates on that one network interface. For example, on a Windows operating system, if an application server opens a socket on port 7842 on an ethernet adapter with an address of 192.168.1.150, the netstat output displays 192.168.1.150.7842 in the Local Address field, indicating that port 7842 is only bound to 192.168.1.150.

If you have more than one network interface and you want to use each one separately, you must have a separate configuration profile for each interface. When network interfaces are used separately, a separate node agent is required for each network interface that has an application server running on it. Two application servers bound to two separate network interfaces on the same machine cannot be in the same node because they have different TCP/IP addresses.

gotcha:

- If you want a specific application server to use a single network interface, perform the following steps for that application server.
- If you want an entire node to use a single network interface, perform the following steps for your node agent and all the application servers in that node.
- If you want an entire cell to use a single network interface, perform the following steps for the deployment manager, node agent, and all the application servers in the node.
- When performing the following steps, do not specify localhost, a loop back address, such as 127.0.0.1, or an * (asterisk) for the TCP/IP addresses.

Procedure

1. Update the com.ibm.CORBA.LocalHost and com.ibm.ws.orb.transport.useMultiHome Object Request Broker (ORB) custom properties.
   a. In the administrative console, navigate to the indicated panel.
      - For an application server, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Container Settings** > **Container services** > **ORB Service**. Then in the Additional Properties section, click **Custom properties**.
      - For a deployment manager, click **System Administration** > **Deployment manager**. In the Additional Properties section, click **ORB Service**. Then, under Additional properties on the **ORB Service** panel, click **Custom properties**.
      - For a node agent, click **System Administration** > **Node agents** > *node_agent* . In the Additional Properties section, click **ORB Service**. Then, under Additional properties on the **ORB Service** panel, click **Custom properties**.
   b. Select the com.ibm.CORBA.LocalHost custom property and specify an IP address or hostname in the Value field. Do not set this property to either localhost or *.

      If the com.ibm.CORBA.LocalHost property is not in the list of already defined custom properties, click **New** and then enter `com.ibm.CORBA.LocalHost` in the Name field and specify an IP address or hostname in the Value field.
   c. Select the com.ibm.ws.orb.transport.useMultiHome custom property and specify `false` in the Value field. If the com.ibm.ws.orb.transport.useMultiHome property is not in the list of already defined custom properties, click **New**, and then enter `com.ibm.ws.orb.transport.useMultiHome` in the Name field and specify `false` in the Value field.
2. Update the daemon_protocol_iiop_listenIPAddress WebSphere variable to indicate the IP addresses to which you want the location service daemon to bind.
   a. In the administrative console, click **Environment** > **WebSphere variables**.

b. Select the DAEMON_protocol_iiop_listenIPAddress variable and specify * to specify bind all, or specify a specific IP address in the Value field. If the DAEMON_protocol_iiop_listenIPAddress variable is not in the list of already defined variables, click **New**, and then enter `DAEMON_protocol_iiop_listenIPAddress` in the Name field and specify the appropriate value in the Value field.

3. Update the Java virtual machine (JVM) com.ibm.websphere.network.useMultiHome custom property for discovery and SOAP connections.

   a. In the administrative console, navigate to the indicated page.

   - For an application server, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Java process management** > **Process definition** > *process_type* > **Java virtual machine** > **Custom properties**.

   - For a deployment manager, click **System Administration** > **Deployment manager** > **Java process management** > **Process definition** > *process_type* > **Java virtual machine** > **Custom properties**.

   - For a node agent, click **System Administration** > **Node agents** > *node_ agent* > **Java process management** > **Process definition** > **Control** > **Java virtual machine** > **Custom properties**.

   b. Select the com.ibm.websphere.network.useMultiHome custom property and specify `false` in the Value field. If the com.ibm.websphere.network.useMultiHome property is not in the list of already defined custom properties, click **New** and then enter `com.ibm.websphere.network.useMultiHome` in the Name field and specify `false` in the Value field.

4. Update the host name for TCP/IP connections.

   a. In the administrative console, navigate to the indicated page.

   - For an application server, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name*, and then, in the Additional Properties section, click **Ports**.

   - For a deployment manager, click **System Administration** > **Deployment manager**, and then, in the Additional Properties section, click **Ports**.

   - For a node agent, click **System Administration** > **Node agents** > *node_ agent*, and then, in the Additional Properties section, click **Ports**.

   b. Update the Host field for each of the listed ports to the value specified for the com.ibm.CORBA.LocalHost ORB custom property in the first step. When you finish, none of the entries listed in the Host column should contain an * (asterisk).

5. Change the Initial State setting for each of the Version 5 JMS servers to `Stopped` .

   a. In the administrative console, click **Servers** > **Server Types** > **Version 5 JMS servers**.

   b. Click one of the listed JMS servers, and change the value specified for the Initial State field to `Stopped`.

   c. Repeat the previous step until the Initial State setting for all of the listed JMS servers is `Stopped`.

6. Change the Initial State setting for each of the listener ports to `Stopped` .

   a. In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name*.

   b. Under Communications, click **Messaging** > **Message Listener Service** > **Listener Ports**.

   c. Click one of the listed listener ports and change the value specified for the Initial State field to `Stopped`.

   d. Repeat the previous step until the Initial State setting for all of the listed listener ports is `Stopped`.

7. Save your changes.

   a. In the administrative console, click **System administration** > **Save Changes to Master Repository**.

   b. Select Synchronize changes with nodes, and then click **Save**.

8. Stop and restart all the affected servers, node agents, and the deployment manager.

Results

You have configured an installation of WebSphere Application Server to communicate on one, and only one network interface on a machine that has more than one network interface.

Example

This example creates two nodes, each using a separate network interface, on a machine that has at least two network interfaces:

1. Use the Profile Management tool to create an application server and federate it into the desired cell.
2. Use the Profile Management tool to create an application server profile, specifying a host name that is different than the host name used for the previously created application server. Federate this application server into the desired cell.
3. Start the node agent and application server that are configured to the first network interface. Follow the preceding steps for the node agent and application server to prepare this node to communicate on the network interface you specified when you configured this application server.
4. Start the second node agent and application server. Follow the preceding steps for the node agent and application server to prepare this node to communicate only on the network interface that you specified when you configured the second application server.
5. Stop all of the node agents and application servers that you created in this example.
6. Restart all of these node agents and application servers.

You have two separate nodes running on two different network interfaces.

What to do next

If you are using a stand-alone Java client or server to communicate with WebSphere Application Server, and you are using the WebSphere Application Server Software Development Kit (SDK), add the following properties to your Java command to enable the ORB for your application to communicate with a specific network interface.

```
-Dcom.ibm.ws.orb.transport.useMultiHome=false
-Dcom.ibm.CORBA.LocalHost=host_name
```

host_name is the TCP/IP address or *hostname* of the network interface for the ORB to use.

gotcha: Do not set *host_name* to localhost, a loop back address, such as 127.0.0.1, or an * (asterisk).

Configuring application servers for UCS Transformation Format

You can use the client.encoding.override=UTF-8 JVM argument to configure an application server for UCS Transformation Format. This format enables an application server to handle most character encodings, including specialized mathematical and technical symbols.

About this task

The client.encoding.override=UTF-8 argument is provided for backwards compatibility. You should only specify this argument if you require multiple language encoding support in the administrative console and there is no other way for you to set the request character encoding required to parse post and query strings.

Before configuring an application server for UCS Transformation Format, you should try to either:

- Explicitly set the ServletRequest Encoding inside of the JSP or Servlet that is receiving the POST and or query string data, which is the preferred J2EE solution, or

- Enable the autoRequestEncoding, option, which uses the client's browser settings to determine the appropriate character encoding. Older browsers might not support this option.

gotcha: If the client.encoding.override=UTF-8 JVM argument is specified, the autoRequestEncoding option does not work even if it is enabled. Therefore, when an application server receives a client request, it checks to see if the charset option is set on the content type header of the request:
1. If it is set, the application server uses the content type header for character encoding.
2. If it is not set, the application server uses the character encoding that is specified for the default.client.encoding system property.
3. If neither charset nor the default.client.encoding system property is set, the application server uses the ISO-8859-1 character set.

The application server never checks for an Accept-Language header. However, if the autoRequestEncoding option is working, the application server checks for an Accept-Language header before checking to see if a character encoding is specified for the default.client.encoding system property.

To configure an application server for UCS Transformation Format:

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere application servers**, and select the server that you want to enable for UCS Transformation Format.
2. Then, in the Server Infrastructure section, click **Java and process management > Process definition > Control > Java virtual machine**.
3. Specify `-Dclient.encoding.override=UTF-8` for the **Generic JVM Arguments** property, and click **OK**. When this argument is specified, UCS Transformation Format is used instead of the character encoding that would be used if the autoRequestEncoding option was in effect.
4. Click **Save** to save your changes.
5. Restart the application server.

Results

The application server uses UCS Transformation Format for encoding.

Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root

    Refers to the top directory for a WebSphere Application Server node.

    The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.

    The default varies based on node type. Common defaults are `configuration_root`/`AppServer` and `configuration_root`/`DeploymentManager`.

configuration_root

    Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.

The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.

The default is `/wasv8config/`*cell_name*`/`*node_name*.

plug-ins_root
> Refers to the installation root directory for Web Server Plug-ins.

profile_root
> Refers to the home directory for a particular instantiated WebSphere Application Server profile.
>
> Corresponding product variables are *server.root* and *user.install.root.*
>
> In general, this is the same as *app_server_root*`/profiles/`*profile_name*. On z/OS, this will always be *app_server_root*`/profiles/default` because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root
> Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.
>
> The corresponding product variable is *smpe.install.root.*
>
> The default is `/usr/lpp/zWebSphere/V8R0`.

Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root
> Refers to the top directory for a WebSphere Application Server node.
>
> The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.
>
> The default varies based on node type. Common defaults are *configuration_root*`/AppServer` and *configuration_root*`/DeploymentManager`.

configuration_root
> Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.
>
> The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.
>
> The default is `/wasv8config/`*cell_name*`/`*node_name*.

plug-ins_root
> Refers to the installation root directory for Web Server Plug-ins.

profile_root
> Refers to the home directory for a particular instantiated WebSphere Application Server profile.
>
> Corresponding product variables are *server.root* and *user.install.root.*
>
> In general, this is the same as *app_server_root*`/profiles/`*profile_name*. On z/OS, this will always be *app_server_root*`/profiles/default` because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root

Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.

The corresponding product variable is *smpe.install.root*.

The default is `/usr/lpp/zWebSphere/V8R0`.

Creating generic servers

A generic server is a server that is managed in the WebSphere Application Server administrative domain even though the server is not a server that is supplied by WebSphere Application Server. The WebSphere Application Server generic servers function enables you to define a generic server as an application server instance within the WebSphere Application Server administration, and associate it with a non-WebSphere WebSphere Application Server or process.

About this task

Generic application servers must be non-Java application processes that are either a started task or a shell script. You cannot create a Java application as a generic server for the product.

The following processes can be created as a generic server provided that they are either started tasks or a shell scripts:
- A C or C++ server or process
- A CORBA server
- A Remote Method Invocation (RMI) server

You can use the wsadmin tool or the administrative console to create a generic server.

Procedure

- **Create a non-Java application as a generic server.** The following steps describe how to use the administrative console to create a non-Java application as a generic application server.

  1. Select **Servers** > **Generic servers**
  2. Click **New**.
  3. Type in a name for the generic server.

     The name must be unique within the node. It is recommended that you use a naming scheme that makes it easy to distinguish your generic application servers from regular WebSphere Application Server servers.
  4. Click **Next**
  5. Click **Finish**. The generic server now appears as an option on the **Generic servers** page in the administrative console.
  6. On the **Generic servers** page, click on the name of the generic server.
  7. Under Additional Properties, click **Process Definition**.
  8. In the Executable name field, enter the name of the non-java process that is launched when you start this generic server.

     For example, if you are using a perl script as a generic server, enter the path to the perl.exe module in the Executable name field.

     If you have additional arguments, such as the name of the perl script and its parameters, enter them in the Executable arguments field. Multiple arguments must be separated by carriage returns. Use the Enter key on your keyboard to create these carriage returns in the Executable arguments field. The following example illustrates how a perl script application that requires two arguments should appear in this field:

     ```
     perl_application.pl
     arg1
     arg2
     ```

> **gotcha:** The Executable target type and Executable target properties are not used for non-Java applications. Executable target type and Executable target properties are only used for Java applications.

9. Click **OK**.

- **Create a Java application as a generic server:** The following steps describe how to use the administrative console to create a Java application as a generic application server.

   1. Select **Servers > Server Types > Generic servers**

   2. Click **New**.

   3. Type in a name for the generic server.

      The name must be unique within the node. It is highly recommended that you use a naming scheme that makes it easy to distinguish your generic application servers from regular WebSphere Application Server servers.

   4. Click **Next**

   5. Click **Finish**. The generic server now appears as an option on the **Application servers** page in the administrative console.

   6. Click **Finish**. The generic server now appears as an option on the **Generic servers** page in the administrative console.

   7. On the Generic servers page, click on the name of the generic server.

   8. Under Additional Properties, click **Process definition**.

   9. In the Executable name field under General Properties, enter the path for the WebSphere Application Server default JVM, `${JAVA_HOME}/bin/java`, which is used to run the Java application when you start this generic server.

   10. In the Executable target type field under General Properties, select whether a Java class name, **JAVA_CLASS**, or the name of an executable JAR file, **EXECUTABLE_JAR**, is used as the executable target of this Java process. The default value for the product is **JAVA_CLASS**.

   11. In the Executable target field under General Properties, enter the name of the executable target. Depending on the executable target type, this is either a Java class containing a main() method, or the name of an executable JAR file.) The default value for WebSphere Application Server is com.ibm.ws.runtime.WsServer.

   12. Click **OK**.

   > **Note:** If the generic server is to run on an application server other than a WebSphere Application Server server, leave the Executable name field set to the default value and specify the Java class containing the main function for your application serve in the Executable target field.

What to do next

After you define a generic server, use the Application Server administrative console to start, stop, and monitor the associated non-WebSphere Application Server server or process when stopping or starting the applications that rely on them.

gotcha: You can use either the **Terminate** or **Stop** buttons in the administrative console to stop any application server, including a generic application server.

Starting and terminating generic application servers

This topic describes how to start and terminate generic servers.

About this task

For the WebSphere Application Server (base) product, you cannot use the administrative console to create a generic application server definition or use the administrative console to start, stop or, in any way, control or manage that application server. The Base product administrative console can only be used to create

server definitions and, if necessary, adjust the server definitions that it creates. To manage Base generic application servers, you need to use the command prompt environment, such as `startServer <genericServerName>` or `stopServer <genericServerName>` or `serverStatus <genericServerName>`. The wsadmin tool has no functional role in the Base application server environment.

If you create a generic server in a WebSphere Application Server, Network Deployment environment, you can use the administrative console to start and terminate this server.

Procedure

1. Start a generic application server.

   There are two ways to start a generic server in a WebSphere Application Server, Network Deployment environment. You can use the managed bean (MBean) NodeAgent launchProcess operation of the wsadmin tool, or you can use the administrative console. To use the administrative console:

   a. In the administrative console, click **Servers > Server Types > Generic servers**.

   b. Select the name of the generic server you want to start, and then click **Start**.

   c. View the **Status** value and any messages or logs to see whether the generic server starts.

2. Terminate generic servers.

   There are two ways to terminate a generic server in a WebSphere Application Server, Network Deployment environment. You can use the MBean terminate launchProcess operation of the wsadmin tool, or you can use the administrative console. To use the administrative console:

   a. In the administrative console, click **Servers > Server Types > Generic servers**.

   b. Select the check box beside the name of the generic server, and then click **Terminate** or **Stop**.

   c. View the **Status** value and any messages or logs to see whether the generic server terminates.

Generic server settings

Use this page to view or change the settings of a generic server.

A generic server is a server that is managed in the product administrative domain, although it is not a server that is provided with the product. The generic server can be any server or process that is necessary to support the Application Server environment, including a Java server, a C or C++ server or process, or a Remote Method Invocation (RMI) server.

To view this administrative console page, click **Servers > Server Types > Generic servers > *server_name***.

On the **Configuration** tab, you can edit fields. On the **Runtime** tab, you can look at read-only information. The **Runtime** tab is available only when the server is running.

Name

Specifies a logical name for the generic server.

Generic server names must be unique within a node. For multiple nodes within a cluster, you can have different generic servers with the same server name as long as the server and node pair are unique. For example, a server named server1 in a node named node1 in the same cluster with a server named server1 in a node named node2 is allowed. Configuring two servers named server1 in the same node is not allowed. The product uses the server name for administrative actions, such as referencing the server in scripting.

It is highly recommended that you use a naming scheme that makes it easy to distinguish your generic application servers from regular product application servers. This will enable you to quickly determine whether to use the Terminate or Stop button in the administrative console to stop a specific application server.

You must use the Terminate button to stop a generic application server.

Data type   String
Default

Configuring transport chains

A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP or HTTP. Network ports can be shared among all of the channels within a chain. The channel framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

Before you begin

Ensure that a port is available for the new transport chain. If you need to set up a shared port, you must:
- Use wsadmin commands to create your transport chain.
- Make sure that all channels sharing that port have the same discrimination weight assigned to them.

About this task

You need to configure transport chains to provide networking services to such functions as the service integration bus component of IBM service integration technologies, the caching proxy, and the high availability manager core group bridge service.

You can use either the administrative console or wsadmin commands to create a transport chain. If you use the administrative console, complete the following steps:

Procedure

1. In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* or **Servers** > **Server Types** > **WebSphere proxy servers** > *server_name*, and then select one of the following options, depending on the type of chain you are creating:

   For application servers, in the Container settings section select one of the following options:
   - Click **SIP Container Settings** > **SIP container transport chains**.
   - Click **Web container settings** > **Web container transport chains**.
   - Click **Container services** > **ORB service** > **ORB service transport chains**.
   - In the Server messaging section, click either **Messaging engine inbound transports** or **WebSphere MQ link inbound transports**.

   For proxy servers, under HTTP proxy server settings, click **Proxy server transports** and select either **HTTPS_PROXY_CHAIN** or **HTTP_PROXY_CHAIN**. Then click **HTTP proxy inbound channel**.

2. Click **New**.

   The Create New Transport Chain wizard initializes. During the transport chain creation process, you are asked to:
   - Specify a name for the new chain.
   - Select a transport chain template
   - Select a port, if one is available to which the new transport chain is bound. If a port is not available or you want to define a new port, specify a port name, the host name or IP address for that port, and a valid port number.

     gotcha: If you are configuring a chain that contains a TCP channel, the wizard displays a list of configured TCP channels and a list of the ports that the listed TCP channels are not using. You must select one of the ports that none of the other TCP channels are using.

Similarly, if you are configuring a transport chain that contains a UDP channel, the wizard displays a list of already configured UDP channels and a list of the ports that these UDP channels are not using. You must select one of the ports that none of the other UDP channels are using.

When you click **Finish**, the new transport chain is added to the list of defined transport chains on the **Transport chain** panel.

3. Click the name of a transport chain to view the configuration settings that are in effect for the transport channels contained in that chain.

To change any of these settings, complete the following actions:

a. Click the name of the channel whose settings you need to change.

b. Change the configuration settings.

Some of the settings, such as the port number, are determined by what is specified for the transport chain when it is created and cannot be changed.

c. Click on **Custom properties** to set any custom properties that are defined for your system.

4. When you your configuration changes, click **OK**.

5. Stop the application server and start it again.

You must stop the application server and start it again before your changes take effect.

What to do next

Update any routines you have that issue a call to start transports during server startup. When a routine issues a call to start transports during server startup, the product issues an error message.

Note: If you create a new web container transport chain, the initial value for the writeBufferSize attribute is 8192, which is too small for most web container transport chains. It is recommended that you use the administrative console or wsadmin scripting to change the value of this attribute to 32768. Complete the following steps if you want to use the administrative console, to change the value of this attribute:

1. Click **Servers** and expand **Server Types**.

2. Click **WebSphere application servers** > *server_name*.

3. Expand **Web Container Settings** and click **Web container transport settings >** *transport_chain_name*.

4. Click **Web container inbound channel**.

5. Specify `32768` in the **Write buffer size** field, and click **OK**.

6. Click **Save**.

To change the value using wsadmin scripting, see the documentation about working with Web container inbound channel properties files.

Transport chains

Transport chains represent a network protocol stack that is used for I/O operations within an application server environment.

Transport chains are part of the channel framework function that provides a common networking service for all components, including the service integration bus component of IBM service integration technologies, WebSphere Secure Caching Proxy, and the high availability manager core group bridge service.

A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP, DCS. or HTTP. Network ports can be shared among all of the channels within a chain. The channel framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

Remember: If you have a routine that issues a call to start transports during server startup, unless you have a mixed-node environment and that server is running in either a Version 6.x node, you must modify your routine to issue a call to start transport chains instead of the transports. The product issues an error message if it receives a call to start transports for a server that is not running in a Version 6.x node.

The transport chain configuration settings determine which I/O protocols are supported for that chain. Following are some of the more common types of channels. Custom channels that support requirements unique to a particular customer or environment can also be added to a transport chain.

DCS channel

Used in a WebSphere Application Server, Network Deployment environment by the core group bridge service, the data replication service (DRS), and the high availability manager to transfer data, objects, or events among application servers.

HTTP inbound channel

Used to enable communication with remote servers. It implements the HTTP 1.0 and 1.1 standards and is used by other channels, such as the web container channel, to serve HTTP requests and to send HTTP specific information to servlets expecting this type of information.

HTTP inbound channels are used instead of HTTP transports to establish the request queue between a web server plug-in, and a web container in which the web modules of an application reside.

HTTP proxy inbound channel

Used to handle HTTP requests between a proxy server and application server nodes.

HTTP Tunnel channel

Used to provide client applications with persistent HTTP connections to remote hosts that are either blocked by firewalls or require an HTTP proxy server, including authentication, or both. An HTTP Tunnel channel enables the exchange of application data in the body of an HTTP request or response that is sent to or received from a remote server. An HTTP Tunnel channel also enables client-side applications to poll the remote host and to use HTTP requests to either send data from the client or to receive data from an application server. In either case, neither the client nor the application server is aware that HTTP is being used to exchange the data.

JFAP channel

Used by the Java Message Service (JMS) server to create connections to JMS resources on a service integration bus.

MQ channel

Used in combination with other channels, such as a TCP channel, within the confines of WebSphere MQ support to facilitate communications between a service integration bus and a WebSphere MQ client or queue manager.

ORB Service channel

Used in combination with other channels, such as a TCP channel, to handle CORBA and RMI/IIOP messages for the ORB Service. It enables clients to make requests and receive responses from servers in a network-distributed environment.

SIP channel

Used to create a bridge in the transport chain between a session initiation protocol (SIP) inbound channel, and a servlet and JavaServer Page engine.

SIP container inbound channel

Used to handle communication between the SIP inbound channel and the SIP servlet container.

SIP inbound channel

> Used to handle inbound SIP requests from a remote client.

SSL channel

> Used to associate an Secure Sockets Layer (SSL) configuration repertoire with the transport chain. This channel is only available when SSL support is enabled for the transport chain. An SSL configuration repertoire is defined in the administrative console, under security, on the **SSL configuration repertoires** > **SSL configuration repertoires** page.

TCP channel

> Used to provide client applications with persistent connections within a Local Area Network (LAN) when a node uses transmission control protocol (TCP) to retrieve information from a network.

UDP channel

> Used to provide client applications with persistent connections within a Local Area Network (LAN) when a node uses user datagram protocol (UDP) to retrieve information from a network.

Web container channel

> Used to create a bridge in the transport chain between an HTTP inbound channel and a servlet and JavaServer Page (JSP) engine.

HTTP transport collection

Use this page to view or manage HTTP transports. Transports provide request queues between web server plug-ins and web containers in which the web modules of applications reside. When you request an application in a web browser, the request is passed to the web server, then along the transport to the web container.

transition: You can use HTTP transports only on a V6.0.x node in a mixed cell environment. You must use HTTP transport channels instead of HTTP transports to handle your HTTP requests on all of your other nodes.

The use of IPv6 (Internet Protocol Version 6) and WS-AT (Web Services Atomic Transactions) are not supported on HTTP transports; they are only supported on HTTP transport channel chains.

To view the HTTP Transport administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Web container settings > Web container > HTTP transports**.

Host

Specifies the host IP address to bind for transport. If the application server is on a local machine, the host name might be `localhost`.

Port

Specifies the port to bind for transport. The port number can be any port that currently is not in use on the system. The port number must be unique for each application server instance on a given machine.

For z/OS, a maximum of two ports, one for HTTP requests and one for HTTPS requests, is allowed for each process configured as an HTTP transport. Additional ports can be configured as HTTP transport chains Additional ports can be configured as HTTP transport channels.

SSL Enabled

Specifies whether to protect transport connections with Secure Sockets Layer (SSL). The default is not to use SSL.

HTTP transport settings

Use this page to view and configure an HTTP transport. The name of the page might be that of an SSL setting such as DefaultSSLSettings. This page is not available if you do not have an HTTP transport defined for your system.

depfeat: You can use HTTP transports only on a V6.0.x node in a mixed cell environment. You must use HTTP transport channels instead of HTTP transports to handle your HTTP requests on all of your other nodes.

The use of IPv6 (Internet Protocol Version 6) and WS-AT (Web Services Atomic Transactions) are not supported on HTTP transports; they are only supported on HTTP transport channel chains.

If you have HTTP transports defined for your system, in the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*, and then in the Container Settings section, click **Web container > HTTP transports >** *host_name* to view or change the settings for your HTTP transport.

Host

Specifies the host IP address to bind for transport.

If the application server is on a local machine, the host name might be `localhost`.

Data type	String

Port

Specifies the port to bind for transport. Specify a port number between 1 and 65535. The port number must be unique for each application server on a given machine.

Data type	Integer
Range	1 to 65535

SSL Enabled

Specifies whether to protect transport connections with Secure Sockets Layer (SSL). The default is not to use SSL.

Data type	Boolean
Default	false

SSL

Specifies the Secure Sockets Layer (SSL) settings type for SSL connections. The options include one or more SSL settings that are defined in the Security Center; for example, DefaultSSLSettings, ORBSSLSettings, or LDAPSSLSettings.

Data type	String
Default	An SSL setting defined in the Security Center

HTTP transport custom properties

You can use the administrative console to set custom properties for an HTTP transport. The HTTP transport custom properties administrative console page only appears if you have an HTTP transport defined for your system.

gotcha: You can use HTTP transports only on a version previous to a Version 6.x node in a mixed cell environment. This panel shows if you are using nodes from a version previous to Version 6.x and have the script compatibility mode enabled. You must use HTTP transport channels instead of HTTP transports to handle your HTTP requests on all of your other nodes. The topic *HTTP Tunnel transport channel custom property* describes the custom properties that you can specify for an HTTP transport channel.

The use of IPv6 (Internet Protocol Version 6) and WS-AT (Web Services Atomic Transactions) are not supported on HTTP transports; they are only supported on HTTP transport channel chains.

If you are using HTTP transports, you can set the following custom properties on either the web container or HTTP transport custom properties page in the administrative console. When set on the web container custom properties page, all transports inherit the properties. Setting the same properties on a transport overrides like settings defined for a web container.

To specify custom properties for a specific transport on the HTTP transport using the administrative console, complete the following steps:
1. Click **Servers** > **Server Types** > **WebSphere application servers** > *server_name*.
2. Under Container Settings section, expand **Web container settings** and click **Web container** > **HTTP transport**.
3. Select a host.
4. In the Additional Properties section, select **Custom Properties**.
5. On the custom properties page, click **New**.
6. On the settings page, enter the property you want to configure in the **Name** field and the value you want to set it to in the **Value** field.
7. Click **Apply** or **OK**.
8. Click **Save** on the console task bar to save your configuration changes.
9. Restart the server.

Following is a list of custom properties provided with the product. These properties are not shown on the settings page for an HTTP transport.
- "ConnectionIOTimeOut"
- "ConnectionKeepAliveTimeout"
- "ConnectionResponseTimeout" on page 398
- "MaxKeepAliveRequests" on page 398
- "MutualAuthCBindCheck" on page 398
- "RemoveServerHeader" on page 399
- "ResponseBufferSize" on page 399
- "ServerHeader" on page 399
- "ServerHeaderValue" on page 399
- "SoLingerValue" on page 399
- "TcpNoDelay" on page 400
- "Trusted" on page 400
- "UseSoLinger" on page 400

ConnectionIOTimeOut:

Use the ConnectionIOTimeOut property to specify how long the J2EE server waits for an I/O operation to complete. Set this variable for each of the HTTP transport definitions on the server. You will need to set this variable for both SSL transport and non-SSL transport. Specifying a value of zero disables the time out function.

Data type   Integer
Default   120 seconds for the z/OS platform

ConnectionKeepAliveTimeout:

Use the ConnectionKeepAliveTimeout property to specify the maximum number of seconds to wait for the next request on a keep alive connection.

Data type	Integer
Default	30 seconds for the z/OS platform

ConnectionResponseTimeout:

Use the ConnectionResponseTimeout property to set the maximum amount of time, in seconds, that the server waits for an application component to respond to an HTTP request. Set this variable for each of the HTTP transport definitions on the server. You must set this variable for both SSL transport and non-SSL transport. If the response is not received within the specified length of time, the servant might fail with ABEND EC3 and RSN=04130007. Setting this timer prevents client applications from waiting for a response from an application component that might be deadlocked, looping, or encountering other processing problems that cause the application component to hang.

This property can be set for both HTTP transports and HTTP transport channels.

Use the server custom properties protocol_http_timeout_output_recovery, and protocol_https_timeout_output_recovery, to indicate the recovery action that you want taken on timeouts for requests received over the HTTP and HTTPS transports.

Data type	Integer
Default	300 seconds

MaxKeepAliveRequests:

Use the MaxKeepAliveRequests property to specify the maximum number of requests which can be processed on a single keep alive connection. This parameter can help prevent denial of service attacks when a client tries to hold on to a keep-alive connection. The Web server plug-in keeps connections open to the application server as long as it can, providing optimum performance.

On the z/OS platform, when this property is set to 0 (zero), the connection is closed after every request.

Data type	Integer
Default	50 requests for the z/OS platform

MutualAuthCBindCheck:

This property is only valid on the z/OS platform. Use the MutualAuthCBindCheck property to specify whether or not a client certificate should be resolved to a SAF principal. If this property is set to `true`, all SSL connections from a browser must have a client certificate, and the user ID associated with that client certificate must have RACF CONTROL authority for CB.BIND.servername. If these conditions are not met, the connection will be closed. Issue the following RACF command to give the user ID associated with that client certificate RACF CONTROL authority:

```
PERMIT CB.BIND.<optional SAF profile prefix>.clustername CLASS(CBIND) ID(clientCertUserid) ACCESS(CONTROL)
```

Data type	String
Value	true or false
Default	false

Important: If you set MutualAuthCBindCheck as a HTTP Transport Custom Property, it does not come into effect for the transport channels. To have MutualAuthCBindCheck come into effect for the transport channels, you have to define it as a web container custom property. See the

information about web container custom properties for information on defining MutualAuthCBindCheck as a web container custom property.

RemoveServerHeader: Use this property to specify whether an existing server header is removed before a response message is sent. If this property is set to `true`, the value specified for the ServerHeaderValue property is ignored.

Data type	String
Value	true or false
Default	false

Attention: This custom property takes effect on the web container level only. You cannot set it on the transport level. To set this custom property, see Modifying the default web container configuration.

ResponseBufferSize:

This property is used to specify, in bytes, the default size of the initial buffer allocation for the response buffer. When the buffer fills up, a flush for this buffer space will automatically occur. If a value is not specified for this property, the default response buffer size of 32K bytes is used.

The setBufferSize() API method can be used to override the value specified for this custom property at the individual servlet level.

Data type	Integer
Default	32000 bytes

ServerHeader:

This property is only valid for the z/OS platform. Use the ServerHeader property to suppress the server HTTP header (Server:) in responses. When the server header custom property is not specified, the default is equal to a setting of `true` and the server header is included in the HTTP response. Set this property to `false` if you want to prevent the inclusion of the server header.

Data type	String
Value	true or false
Default	true

ServerHeaderValue: Use this property to specify a server header this is added to outgoing response messages if server header is not already provided. This property is ignored if the RemoveServerHeader property is set to true.

Data type	string
Default	WebSphere Application Server/*x.x*
	x.x is the version of WebSphere Application Server that you are using.

Attention: This custom property takes effect on the web container level only. You cannot set it on the transport level. To set this custom property, see Modifying the default web container configuration.

SoLingerValue: Use this property to specify, in seconds, the amount, that the socket close operation waits for data contained in the TCP/IP send buffer to be sent. This property is ignored if the UseSoLinger property is set to `false`.

Data type	Integer
Default	20 seconds

TcpNoDelay: Use this property to set the socket TCP_NODELAY option which enables and disables the use of the TCP Nagle algorithm for connections received on this transport. When this property is set to `true`, use of the Nagle algorithm is disabled.

Data type	String
Value	true or false
Default	true

Trusted: Use the Trusted property to indicate that the application server can use the private headers that the web server plug-in adds to requests.

Data type	String
Value	true or false
Default	false

Important: This property must be set to false for Secure Sockets Layer (SSL) client certificate
authentication to work.

UseSoLinger: Use this property to set the socket SO_LINGER option. This property configures whether the socket close operation waits until all of the data contained in the TCP/IP send buffer is sent before closing a connection. If this property is set to `true`, and the time expires before the all of the content of the send buffer sent, any data remaining in the send buffer is lost.

The SoLingerValue property is ignored if this property is set to `false`.

Data type	String
Value	true or false
Default	true

Transport chains collection

Use this page to view or manage transport chains. Transport chains enable communication through transport channels, or protocol stacks, which are usually socket based.

A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP or HTTP. Network ports can be shared among all of the channels within a chain. The Channel Framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

The **Transport chains** page lists the transport chains defined for the selected application server. Transport chains represent network protocol stacks operating within this application server.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Ports**. Click on **View associated transports** for the port whose transport chains you want to view.

Name
Specifies a unique identifier for the transport chain. The name must consist of alphanumeric or national language characters and can start with a number. The name must be unique within the product configuration. Click on the name of a transport chain to change its configuration settings.

Enabled

When set to true, indicates that the transport chain is activated at application server startup.

Host

Specifies the host IP address to bind for the transport chain. If the application server is on a local machine, the host name might be localhost.

Port

Specifies the port to bind for the transport chain. The port number can be any port that is not already bound to another transport chain.

SSL Enabled

When enabled, users are notified that there is a channel that enables Secure Sockets Layer (SSL) in the listed transport chain. When SSL is enabled, all traffic going through this transport is encrypted and digitally secured.

Transport chain settings

Use this page to view a list of the types of transport channels configured for the selected transport chain. A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP, HTTP, or DCS.

To view this administrative console page, click **Servers > Server Types** , and then click either **WebSphere application servers** or **WebSphere proxy servers**. Click a server name, and then click **Ports > View associated transports** for the port whose transport chains you want view, and then click the name of a specific chain.

Name

Specifies the name of the selected transport chain.

You can edit this field to rename this transport chain. However, remember that the name must be unique within the product configuration.

Enabled

When checked, this transport chain is activated at application server or proxy server startup.

Transport channels

Lists the transport channels configured for this transport chain and their configuration settings. Click the name of a transport channel to view the configuration settings for that channel.

HTTP tunnel transport channel settings

Use this page to view and configure an HTTP tunnel transport channels. Inbound connections sent through this channel are tunneled over HTTP, allowing intermediates to view this data as the body of an HTTP message instead of in its natural format. This type of channel is often used to circumvent firewalls with protocol restrictions.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Ports**. Click on **View associated transports** for the port associated with the HTTP Tunnel transport channel whose settings you want to look at.

Transport channel name

Specifies the name of the HTTP tunnel transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels within the product environment. For example, an HTTP tunnel transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type   string

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type   Positive integer
Default   0

HTTP transport channel settings

Use this page to view and configure an HTTP transport channel. This type of transport channel handles HTTP requests from a remote client.

An HTTP transport channel parses HTTP requests and then finds an appropriate application channel to handle the request and send a response.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Ports**. Locate the port for the HTTP channel whose settings you want to view or configure, and click **View associated transports**. Click the name of the transport chain that includes this HTTP transport, and then click the name of the HTTP transport channel.

Transport channel name

Specifies the name of the HTTP transport channel.

The name field cannot contain any of the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels in your system. For example, an HTTP transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type   String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled, and the transport chain includes multiple channels to which it might forward data. The channel in the chain that has the lowest discrimination weight is the first channel that looks at incoming data and determines whether it owns that data.

Data type   Positive integer
Default   0

Read timeout

Specifies the amount of time, in seconds, that the HTTP transport channel waits for a read request to complete on a socket after the first read occurs. The read being waited for could be part of the body of the read request, such as a POST, or part of the headers, if all of the headers are not read as part of the first read that occurs on the socket for this request.

transition: The value specified for this property, in conjunction with the value specified for the Write timeout property, provides the timeout functionality that the ConnectionIOTimeout custom property provided in previous releases.

Data type	Integer
Default	60 seconds

Write timeout

Specifies the amount of time, in seconds, that the HTTP transport channel waits on a socket for each portion of the response data to be transmitted. This timeout typically only occurs in situations where the writes are lagging behind new requests. This situation can occur when a client has a low data rate or the network interface card (NIC) for the server is saturated with I/O.

transition: The value specified for this property, in conjunction with the value specified for the Read timeout property, provides the timeout functionality that the ConnectionIOTimeout custom property provided in previous releases.

If some of your clients require more than 300 seconds to receive data being written to them, change the value specified for the Write timeout parameter. Some clients are slow and require more than 300 seconds to receive data that is sent to them. To ensure they are able to obtain all of their data, change the value specified for this parameter to a length of time in seconds that is sufficient for all of the data to be received. Make sure that if you change the value of this setting, that the new value still protects the server from malicious clients.

Data type	Integer
Default	60 seconds

Persistent timeout

Specifies the amount of time, in seconds, that the HTTP transport channel allows a socket to remain idle between requests.

transition: The value specified for this property provides the timeout functionality that the ConnectionKeepAliveTimeout custom property provided in previous releases.

Data type	Integer
Default	30 seconds

Use persistent (keep-alive) connections

When selected, specifies that the HTTP transport channel connections are left open between requests. Leaving the connections open can save setup and tear down costs of sockets if your workload has clients that send multiple requests.

If your clients only send single requests over substantially long periods of time, it is probably better to disable this option and close the connections right away rather than to have the HTTP transport channel setup the timeouts to close the connection at some later time.

The default value is true, which is typically the optimal setting.

gotcha: If a value other than 0 is specified for the maximum persistent requests property, the Use persistent (keep-alive) connections property setting is ignored.

Unlimited persistent requests per connection

When selected, specifies that the number of persistent requests per connection is not limited.

Maximum persistent requests per connection

When selected, specifies that the number of persistent requests per connection is limited to the number specified for the Maximum number of persistent requests property. This property setting is ignored if the Use persistent (keep-alive) connections property is not enabled.

Change the value specified for the Maximum persistent requests parameter to increase the number of requests that can flow over a connection before it is closed. When the Use persistent connections option is enabled, the Maximum persistent requests parameter controls the number of requests that can flow over a connection before it is closed. The default value is 100. This value should be set to a value such that most, if not all, clients always have an open connection when they make multiple requests during the same session. A proper setting for this parameter helps to eliminate unnecessary setting up and tearing down of sockets.

For test scenarios in which the client will never close a socket or where sockets are always proxy or web servers in front of your application server, a value of -1 disables the processing, which limits the number of requests over a single connection. The persistent timeout still shuts down some idle sockets and protect your server from running out of open sockets.

Related Information: The behavior of persistence is the same as keep-alive connections from the HTTP Transports. The MaxKeepAliveConnections setting, which specifies the maximum number of concurrent keep alive (persistent) connections across all HTTP transports, and the thread pool size are not directly related to persistence. Persistence operates independently of the MaxKeepAliveConnections setting and thread pool size settings.

Maximum persistent requests per connection

Specifies the maximum number of persistent requests that are allowed on a single HTTP connection. You can add a value to this field only if the **Maximum persistent requests per connection** property is selected.

When the Use persistent connections option is enabled, the Maximum persistent requests parameter controls the number of requests that can flow over a connection before it is closed. The default value is 100. This value should be set to a value such that most, if not all, clients always have an open connection when they make multiple requests during the same session. A proper setting for this parameter helps to eliminate unnecessary setting up and tearing down of sockets.

For test scenarios in which the client will never close a socket or where sockets are always proxy or web servers in front of your application server, a value of -1 will disable the processing which limits the number of requests over a single connection. The persistent timeout will still shutdown some idle sockets and protect your server from running out of open sockets.

If a value of 0 or 1 is specified, only one request is allowed per connection.

Data type   Integer
Default   100

Maximum header field size

Specifies, in bytes, the maximum size for a header that can be included on an HTTP request.

Setting this property to a realistic size for your applications helps you to prevent denial of service (DoS) attacks that use large headers within an HTTP request as an attempt to make a system resource, such as the applications that handle HTTP requests, essentially unavailable to intended users.

The default for this property is 32768 bytes.

Maximum headers

Specifies the maximum number of headers that can be included in a single HTTP request.

Setting this property to a realistic number for your applications helps you to prevent denial of service (DoS) attacks that use a large number of headers within an HTTP request as an attempt to make a system resource, such as the applications that process HTTP requests, essentially unavailable to their intended users.

The default for this property is 50.

gotcha: Even if you do not change the value of this property, when you use this administrative console page to change other HTTP transport channel settings, the value specified for this property when you do your save is automatically saved to the corresponding property in the server.xml file. This change will override any value you previously set for this property in the server.xml file even if you did not intend to update the current value of this property in the server.xml file.

Limit request body buffer size

When selected, specifies that size of the body of an HTTP request is limited.

This property can be used to prevent denial of service attacks that use large HTTP requests as an attempt to make a system resource, such as the applications that process HTTP requests, essentially unavailable to their intended users.

Maximum request body buffer size

Specifies, in bytes, the maximum size limit for the body of an HTTP request. If this size is exceeded, the request is not processed.

A value can be added to this field only if the **Limit request body buffer size** property is selected.

Logging

You can use the settings in this section to configure and enable National Center for Supercomputing Applications (NCSA) access logging, or HTTP error logging. If you are running the product on z/OS, you can also use this section to configure and enable Fast Response Cache Accelerator (FRCA) logging. Enabling any of these logging services slows server performance.

If you want any of the enabled logging services to start when the server starts, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then in the Troubleshooting section, click **HTTP error, NCSA access and FRCA logging**, and select **Enable logging service at server start-up**. When this option is selected, any HTTP error, NCSA or FRCA logging service that is enabled automatically starts when the server starts.

gotcha: If you are running the product on z/OS, HTTP error, NCSA access, and FRCA logging settings must be specified on the controller. These settings are ignored if they are specified on the servant or adjunct.

NCSA access logging

By default, the **Use global logging service** option is selected for NCSA access logging. This setting means that the NCSA access logging settings default to the settings specified for NCSA access logging on the **HTTP error, NCSA access and FRCA logging** page in the administrative console. If you want to change these settings for this specific HTTP transport channel, expand the **NCSA Access logging** section, and select the **Use chain-specific logging** option.

After you select the **Use chain-specific logging** option, you can make the following configuration changes:
- Explicitly enable or disable NCSA access logging.
- Specify an access log file path that is different from the default path.

- Specify a maximum size for the access log file that is different from the default maximum size.
- Explicitly select the format of the NCSA access log file.

Enable access logging

When selected, a record of inbound client requests that the HTTP transport channel handles is kept in the NCSA access log file.

Access log file path

Specifies the directory path and name of the NCSA access log file. Standard variable substitutions, such as $(SERVER_LOG_ROOT), can be used when specifying the directory path.

If you are running the product on z/OS, you should use a server-specific variable, such as $(SERVER_LOG_ROOT), to prevent log file name collisions.

Access log maximum size

Specifies the maximum size, in megabytes, of the NCSA access log file. When this size is reached, the *logfile_name* archive log file is created. However, every time that the original log file overflows this archive file, the file is overwritten with the most current version of the original log file.

Maximum number of historical files

Specifies the maximum number of historical versions of the NCSA access log file that are kept for future reference.

NCSA access log format

Specifies in which format the client access information appears in the NCSA log file. If Common is selected, the log entries contain the requested resource and a few other pieces of information, but does not contain referral, user agent, and cookie information. If Combined is selected, referral, user agent, and cookie information is included.

FRCA logging

By default, the **Use global logging service** option is selected for FRCA logging. This setting means that the FRCA logging settings default to the settings that are specified for FRCA logging on the **HTTP error, NCSA access and FRCA logging** page in the administrative console. If you want to change these settings for this specific HTTP transport channel, expand the **FRCA logging** section, and select the **Use chain-specific logging** option.

This field only displays if you are running the product on z/OS.

After you select the **Use chain-specific logging** option, you can make the following configuration changes:
- Explicitly enable or disable FRCA logging.
- Specify an access log file path that is different from the default path.
- Specify a maximum size for the access log file that is different from the default maximum size.
- Explicitly select the format of the FRCA log file.

Enable FRCA logging

When selected, a record of inbound client requests that the HTTP transport channel handles is kept in the FRCA log file.

This field only displays if you are running the product on z/OS.

FRCA log file path

Specifies the directory path and name of the FRCA log file. you should use a server-specific variable, such as $(SERVER_LOG_ROOT), to prevent log file name collisions.

This field only displays if you are running the product on z/OS.

FRCA log maximum size

Specifies the maximum size, in megabytes, of the FRCA log file. When this size is reached, the *logfile_name* archive log file is created. However, every time that the original log file overflows this archive file, the file is overwritten with the most current version of the original log file.

This field only displays if you are running the product on z/OS.

Maximum number of historical files

Specifies the maximum number of historical versions of the FRCA log file that are kept for future reference.

This field only displays if you are running the product on z/OS.

FRCA log format

Specifies in which format the client access information appears in the FRCA log file. If Common is selected, the log entries contain the requested resource and a few other pieces of information, but does not contain referral, user agent, and cookie information. If Combined is selected, referral, user agent, and cookie information is included.

This field only displays if you are running the product on z/OS.

Error logging

By default, the **Use global logging service** option is selected for Error logging. This setting means that the Error logging settings default to the settings that are specified for Error logging on the **HTTP error, NCSA access and FRCA logging** page in the administrative console. If you want to change these settings for this specific HTTP transport channel, expand the **Error logging** section, and select the **Use chain-specific logging** option.

After you select the **Use chain-specific logging** option, you can make the following configuration changes:

- Explicitly enable or disable HTTP Error logging.
- Specify the access log file path. This path can be different from the default path.
- Specify a maximum size for the error log file. This value can be larger or smaller than the default maximum size.
- Specify the type of error messages that you want included in the HTTP error log file.

Enable error logging

When selected, HTTP errors that occur while the HTTP channel processes client requests are recorded in the HTTP error log file.

Error log file path

Indicates the directory path and the name of the HTTP error log file. Standard variable substitutions, such as $(SERVER_LOG_ROOT), can be used when specifying the directory path.

If you are running the product on z/OS, you should use a server-specific variable, such as $(SERVER_LOG_ROOT), to prevent log file name collisions.

Error log maximum size

Indicates the maximum size, in megabytes, of the HTTP error log file. When this size is reached, the *logfile_name* archive log file is created. However, every time that the original log file overflows this archive file, this file is overwritten with the most current version of the original log file.

Maximum number of historical files

Specifies the maximum number of historical versions of the HTTP error log file that are kept for future reference.

Error log level

Specifies the type of error messages that are included in the HTTP error log file.

You can select:

Critical
Only critical failures that stop the Application Server from functioning properly are logged.

Error The errors that occur in response to clients are logged. These errors require Application Server administrator intervention if they result from server configuration settings.

Warning
Information on general errors, such as socket exceptions that occur while handling client requests, are logged. These errors do not typically require Application Server administrator intervention.

Information
The status of the various tasks that are performed while handling client requests is logged.

Debug
More verbose task status information is logged. This level of logging is not intended to replace RAS logging for debugging problems, but does provide a steady status report on the progress of individual client requests. If this level of logging is selected, you must specify a large enough log file size in the **Error log maximum size** field to contain all of the information that is logged.

TCP transport channel settings

Use this page to view and configure a TCP transport channels. This type of transport channel handles inbound TCP/IP requests from a remote client.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Ports**. Click on **View associated transports** for the port associated with the TCP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the TCP transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels in a WebSphere Application Server environment. For example, an HTTP proxy inbound channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type	string

Port

Specifies the TCP/IP port this transport channel uses to establish connections between a client and an application server. The TCP transport channel binds to the hostnames and ports listed for the Port property. You can specify the wildcard * (an asterisk), for the hostname if you want this channel to listen to all hosts that are available on this system. However, before specifying the wildcard value, make sure this TCP transport channel does not have to bind to a specific hostname.

Data type	string

Maximum open connections

Specifies the maximum number of connections that are available for a server to use.

Leave the Maximum open connections property set to the default value 20000, which is the maximum number of connections allowed. The transport channel service by default manages high client connection counts and requires no tuning.

Default   20,000

Inactivity timeout

Specifies the amount of time, in seconds, that the TCP transport channel waits for a read or write request to complete on a socket.

If client connections are being closed without data being written back to the client, change the value specified for the Inactivity timeout parameter. This parameter controls the maximum number of connections available for a server's use. Upon receiving a new connection, the TCP transport channel waits for enough data to arrive to dispatch the connection to the protocol specific channels above the TCP transport channel. If not enough data is received during the time period specified for the Inactivity timeout parameter, the TCP transport channel closes the connection.

The default value for this parameter is 60 seconds, which is adequate for most applications. You should increase the value specified for this parameter if your workload involves a lot of connections and all of these connections can not be serviced in 60 seconds.

gotcha: The value specified for this property might be overridden by the wait times established for channels above this channel. For example, the wait time established for an HTTP transport channel overrides the value specified for this property for every operation except the initial read on a new socket.

Data type   Integer
Default   60 seconds

Address exclude list

Lists the IP addresses that are not allowed to make inbound connections.

Use a comma to separate the IPv4 or IPv6 or both addresses to which you want to deny access on inbound TCP connection requests.

All four numeric values in an IPv4 address must be represented by a number or the wildcard character * (an asterisk).

Following are examples of valid IPv4 addresses that can be included in an Address exclude list:

```
*.1.255.0
254.*.*.9
1.*.*.*
```

All eight numeric values of an IPv6 address must be represented by a number or the wildcard character * (an asterisk). No shortened version of the IPv6 address should be used. Even though a shortened version is processed with no error given, it does not function correctly in this list. Each numeric entry should be a 1- 4 digit hexadecimal number.

Following are examples of valid IPv6 addresses that can be included in an Address exclude list:

```
0:*:*:0:007F:0:0001:0001
F:FF:FFF:FFFF:1:01:001:0001
1234:*:4321:*:9F9f:*:*:0000
```

gotcha: The **Address include list** and **Host name include list** are processed before the **Address exclude list** and the **Host name exclude list**. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it will not be allowed access.

Address include list

Lists the IP addresses that are allowed to make inbound connections. Use a comma to separate the IPv4 or IPv6 or both addresses to which you want to grant access on inbound TCP connection requests.

All four numeric values in an IPv4 address must be represented by a number or the wildcard character * (an asterisk).

Following are examples of valid IP addresses that can be included in an Address include list:

```
*.1.255.0
254.*.*.9
1.*.*.*
```

All eight numeric values of an IPv6 address must be represented by a number or the wildcard character * (an asterisk). No shortened version of the IPv6 address should be used. Even though a shortened version is processed with no error given, it does not function correctly in this list. Each numeric entry should be a 1- 4 digit hexadecimal number.

Following are examples of valid IPv6 addresses that can be included in an **Address include list**:

```
0:*:*:0:007F:0:0001:0001
F:FF:FFF:FFFF:1:01:001:0001
1234:*:4321:*:9F9f:*:*:0000
```

gotcha: The Address include list and the Host name include list are processed before the Address exclude list and the Host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it will not be allowed access.

Host name exclude list

List the host names that are not allowed to make connections. Use a comma to separate the URL addresses to which you want to deny access on inbound TCP connection requests.

A URL address can start with the wildcard character * (an asterisk) followed by a period; for example, `*.Rest.Of.Address`. If a period does not follow the wildcard character, the asterisk will be treated as a normal non-wildcard character. The wildcard character cannot appear any where else in the address. For example, `ibm.*.com` is not a valid hostname.

Following are examples of valid URL addresses that can be included in a Host name exclude list:

```
*.ibm.com
www.ibm.com
*.com
```

gotcha: The Address include list and Host name include list are processed before the Address exclude list and the Host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.

- If an address is included in both an inclusion list and in an exclusion list, it is not allowed access.

Host name include list

Lists the host names that are allowed to make inbound connections. Use a comma to separate the URL addresses to which you want to grant access on inbound TCP connection requests.

A URL address can start with the wildcard character * (an asterisk) followed by a period; for example, `*.Rest.Of.Address`. If a period does not follow the wildcard character, the asterisk will be treated as a normal non-wildcard character. The wildcard character cannot appear any where else in the address. For example, `ibm.*.com` is not a valid hostname.

Following are examples of valid URL addresses that can be included in a hostname include list:

```
*.ibm.com
www.ibm.com
*.com
```

Note: The Address include list and Host name include list are processed before the Address exclude list and the Host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it is not allowed access.

DCS transport channel settings

Use this page to view and configure an DCS transport channels. This type of transport channel handles inbound Distribution and Consistency Services (DCS) messages.

By default, two channel transport chains are defined for an application server that contains a DCS channel:
- The chain named DCS contains a TCP and a DCS channel.
- The chain named DCS-Secure contains a TCP, an SSL, and a DCS channel.

Both of these chains terminate in, or use the same TCP channel instance. This TCP channel is associated with the DCS_UNICAST_ADDRESS port and is not used in any other transport chain. One instance of an SSL channel is reserved for use in the DCS-Secure chain. It also is not used in any other transport chains.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Ports**. Click **View associated transports** for the port associated with the DCS transport channel whose settings you want to look at.

Transport channel name

Specifies the name of the DCS transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels in the product environment. For example, a DCS transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type   String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

| **Data type** | Positive integer |
| **Default** | 0 |

ORB service transport channel settings

Use this page to view and configure an Object Request Broker (ORB) Service transport channels. This type of transport channel handles CORBA and RMI/IIOP inbound messages for the ORB Service. It enables clients to make requests and receive responses from servers in a network-distributed environment.

By default, two channel transport chains are defined for an application server that contains an ORB Service channel:
- The chain named ORB contains a TCP and an ORB Service channel.
- The chain named ORB-Secure contains a TCP, an SSL, and an ORB Service channel.

Both of these chains terminate in, or use the same TCP channel instance. This TCP channel is associated with the IIOP port and is not used in any other transport chain. One instance of an SSL channel is reserved for use in the DCS-Secure chain. It also is not used in any other transport chains.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Ports**. Click on **View associated transports** for the port associated with the ORB Service transport channel whose settings you want to look at.

Transport channel name

Specifies the name of the ORB service transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels in the product environment. For example, an ORB service transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

| **Data type** | string |

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

| **Data type** | Positive integer |
| **Default** | 0 |

SSL inbound channel

Use this page to determine which SSL inbound channel options to specify for the application server.

To view this administrative console page:

1. Click **Servers** > **Server Types** > **WebSphere application servers** > *server_name*.
2. Under Container settings, click **Web container settings** > **Web container transport chains** > i*secure_transport_chain*.
3. Under Transport channels, click **SSL Inbound Channel (SSL_1)**.

Transport Channel Name

Specifies the name of the SSL inbound channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels in an application server environment. For example, an SSL inbound channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type   String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type   Positive integer
Default   0

Centrally managed

Specifies that the selection of an SSL configuration is based upon the outbound topology view for the Java Naming and Directory Interface (JNDI) platform.

Centrally managed configurations support one location to maintain SSL configurations rather than spreading them across the configuration documents.

Default: Enabled

Specific to this endpoint

Specifies the SSL configuration alias that you want to use for outbound SSL communications.

This option overrides the centrally managed configuration for the JNDI (LDAP) protocol.

Session Initiation Protocol (SIP) inbound channel settings

Use this page to configure the SIP inbound channel settings.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Ports**. Click on **View associated transports** for the port associated with the UDP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the SIP inbound transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels in a WebSphere Application Server environment. For example, a SIP transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Default   UDP_(n) where (n) represents the number of instances of this channel in the system

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type   Positive integer
Default   10

Session Initiation Protocol (SIP) container inbound channel settings

Use this page to configure the SIP container inbound channel settings.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Ports**. Click on **View associated transports** for the port associated with the UDP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the SIP container inbound transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels in a WebSphere Application Server environment. For example, a SIP container transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Default   UDP_(n) where (n) represents the number of instances of this channel in the system

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the channel chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type   Positive integer
Default   10

Creating a new port

To create a new port and set up a channel chain to listen on a new port:

1. Go to the **Proxy Servers** > **SIP Proxy 1** > **Transport Chain** > **UDP_SIP_PROXY CHAIN** panel and select **UDP inbound channel (UDP 1)**.
2. On the following panel, select the **Port** (i.e., PROXY SIP ADDRESS (*:5060).
3. On the following panel, select **New**.

User Datagram Protocol (UDP) Inbound channel settings

Use this page to configure the UDP Inbound channel settings.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Ports**. Click on **View associated transports** for the port associated with the UDP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the UDP inbound transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels in a WebSphere Application Server environment. For example, a UDP transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Default	UDP_(n) where (n) represents the number of instances of this channel in the system

Address exclude list

Specifies the IP addresses that are not allowed to make inbound connections. Use a comma to separate the IPv4 and/or IPv6 addresses to which you want to deny access on inbound UDP connection requests.

The address include list and host name include list are processed before the address exclude list and the host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it is not allowed access.

Data type	String
Range	Valid IPv4 and IPv6 addresses with a wildcard character (*), an asterisk. All four elements of an IPv4 address must be represented by a number or a wildcard character. All eight numeric values of an IPv6 address must be represented by a number or the wildcard character (*).
Example	The following examples are valid IPv4 addresses that can be included in an Address exclude list:

```
*.1.255.0
254.*.*.9
1.*.*.*
```

All eight numeric values of an IPv6 address must be represented by a number or the wildcard character (*), an asterisk. No shortened version of the IPv6 address should be used. Even though a shortened version is processed with no error given, it does not function correctly in this list. Each numeric entry should be a 1- 4 digit hexadecimal number. The following examples are valid IPv6 addresses that can be included in an Address exclude list:

```
0:*:*:0:007F:0:0001:0001
F:FF:FFF:FFFF:1:01:001:0001
1234:*:4321:*:9F9f:*:*:0000
```

Address include list

Specifies the IP addresses that are allowed to make inbound connections. Use a comma to separate the IPv4 and/or IPv6 addresses to which you want to allow access on inbound UDP connection requests.

Data type	String

Range	Valid IPv4 and IPv6 addresses with a wildcard character (*), an asterisk. All four elements of an IPv4 address must be represented by a number or a wildcard character (*). All eight numeric values of an IPv6 address must be represented by a number or the wildcard character (*).

Web container inbound transport channel settings

Use this page to view and configure a web container inbound channel transport. This type of channel transport handles inbound web container requests from a remote client.

To view this administrative console page, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Web Container Settings** > **Web container** > **Web container transport chains** > *transport_chain* > **Web container inbound channel (***transport_channel_name***)** .

Transport Channel Name
Specifies the name of the web container inbound transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels in a WebSphere Application Server environment. For example, a web container inbound transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type	String

Discrimination weight
Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type	Positive integer
Default	0

Write buffer size
Specifies the amount of content in bytes to buffer unless the servlet explicitly calls flush/close on the response/writer output stream.

Data type	bytes
Default	32768 bytes

DataPower appliance manager transport channel settings

Use this page to view and configure a DataPower® appliance manager transport channel. This type of transport channel handles events from managed DataPower appliances.

To view this administrative console page, click **System administration > Deployment manager > Ports**. Find DataPowerMgr_inbound_secure, in the list of channels, and click **View associated transports**. In the list of associated transports, click DataPowerManagerInboundSecure, and then, under Transport Channels, click **DataPower appliance manager inbound channel (DPMGRDPMGR_1)**.

Transport channel name

Specifies the name of the transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % '

This name must be unique across all channels in a WebSphere Application Server environment. For example, an HTTP transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type	String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the tranport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type	Positive integer
Default	0

HTTP transport channel custom properties

If you are using an HTTP transport channel, you can add any of the following custom properties to the configuration settings for that channel.

gotcha: There are four Web container transport chains:

WCInboundAdmin

WCInboundAdminSecure

WCInboundDefault

WCInboundDefaultSecure

An application server or a proxy server inherits the custom property values that are specified for the WCInboundAdmin or WCInboundAdminSecure transport chain because one of these chains is usually the first chain to get activated when the application server is initialized. Therefore, before specifying any custom properties for a Web container transport chain, you should disable the WCInboundAdmin and WCInboundAdminSecure transport chains.

To add a custom property:

1. In the administrative console, click **Servers > Server Types**, and then select one of the following options, depending on the type of chain you are creating:
   - **Application servers > > *server_name*.** Under Web Container Settings, click **Web container transport chains >** *chain_name* **> HTTP Inbound Channel > Custom Properties > New**.
   - **WebSphere Proxy servers > *server_name*.** Under HTTP Proxy Server Settings, click **Proxy server transports**. Then, select either **HTTPS_PROXY_CHAIN** or **HTTP_PROXY_CHAIN**, and then click **> HTTP Inbound Channel > Custom Properties > New**.
2. Under **General Properties** specify the name of the custom property in the Name field and a value for this property in the Value field. You can also specify a description of this property in the Description field.
3. Click **Apply** or **OK**.
4. Click **Save** to save your configuration changes.
5. Restart the server.

Following are the descriptions of the HTTP transport channel custom properties provided with the product. These properties are not shown on the settings page for an HTTP transport channel. You can use the custom properties page to define the following properties:

- "ConnectionResponseTimeout"
- "CookiesConfigureNoCache"
- "EnableBuildBackupList"
- "HonorTransferEncoding" on page 419
- "limitFieldSize" on page 419
- "limitNumHeaders" on page 419
- "localLogFilenamePrefix" on page 419
- "RemoveServerHeader" on page 420
- "ServerHeaderValue" on page 420

ConnectionResponseTimeout

Use the ConnectionResponseTimeout property to set the maximum amount of time, in seconds, that the server waits for an application component to respond to an HTTP request. Set this variable for each of the HTTP transport channel definitions on the server. You must set this variable for both SSL transport channels and non-SSL transport channels. If the response is not received within the specified length of time, the servant might fail with ABEND EC3 and RSN=04130007. Setting this timer prevents client applications from waiting for a response from an application component that might be deadlocked, looping, or encountering other processing problems that cause the application component to hang.

Use the server custom properties protocol_http_timeout_output_recovery, and protocol_https_timeout_output_recovery, to indicate the recovery action that you want taken on timeouts for requests received over the HTTP and HTTPS transport channels.

Data type	Integer
Default	300 seconds

CookiesConfigureNoCache

Use the CookiesConfigureNoCache property to specify whether the presence of a Set-Cookie header in an HTTP response message triggers the addition of several cache related headers. If this property is set to true, an Expires header with a very old date, and a Cache-Control header that explicitly tells the client not to cache the Set-Cookie header are automatically added. These headers are not automatically added if this property is set to false.

This property is functionality equivalent to the com.ibm.websphere.cookies.no.header property that was available in previous versions of the product.

Data type	Boolean
Default	True

EnableBuildBackupList

Use the EnableBuildBackupList property to enable the HTTP channel to scan for the history files in the access and error logs directory, and rolling these files over with any newer log files created.

When this property is set to `true`, the HTTP Channel scans for the history files in the access and error logs directory, and rolls these files over with any newer log files created.

gotcha: After you configure the HTTP error log and the NCSA access log, make sure that the **Enable NCSA access logging** field is selected for the HTTP channels for which you want logging to occur. To verify that this field is selected for an HTTP channel, click **Servers > Application**

Servers > server > Web Container Transport Chains > HTTP Inbound Channel. This setting has to be enabled before setting this custom property to `true` has any effect on the HTTP channel functionality.

Data type	Boolean
Default	False

HonorTransferEncoding

Use the HonorTransferEncoding property to indicate whether the HTTP transport channels should convert a chunked message to a content-length delimited message when there is only one chunk.

When this property is set to `true`, the HTTP transport channels write out the chunks instead of switching to a content-length message even if the message only consists of one chunk. There is a performance impact to this setting because the HTTP transport channels does two writes for every single-chunk message: the first write is for the message, and the second write is for the zero byte chunk that marks the end of the message

When this property is set to `false`, the HTTP transport channels convert a chunked message to a content-length delimited message when there is only one chunk. This setting improves channel performance because the channel only does one write for a single-chunk message that is converted to a content-length message.

Data type	Boolean
Default	False

limitFieldSize

Use the limitFieldSize property to enforce the size limits on various HTTP fields, such as request URLs, or individual header names or values. Enforcing the size limits of these fields guards against possible Denial of Service attacks. An error is returned to the remote client if a field exceeds the allowed size.

Data type	Integer
Default	32768
Range	50-32768

limitNumHeaders

Use the limitNumHeaders property to limit the number of HTTP headers that can be present in an incoming message. If this limit is exceeded, an error is returned to the client.

Data type	Integer
Default	500
Range	50 to 4000

localLogFilenamePrefix

Use the localLogFilenamePrefix property to specify a prefix for the filename of the network log file. Normally, when inprocess optimization is enabled, requests through the inprocess path are logged based on the logging attributes set up for the web container's network channel chain. You can use this property to add a prefix to the filename of the network log file. This new filename is then used as the filename for the log file for inprocess requests. Requests sent through the inprocess path are logged to this file instead of to the network log file. For example, if the log file for a network transport chain is named `.../httpaccess.log`, and this property is set to `local` for the HTTP channel in that chain, the filename of the log file for inprocess requests to the host associated with that chain is `.../localhttpaccess.log`.

gotcha: If you specify a value for the localLogFilenamePrefix custom property, you must also set the accessLogFileName HTTP channel custom property to the fully qualified name of the log file you

want to use for in process requests. You cannot specify a variable, such as $(SERVER_LOG_ROOT), as the value for this custom property.

Data type   String

RemoveServerHeader

Use the RemoveServerHeader property to force the removal of any server header from HTTP responses that the application server sends, thereby hiding the identity of the server program.

Data type   Boolean
Default   False

ServerHeaderValue

Use the ServerHeaderValue property to specify a header that is added to all outgoing HTTP responses if a server header does not already exist.

Data type   String
Default   WebSphere Application Server v/*x.x*, where *x.x* is the
                                                   version of WebSphere Application Server that is running
                                                   on your system.

HTTP Tunnel transport channel custom properties

If you are using an HTTP Tunnel transport channel, you can add the following custom properties to the configuration settings for that channel.

To add a custom property:

1. In the administrative console, click **Servers > Server Types > Application servers >** *server_name* > **Ports**. Click on **View associated transports** for the HTTP Tunnel port to whose configuration settings you want to add this custom property.
2. Click **New**.
3. Under **General Properties** specify the name of the custom property in the Name field and a value for this property in the Value field. You can also specify a description of this property in the Description field.
4. Click **Apply** or **OK**.
5. Click **Save** to save your configuration changes.
6. Restart the server.

You can use the custom properties page to define the following HTTP tunnel transport channel custom properties:

- "pluginConfigurable"

Following is a description of the HTTP Tunnel transport channel custom property that is provided with the product. This property is not shown on the settings page for an HTTP Tunnel transport channel.

pluginConfigurable

Indicates whether or not the configuration settings for the HTTP Tunnel transport channel are included in the plugin-cfg.xml file for the web server associated with the application server that is using this channel.

Configuration settings for each of the web container transport channels defined for an application server are automatically included in the plugin-cfg.xml file for the web server associated with that application server.

Data type	Boolean
Default	False

TCP transport channel custom properties

If you are using a TCP transport channel, you can use TCP transport channel custom properties to configure internal TCP transport channel properties.

To add a TCP transport channel custom property, perform the following actions.

1. In the administrative console, click **Servers > Server Types**, and then follow one of the following paths:
   - **Application servers >** *server_name*, and then select one of the following options, depending on the type of chain you are creating:
     – Expand **SIP container settings**, and click **SIP container transport chains**.
     – Expand **Web container settings**, and click **Web container transport chains**.
     – Under **Container services**, and click **ORB Service > ORB Service Transport Chains**.
     – Expand **Server messaging**, and click either **Messaging engine inbound transports** or **WebSphere MQ link inbound transports**.
   - **Proxy servers**, and then expand **HTTP proxy server settings**, and click **Proxy server transports** and select either **HTTPS_PROXY_CHAIN** or **HTTP_PROXY_CHAIN**. Then click **HTTP proxy inbound channel**
2. Select the transport chain that includes the TCP channel for which you want to specify the custom property.
3. Select the **TCP inbound channel**.
4. Click **Custom properties > New**, expand **General properties**, and specify the name of the custom property in the **Name** field and a value for this property in the **Value** field. You can also specify a description of this property in the **Description** field.
5. Click **Apply** or **OK**.
6. Click **Save** to save your configuration changes.
7. Restart the server.

The following TCP transport channel custom property or properties is provided with the product. They are not shown on the settings page for a TCP transport channel.
- "listenBacklog"
- "zaioFreeInitialBuffers"

listenBacklog

Use the listenBacklog property to specify the maximum number of outstanding connection requests that the operating system can buffer while it waits for the application server to accept the connections. If a client attempts to connect when this operating system buffer is full, the connect request is rejected. The value of this property is specific to each transport.

If you need to control the number of concurrent connections, use the **Maximum open connections** field on the administrative console TCP transport channel settings page.

Data type	Integer
Default	511

zaioFreeInitialBuffers

Use the zaioFreeInitialBuffers property to indicate that the TCP channel should release the initial read buffers used on new connections as soon as these buffers are no longer needed for the connection. By default, this initial read buffer is cached for each connection. When a connection is closed, the read buffer

is reused to avoid a memory allocation. This process works well for non-persistent connections, where there is one request per connection. However, for highly persistent connections, the buffer might be held for a considerable amount of time even though it is not being used. For workloads that require a large number of connected clients, this situation can cause a shortage of Language Environment (LE) heap space. Unless your workload consists mainly of non-persistent connections, you should set this custom property to true to enable the release of the initial read buffers.

Note: If you set this property to true, you must also add the following argument to the JVM generic arguments that are configured for the application server that is using this TCP channel:

```
-Dcom.ibm.ws.buffermgmt.impl.WsByteBufferPoolManagerImpl=
    com.ibm.ws.buffermgmt.impl.ZOSWsByteBufferPoolManagerImpl
```

Data type	String
Default	false

Web container transport chain custom properties

Use this page to set custom properties for a web container transport channel.

To specify custom properties for a specific transport on the web container transport chain:

1. In the administrative console click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Web Container Settings** > **Web container transport chains**.
2. Select a transport chain.
3. Under **Transport Channels** select **Web container inbound channel (channel_name)**.
4. Under Additional Properties select **Custom properties**.
5. On the Custom properties page, click **New**.
6. On the settings page, enter the property that you want to configure in the Name field and the value that you want to set it to in the Value field.
7. Click **Apply** or **OK**.
8. Click **Save** on the console task bar to save your configuration changes.
9. Restart the server.

Following is a list of custom properties provided with the Application Server. These properties are not shown on the settings page for a web container transport. You can use the custom properties page to define the following custom properties:

- "disableRequestMessageChunking"
- "maxRequestMessageBodySize" on page 423
- "useStrictSSLConnectTimeout" on page 423

disableRequestMessageChunking

This custom property disables request message chunking when set to *true*. All of the request body up to protocol_http_large_data_inbound_buffer is buffered in memory.

For WCInboundAdmin and WCInboundAdminSecure transport chains, chunking is enabled by default to install large EAR files through the administrative console. For example, the settings for these chains are *disableRequestMessageChunking=false* . When chunking is enabled, the protocol_http_large_data_inbound_buffer value is ignored because the entire HTTP request is not buffered in the controller.

When chunking is disabled, the protocol_http_large_data_inbound_buffer value is used because the entire HTTP request is buffered in the controller.

Property name	disableRequestMessageChunking

Data type	string
Value	True or False
Defaults	By default, administrative chains have the disableRequestMessageChunking custom property explicitly set to true.

maxRequestMessageBodySize

When disableRequestMessageChunking is set to false, this is the maximum amount of request body that is buffered in memory before sending the next chunk to the servant. The maxRequestMessageBodySize custom property is valid only if the disableRequestMessageChunking custom property is set to false.

Property name	maxRequestMessageBodySize
Value	The default value is 32 kilobytes (KB). The minimum value is 32 and the maximum value is 8192, which is equivalent to 8MB.

useStrictSSLConnectTimeout

When this property is set to `true`, during a handshake with the client, the SSL Channel calculates the amount of time that can elapse before a the TCP timeout occurs based on the setting for the Socket timeout on the TCP channel. Therefore, when this property is set to `true`, the handshake can never take longer than the amount of time specified for the Socket timeout on the TCP Channel.

This property only applies to the SSL channel for a secure Web container transport chain, and is set by clicking the *name of the transport chain*> **SSL inbound channel > Custom properties**.

Property name	useStrictSSLConnectTimeout
Value	The default value is false.

Configuring inbound HTTP request chunking

Inbound HTTP request chunking is used to eliminate the restriction on messages greater than 10MB. The 10MB restriction is set because the entire message is buffered in the controller before the HTTP request is dispatched to the servant, therefore, the controller may fail with an out of memory condition when multiple large HTTP messages are processed simultaneously. With chunking enabled, the message is broken up into smaller pieces before it is processed by the web container and application. As a result, only one small chunk is buffered in memory at a time in the controller thus greatly reducing the amount of memory consumed by large HTTP messages. Applications do not require changes to enable inbound HTTP chunking.

About this task

Inbound HTTP request chunking, is configured at the web container transport chain level. You can configure each web container chain to enable or disable chunking. When chunking is enabled for a particular chain, you can also configure the maximum chunk size for chunking enabled for each chain.

All HTTP web container chains have chunking enabled by default.

Procedure

1. In the administrative console click **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Web Container Settings** > **Web container transport chains**.
2. Select a transport chain.
3. Under Transport Channels select **Web container inbound channel (channel_name)**.

4. Under Additional Properties select **Custom properties** to configure inbound HTTP request message chunking. See the article, "Web container transport chain custom properties" on page 422 for details about request message chunking settings.

   a. If the disableRequestMessageChunking property is already defined, select the **disableRequestMessageChunking** property from the list.

   b. If the disableRequestMessageChunking property is not defined, click **new**.

5. On the settings page, do one of the following:

   • To enable request message chunking, enter the property, **disableRequestMessageChunking** in the Name field and the enter the value, false, in the Value field. Click **Apply** or **OK** so save the custom property changes.

   • To disable request message chunking, enter the property, **disableRequestMessageChunking** in the Name field and the enter the value, **true**, in the Value field. Click **Apply** or **OK** so save the custom property changes.

6. Configure message chunk size if request message chunking is enabled. See the article, "Web container transport chain custom properties" on page 422 for details on these settings.

   a. On the Custom Properties page, click **New**.

   b. On the settings page, enter the property, **maxRequestMessageBodySize**, in the Name field and the enter a size, specified in kilobytes, between 32 and 8192 in the Value field.

   c. Click **Apply** or **OK**.

7. Click **Save** on the console task bar to save your configuration changes.

8. Restart the server.

Transport chain problems

Review the following topics if you encounter a transport chain problem.

TCP transport channel fails to bind to a specific host/port combination

If a TCP transport channel fails to bind to a specific port, one of the following situations might have occurred:

• You are trying to bind the channel to a port that is already bound to another application, such as another instance of an application server.

• You are trying to bind to a port that is in a transitional state waiting for closure. This socket must transition to closed before you restart the server. The port might be in TIME_WAIT, FIN_WAIT_2, or CLOSE_WAIT state. Issue the `netstat -a` command from a command prompt to display the state of the port to which you are trying to bind.

Deleting a transport chain

Transport chains cannot be deleted the same way that HTTP transports can be deleted. Because you cannot have multiple HTTP transports associated with the same port, when you delete an HTTP transport, you effectively delete the associated port and stop all traffic on that port. However, the process is more complicated for a transport chain because multiple transport chains might be associated with the same port and you do not want to disrupt traffic on transport chains that you are not deleting.

Before you begin

Determine whether you want to delete a particular transport chain or all of the transport chains that are associated with a specific port.

About this task

You might have to delete one or more transport chains if you have to delete a port.

To delete a transport chain:

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Ports**.
2. In the list of available ports, locate the port that you want to delete and click **View associated transports** for that port.
3. Select the transport chain you want to delete, and click **Delete**. If you intend to delete the port that is associated with this transport chain, repeat this step for all of the transport chains associated with this port.
4. Click **Save** to save your changes.

What to do next

If you delete all of the transport chains associated with a port, you can delete the port.

Disabling ports and their associated transport chains

Transport chains cannot be disabled the same way that HTTP transports can be disabled. Because you cannot have multiple HTTP transports associated with the same port, when you disable an HTTP transport, you effectively disable the associated port and stop all traffic on that port. However, the process is more complicated for a port that has associated transport chains because multiple transport chains might be associated with the same port, and you might not want to disrupt traffic on all of the transport chains at the same time.

Before you begin

Determine whether you want to disable a particular transport chain or all of the transport chains that are associated with a specific port.

About this task

You might need to disable a transport chain if you want to temporarily stop all incoming traffic on a particular port or on a particular transport chain that is associated with that port.

To disable a specific transport chain:

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name* **> Ports**.
2. In the list of available ports, locate the port that you want to delete and click **View associated transports** for that port.
3. Click the transport chain you want to disable.
4. Unselect the **Enabled** field, and click **OK**. If you want to temporarily stop all of the incoming traffic on a port, repeat this step for all of the transport chains associated with this port.
5. Click **Save** to save your changes.

What to do next

When you want traffic to resume on these disabled transport chains, repeat the preceding steps for all of the transport chains you disabled, and select the **Enabled** field.

Creating custom services

You can create one or more custom services for an application server. Each custom services defines a class that is loaded and initialized whenever the server starts and shuts down. Each of these classes must implement the com.ibm.websphere.runtime.CustomService interface. After you create a custom service, use the administrative console to configure that custom service for your application servers.

About this task

Custom services run in servants, not in controllers. For example, because there can be more than one servant started in the life of a server and these servants can be started long after the server (controller) is up, as needed by WLM, a custom service runs during the start of each servant.

If you need to define a hook point that runs when a server starts and shuts down, create a custom service class and then use the administrative console to configure a custom service instance. When the application server starts, the custom service starts and initializes.

Following is a list of restrictions that apply to the product custom services implementation. Most of these restrictions apply only to the initialize method:
- The initialize and shutdown methods must return control to the runtime.
- No work is dispatched into the server instance until all custom service initialize methods return.
- The initialize and shutdown methods are called only once on each service, and once for each operating system process that makes up the server instance.
- Initialization of process level static data, without leaving the process, is supported.
- Only JDBC RMLT (resource manager local transaction) operations are supported. Every unit of work (UOW) must be completed before the methods return.
- Creation of threads is not supported.
- Creation of sockets and I/O, other than file I/O, is not supported.
- Running standard Java Platform, Enterprise Edition (Java EE) code, such as client code, servlets, and enterprise beans, is not supported.
- The Java Transaction API (JTA) interface is not available.
- This feature is available in Java EE server processes and distributed generic server processes only.
- While the runtime makes an effort to call shutdown, there is no guarantee that shutdown will be called prior to process termination.
- JNDI operations that request resources are not supported.

Procedure

1. Develop a custom service class that implements the com.ibm.websphere.runtime.CustomService interface.

   The com.ibm.websphere.runtime.CustomService interface includes an initialize and shutdown methods. The application server uses the initialize method to pass properties to the custom service. These properties can include:
   - A property that provides the name of an external file that contains configuration information for the service. You can use the externalConfigURLKey property to retrieve this information.
   - Properties that contain name-value pairs that are stored for the service, along with the other system administration configuration data for the service.

Both the initialize and shutdown methods declare that they might create an exception, although no specific exception subclass is defined. If either method creates an exception, the runtime logs the exception, disables the custom service, and continues to start the server.

2. Configure the custom service.

   In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*, and then under Server Infrastructure, click **Custom Services > New**. Then, on the settings page for a custom service instance, create a custom service configuration for an existing application server or node agent, supplying the name of the class implemented. If your custom service class requires a configuration file, specify the fully-qualified path name to that configuration file in the **externalConfigURL** field. This file name is passed into your custom service class.

   To invoke a native library from the custom service, provide the path name in the **Classpath** field in addition to the path names that are used to locate the classes and JAR files for the custom service. This procedure adds the path name to the extension classloader, which allows the custom service to locate and correctly load the native library.

3. Stop the application server, and then restart it.

   Stop the application server, and then restart the server.

Results

Each custom services defines a class that is loaded and initialized whenever the server starts and shuts down.

The custom service loads and initializes whenever the server starts and shuts down.

Example

As previously mentioned, your custom services class must implement the com.ibm.websphere.runtime.CustomService interface. In addition, your class must implement the initialize and shutdown methods. The following example, shows the code that declares the class *ServerInit* that implements your custom service. This code assumes that your custom service class needs a configuration file. This example also includes the code that accesses the external configuration file. If your class does not require a configuration file, you do not have to include the configProperties portion of this code.

```
public class ServerInit implements com.ibm.websphere.runtime.CustomService
{
/**
* The initialize method is called by the application server runtime when the
* server starts. The Properties object that the application server passes
* to this method must contain all of  the configuration information that this
* service needs to initialize properly.
*
* @param configProperties java.util.Properties
*/
    static final java.lang.String externalConfigURLKey =
        "com.ibm.websphere.runtime.CustomService.externalConfigURLKey";

    static String ConfigFileName="";

    public void initialize(java.util.Properties configProperties) throws Exception
    {
        if (configProperties.getProperty(externalConfigURLKey) != null)
        {
            ConfigFileName = configProperties.getProperty(externalConfigURLKey);
        }

        // Implement rest of initialize method
    }
/**
* The shutdown method is called by the application server runtime when the
* server begins its shutdown processing.
```

```
*
   public void shutdown() throws Exception
   {
       // Implement shutdown method
   }
```

What to do next

Check the application server to verify that the initialize and shutdown methods of the custom service run the way that you want them to run.

Custom service collection

Use this page to view a list of services available to the application server and to see whether the services are enabled. A custom service provides the ability to plug into an application server and define code that runs when the server starts or shuts down.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, in the Server Infrastructure section, click **Administration > Custom services**.

External Configuration URL

Specifies the URL for a custom service configuration file.

If your custom services class requires a configuration file, the value provides a fully-qualified path name to that configuration file. This file name is passed into your custom service class.

Classname

Specifies the class name of the service implementation. This class must implement the Custom Service interface.

Display Name

Specifies the name of the service.

Enable service at server startup

Specifies whether the server attempts to start and initialize the service when its containing process (the server) starts. By default, the service is not enabled when its containing process starts.

Custom service settings

Use this page to configure a service that runs in an application server.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, in the Server Infrastructure section, click **Administration > Custom services >***custom_service_name*.

Enable service at server startup:

Specifies whether the server attempts to start and initialize the service when its containing process (the server) starts. By default, the service is not enabled when its containing process starts.

Data type	Boolean
Default	false

External Configuration URL:

Specifies the URL for a custom service configuration file.

If your custom services class requires a configuration file, specify the fully-qualified path name to that configuration file for the value. This file name is passed into your custom service class.

Data type	String
Units	URL

Classname:

Specifies the class name of the service implementation. This class must implement the Custom Service interface.

Data type	String
Units	Java class name

Display Name:

Specifies the name of the service.

Data type	String

Description:

Describes the custom service.

Data type	String

Classpath:

Specifies the class path used to locate the classes and JAR files for this service.

Data type	String
Units	Class path

Defining application server processes

To enhance the operation of an application server, you can define command-line information for starting or initializing an application server process. Such settings define runtime properties such as the program to run, arguments to run the program, and the working directory.

About this task

A process definition can include characteristics such as Java virtual machine (JVM) settings, standard in, error and output paths, and the user ID and password under which a server runs.

You can define application server processes using the administrative console or the wsadmin tool.

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere application servers**, and then click on an application server name.
2. In the Server Infrastructure section, click **Java and process management > Process definition**.
3. Select either **Control**, **Servant**, or **Adjunct**.

4. On the settings page for a process definition, specify the name of the executable to run, any arguments to pass when the process starts running, and the working directory in which the process will run. Then click **OK**.

5. Specify process execution statements for starting or initializing a UNIX or IBM i process.

6. Specify monitoring policies to track the performance of a process.

7. Specify process logs to which standard out and standard error streams write. Complete this step if you do not want to use the default file names.

8. Specify name-value pairs for properties needed by the process definition.

   gotcha: Each custom property name must be unique. If the same name is used for multiple properties, the process uses the value specified for the first property that has that name.

9. Stop the application server, and then have the executable, that the process definition specifies, restart the server. If the executable cannot restart the application server, the executable should use the generic server.

10. Check the server to verify that the process definition runs and operates as intended.

Process definition settings

Use this page to configure a process definition. A process definition includes the command line information necessary to start or initialize a process.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, in the Server Infrastructure section, click **Java and process management > Process definition**.

On z/OS you must then click **Control**, **Servant**, or **Adjunct**.

For z/OS, this page provides command-line information for starting, initializing, or stopping a process. Each of the commands for which information is provided can be used for the control process. Only the Start command and Start command arguments properties apply for the servant process. Specify the commands for the control process on one process definition panel and the commands for the servant process on another process definition panel. Do not specify the commands for the two different processes on the same panel.

Start command (startCommand)

This command line information specifies the platform-specific command to launch the server process.
z/OS control process

Table 46. Data type, Format, and Example. *The following table describes the data type, format, and example.*

Data type	String
Format	START *control_JCL_procedure_name*
Example	START BBO6ACR

z/OS servant process

For the z/OS servant process, the value on the start command specifies the procedure name that workload manager (WLM) uses to start the servant process. WLM only uses this value if the WLM dynamic application environment feature is installed.

Table 47. Data type, Format, and Example. *The following table describes the data type, format, and example.*

Data type	String
Format	*servant_JCL_procedure_name*
Example	BBO6ASR

Start command arguments (startCommandArgs)

This command line information specifies any additional arguments required by the start command.

z/OS control process

Table 48. Data type, Format, Example. The following table describes the data type, format, and example.

Data type	String
Format	JOBNAME=*server_short_name*, ENV=*cell_short_name.node_short_name.server_short_name*
Example	JOBNAME=BBOS001,ENV=SY1.SY1.BBOS001

z/OS servant process

Table 49. Data type, Format, Example. The following table describes the data type, format, and example.

Data type	String
Format	JOBNAME=*server_short_name*S, ENV=*cell_short_name.node_short_name.server_short_name*
Example	JOBNAME=BBOS001S,ENV=SY1.SY1.BBOS001

gotcha: For z/OS, the server short name (JOBNAME) contains 7 characters by default, but you can lengthen the short name to 8 characters.

Stop command (stopCommand)

This command line information specifies the platform-specific command to stop the server process

For z/OS, if this field is left blank, then the MVS STOP command is used to stop the generic server.

Table 50. Data type, Format, Example. Specify two commands in the field, one for the Stop command, and one for the Immediate Stop (CANCEL) command.

Data type	String
Format	STOP *server_short_name*;CANCEL *server_short_name*
z/OS example	STOP BBOS001;CANCEL BBOS001

Stop command arguments (stopCommandArgs)

This command line information specifies any additional arguments required by the stop command.

Table 51. Data type, Format, Example. Specify arguments for the Stop command and the Immediate Stop (CANCEL) command.

Data type	String
Format	*stop command arg string*;*immediate stop command arg string*
z/OS example	;ARMRESTART In this example, Stop has no arguments. Immediate Stop has the argument ARMRESTART. A semicolon precedes ARMRESTART.

Terminate command (terminateCommand)

This command line information specifies the platform-specific command to terminate the server process.

Table 52. Data type, Format, Example. Specify arguments for the terminate command.

Data type	String
Format	FORCE *server_short_name*
z/OS example	FORCE BBOS001

Terminate command arguments (terminateCommandArgs)

This command line information specifies any additional arguments required by the terminate command.

The default is an empty string.

Table 53. Data type, Format, Example. Specify additional arguments for the terminate command.

Data type	String
Format	*terminate command arg string*
z/OS example	ARMRESTART

Working directory

Specifies the file system directory that the process uses as its current working directory. This setting only applies for IBM i and distributed platforms. The process uses this directory to determine the locations of input and output files with relative path names.

This field does not display for the z/OS control process.

gotcha: On z/OS, the working directory is always the UNIX System Services directory that is defined using the OMVS setting of the RACF user profile for the user that starts the servant. Therefore, even if you specify a directory in this field, the UNIX System Services directory is used as the working directory. To provide compatibility between applications that run on a z/OS platform and on a distributed platform, set the UNIX System Services directory to the same value that you specify for the **Working directory** field on your distributed platform system.

Table 54. Data type. The following table describes the data type.

Data type	String

Executable target type

Specifies whether the executable target is a Java class or an executable JAR file.

Executable target

Specifies the name of the executable target. If the target type is a Java class name, this field contains the main() method. If the target type is an executable JAR file, this field contains the name of that JAR file.

Table 55. Data type. The following table describes the data type.

Data type	String

Process execution settings

Use this page to view or change the process execution settings for a server process.

A server process applies to either an application server, a node agent or a deployment manager.

If you are running on z/OS, to view this administrative console page for an application server, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, in the Server Infrastructure section, click **Java and process management**, click either **Servant**, **Control** or **Adjunct**, and then click **Process execution**.

To view this administrative console page for a node agent, click **System Administration > Node agents >***nodeagent_name*. Then, under Server Infrastructure, click **Java and process management > Process definition > Process execution**.

To view this administrative console page for a deployment manager, click **System Administration >** **Deployment manager**. Then, in the Server Infrastructure section, click **Java and process management**, click either **Servant**, **Control** or **Adjunct**, and then click **Process execution**.

Process Priority:

Specifies the operating system priority for the process. The administrative process that launches the server must have root operating system authority in order to honor this setting.

Data type	Integer
Default	20

UMASK:

Specifies the user mask under which the process runs (the file-mode permission mask).

The deployment manager and application servers must run with a 007 umask in order to support system management functions. Therefore, it is recommended that you do not change the default value of this setting for the deployment manager or the controller.

If the process is running in a servant, you can either specify a different user mask setting in this field or you can define a new environment variable for the servant that changes this setting. The new environment variable is _BPX_BATCH_UMASK. You define this new environment variable using the administrative console. To view the administrative console page, click **Environment > WebSphere variables**. To define the new variable, select the appropriate scope from the list of available options and then click **New** to create the name _BPX_BATCH_UMASK and set the desired value.

Note: After defining and setting _BPX_BATCH_UMASK, you will need to restart the server to pick up the new UMASK setting.

Data type	Integer
Default	007

Run As User:

Specifies the user that the process runs as. This user ID must be defined to the security system.

This field does not apply if you are running on a z/OS operating system. z/OS users must use RACF to associate a user to an address space. A process display shows the RACF associated user as the running user.

Note: For the Application Server to transition to the user that is specified in this option, the user that launching the process must be a root user. This is a restriction of the operating system.

Data type	String

Run As Group:

Specifies the group that the process is a member of and runs as.

This field does not apply if you are running on a z/OS operating system. z/OS users must use RACF to associate a group to an address space. A process display shows the RACF associated group as the running user.

Data type	String

Run In Process Group:

Specifies a specific process group for the process. A process group is a mechanism that the operating system uses to logically associate multiple processes and operate on them as a single unit. Usually, the operating system uses this mechanism for signal distribution.

This field does not apply if you are running on a z/OS operating system. z/OS users must use RACF to associate a process group to an address space. A process display shows the RACF associated process group as the running user.

Specific operating systems might allow other operations to be performed on a process group. Refer to your operating system documentation for more information on the operations that can be performed on a process group.

Data type	Integer
Default	0, which indicates that the process is not assigned to a specific process group.

Monitoring policy settings

Use this page to view or change settings that control how the node agent monitors and restarts a process.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, under Server Infrastructure, click **Java and process management > Monitoring policy**.

Maximum Startup Attempts:

Specifies the maximum number of times the product tries to start an application server in response to a start request. If the server cannot be started within the specified number of attempts, an error message is issued that indicates that the application server could not be started.

Data type	Integer
Default	3

Ping Interval:

Specifies, in seconds the frequency of communication attempts between the parent process, such as the node agent, and the process it has spawned, such as an application server. Adjust this value based on your requirements for restarting failed servers. Decreasing the value detects failures sooner; increasing the value reduces the frequency of pings, reducing system overhead.

Data type	Integer
Range	Set the value greater than or equal to 0 (zero) and less than 2147483. If you specify a value greater than 2147483, the application server acts as though you set the value to 0. When you specify a value of 0, no checking is performed.
Default	60

In a z/OS environment, the Ping Interval setting for a deployment manager or a node agent is ignored. However, the Ping Interval setting for an application server is used by the node agent to control the native z/OS operating system PidWaiter monitoring function. PidWaiter monitoring is similar in functionality to the

pinging function that is used in a distributed platform environment. Both of these monitoring functions determine whether an application server is still running. The only difference between the two monitoring functions is that PidWaiter monitoring does not send any of the TCP/IP messages that Ping Interval monitoring sends.

gotcha:

- If you set this property to 0. which indicates that no checking is performed, certain threads, such as PidWaiter, might terminate before the threads return their status to the initiating thread. In these situations, the deployment manager might not notifying the node agents of certain events, which could negatively impact stopServer.sh processing.
- If you set this property to a value greater than 0 but less than or equal to 5, the actual value used for the Ping Interval is 5.

You can also set the following two properties to considerably reduce the number of DNS lookups that might occur because of this monitoring activity:

1. You can add the JVM custom property com.ibm.websphere.management.monitoring.pingInterval for the controller for each process. The default value for this property is 60 seconds. It is not recommended that you change this default value unless you need to minimize the number of DNS lookups that occur. If you need to minimize the number of DNS lookups that occur, set this property to a time interval that is more appropriate for your system.

   When this property is set for the deployment manager, it regulates how frequently the deployment manager checks to see if the node agent is still running. When it is set for the node agent, it regulates how frequently the node agent checks to see if the deployment manager is still running. When it is set for an application server, it regulates how frequently the application server checks to see if the node agent is still running.

2. You can add the environment variable protocol_iiop_resolve_foreign_hostname at the cell level, and set to 0. Setting this variable to 0 disables the IIOP resolve foreign hostname function, thereby eliminating the DNS lookups this function performs.

Adding these two properties does not completely eliminate DNS lookups from within product processes.

Ping Timeout:

When a parent process is spawning a child process, such as when a process manager spawns a server, the parent process pings the child process to see whether the child was spawned successfully. This value specifies the number of seconds that the parent process should wait (after pinging the child process) before assuming that the child process failed.

Data type	Integer
Units	Seconds
Range	Set the value greater than or equal to 0 (zero) and less than 2147483647. If you specify a value greater than 2147483647, the application server acts as though you set the value to 0.
Default	300

Automatic Restart:

Specifies whether the process should restart automatically if it fails.

If you change the value specified for this field, you must restart the application server and the node agent before the new setting takes effect.

This setting does not affect what you specified for the Node Restart State setting. The two settings are mutually exclusive.

Data type	Boolean
Default	false for the z/OS environment

Node Restart State:

The setting only displays for the WebSphere Application Server, Network Deployment product. It specifies the desired behavior of the servers after the node completely shuts down and restarts.

If a server is already running when the node agent stops, that server is still running after the node agent restarts. If a server is stopped when the node agent restarts, whether the node agent starts the server depends on the setting for this property:

- If this property is set to STOPPED, node agent does not start the server.
- If this property is set to RUNNING, the node agent always starts the server.
- If this property is set to PREVIOUS, the node agent starts the server only if the server was running when the node agent stopped.

Note: Changes you make to the node restart state become effective after the node is synchronized. At the next NodeAgent restart (after the synch), the node restart state will be honored.

This setting does not affect what you specified for the Automatic Restart setting. The two settings are mutually exclusive.

Data type	String
Default	STOPPED
Range	Valid values are STOPPED, RUNNING, or PREVIOUS. If you want the process to return to its current state after the node restarts, use PREVIOUS.

Process definition type settings

Use this page to view or change settings for a process definition type. This page only displays if you are running the product on z/OS.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, under Server Infrastructure, click **Java and process management > Process definition**.

Control:

Specifies the process definitions for the control process

Servant:

Specifies the process definitions for the servant process.

Running multiple TCP/IP stacks

You might want to run multiple TCP/IP stacks on the same system to provide network isolation for one or more of your applications. For instance, you may have multiple Open System Adapter (OSA) features, each one connecting your system to a different network. You can assign a TCP/IP stack to each feature.

Before you begin

When configuring the product on a system with multiple stacks, you must first establish the product's stack affinity to the desired stack. Establishing stack affinity binds all socket communications to that stack, and allocates the proper host name resolution configuration data sets to the product. These data sets enable host name lookups to have the desired results.

Use the NETWORK DOMAINNAME parameter of SYS1.PARMLIB(BPXPRMxx) to specify the common INET physical file system, C_INET PFS, and then use this file system to set up multiple TCP/IP stacks. This physical file system allows you to configure multiple physical file systems (network sockets) and make them active concurrently.

If you plan to configure the product to use a non-default TCP/IP stack, consult *z/OS UNIX System Services Planning*, and *z/OS Communications Server: IP Configuration Reference*, for details.

About this task

gotcha: In the steps below, you will set a number variables. It is important to understand that these variables should be set at the node level.

To configure the product on a system with multiple stacks:

Procedure

1. Configure the data set for each application server's host name resolution. In the administrative console, click**Environment > WebSphere variables > New**.

   a. Add the RESOLVER_CONFIG UNIX process variable and specify the data set name in the **value** field.

   b. Export the RESOLVER_CONFIG variable in client shell scripts.

   • You can also use JCL to specify the name resolution configuration data set. To use JCL, add //SYSTCPD DD DSN=some.tcpip.DATA,DISP=SHR to the server JCL. The RESOLVER_CONFIG variable overrides the SYSTCPD DD statement.

   See *z/OS Communications Server: IP Configuration Reference*, for more information on the RESOLVER_CONFIG variable.

2. Establish the Application Server's stack affinity to the desired stack.

   a. In the administrative console, click **Environment > WebSphere variables** and set the _BPXK_SETIBMOPT_TRANSPORT UNIX process variable to the value of the desired transport. If this variable does not exist, click **New** and add it.

   b. Export the _BPXK_SETIBMOPT_TRANSPORT variable in client shell scripts.

   To set the **BPXK_SETIBMOPT_TRANSPORT** variable in the was.env file for the Daemon, you must prefix the variable with **DAEMON_**. This additional information causes the transformer that generates the was.env files to put add the variable to the was.env file for the Daemon. Because the **_BPXK_SETIBMOPT_TRANSPORT** variable already has a leading underscore, the final version of this variable, when set for the Daemon, contains two underscores preceding the variable name, as shown here **DAEMON__BPXK_SETIBMOPT_TRANSPORT**.

   gotcha: If you are setting this variable for the Daemon, you probably want to set it at the cell level to give all the Daemons in that cell the same setting. Unless one of the Daemons is serving multiple nodes, if for some reason you need to specify different settings for different Daemons in a cell, , you can set this variable at the node level.

   See *z/OS UNIX System Services Planning*, for more information on the _BPXK_SETIBMOPT_TRANSPORT variable.

Bind-specific support for TCP/IP resources on z/OS

Bind-specific support enables you to control the use of the product TCP/IP resources.

This support allows you to have the Application Server ORB and other products and applications on the same z/OS system without requiring the client code to configure unique ports. In other words, this support allows use of port 2809 by the Application Server and other products and applications on the same system. This support allows the utilization of multiple TCP/IP stacks (Common INET) by the product ORB and the use of multiple IP addresses on the same TCP/IP stack.

To use bind-specific support, use the **ORB_LISTENER_ADDRESS** end point, which specifies the IP address in dotted decimal format. The application servers listen for client connection requests on this IP address.

Restriction: DNS host names are not supported for the **ORB_LISTENER_ADDRESS** end point value.

Because a given IP address is associated with a given TCP/IP stack, you can specify the ORB_LISTENER_ADDRESS endpoint so that the applications servers use a specific TCP/IP stack.

In addition, because you can define multiple IP addresses for a given TCP/IP stack, the Application Server port 2809 servers could share the same TCP/IP stack with other products and applications requiring port 2809, because you made their IP addresses unique with the ORB_LISTENER_ADDRESS end point.

Alternatively, you can, without the use of bind-specific support, define alternate ports for port 2809 and the location service daemon, which are the only values defined by the CORBA standard. However it is not clear that all client ORBs will easily support configuring the Application Server port to something other than 2809. Configure the ports for the location service daemon and node by specifying port numbers on the z/OS location service daemon settings page in the administrative console.

For more information about multiple TCP/IP stacks (Common INET), see *z/OS UNIX System Services Planning*. For more information about multiple IP addresses on the same TCP/IP stack, see *z/OS Communications Server: IP Configuration Reference*.

Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root

> Refers to the top directory for a WebSphere Application Server node.
>
> The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.
>
> The default varies based on node type. Common defaults are `configuration_root`/`AppServer` and `configuration_root`/`DeploymentManager`.

configuration_root

> Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.
>
> The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.
>
> The default is `/wasv8config/`*cell_name*`/`*node_name*.

plug-ins_root
>Refers to the installation root directory for Web Server Plug-ins.

profile_root
>Refers to the home directory for a particular instantiated WebSphere Application Server profile.
>
>Corresponding product variables are *server.root* and *user.install.root.*
>
>In general, this is the same as *app_server_root*/profiles/*profile_name*. On z/OS, this will always be *app_server_root*/profiles/default because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root
>Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.
>
>The corresponding product variable is *smpe.install.root.*
>
>The default is /usr/lpp/zWebSphere/V8R0.

Configuring the JVM

As part of configuring an application server, you might define settings that enhance the way your operating system uses of the Java virtual machine (JVM).

About this task

The JVM is an interpretive computing engine that is responsible for running the byte codes in a compiled Java program. The JVM translates the Java byte codes into the native instructions of the host machine. The application server, being a Java process, requires a JVM to run and to support the Java applications running on it. JVM settings are part of an application server configuration.

To view and change the JVM configuration for an application server process, use the Java virtual machine page of the administrative console or use wsadmin scripts to change the configuration.

Procedure

1. In the administrative console, click **Servers** > **Server Types** > **WebSphere application servers** > *server_name*.
2. Under Server Infrastructure, click **Java and process management** > **Process definition**.
3. Select **Control**.
4. Select **Java virtual machine**.
5. Specify values for the JVM settings as needed, and click **OK.** For more information, see the documentation about Java virtual machine settings.

   Note: Java 5.0 SR10 and Java 6 SR5 correct issues in which the Java virtual machine (JVM) does not shut down correctly. If you have an application that depends on the previous behavior, which is not correct, you can revert to the previous behavior by adding the -XXallowvmshutdown:false argument to the Generic JVM arguments section.
6. Click **Save** in the messages section of the administrative console panel to save the changes to the master configuration.
7. Restart the application server.

Example

"Configuring application servers for UCS Transformation Format" on page 386 provides an example that involves specifying a value for the **Generic JVM Arguments** property on the Java virtual machine page to enable UTF-8 encoding on an application server. Enabling UTF-8 allows multiple language encoding support to be used in the administrative console.

"Configuring JVM sendRedirect calls to use context root" on page 448 provides an example that involves defining a property for the JVM.

Java virtual machine settings

Use this page to view, and change the Java virtual machine (JVM) configuration settings of a process for an application server.

To view this administrative console page, connect to the administrative console and navigate to the Java virtual machine panel.

For the z/OS platform follow one of the following paths.

Application server	Click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, in the Server Infrastructure section, click **Java and process management > Process definition > Control > Java virtual machine**
Deployment manager	Click **System Administration > Deployment manager**. Then, in the Server Infrastructure section, click **Java and process management > Process definition > Control > Java virtual machine**
Node agent	Click **System Administration > Node agent >** *node_agent*. Then, in the Server Infrastructure section, click **Java and process management > Process definition > Java virtual machine**

Classpath

Specifies the standard class path in which the Java virtual machine code looks for classes.

If you need to add a classpath to this field, enter each classpath entry into a separate table row. You do not have to add a colon or semicolon at the end of each entry.

The only classpaths that should be added to this field are the ones that specify the location of the following items:
- An inspection or monitoring tool to your system.
- JAR files for a product that runs on top of this product.
- JVM diagnostic patches or fixes.

Processing errors might occur if you add classpaths to this field that specify the location of the following items:
- JAR files for resource providers, such as DB2. The paths to these JAR files should be added to the relevant provider class paths.
- A user JAR file that is used by one or more of the applications that you are running on the product. The path to this type of JAR file should be specified within each application that requires that JAR file, or in server-associated shared libraries.
- An extension JAR file. If you need to add an extension JAR file to your system, you should use the ws.ext.dirs JVM custom property to specify the absolute path to this JAR file. You can also place the JAR file in the `WAS_HOME/lib/ext/` directory, but using the ws.ext.dirs JVM custom property is the recommended approach for specifying the path to an extension JAR file.

Data type   String

Boot classpath

Specifies bootstrap classes and resources for JVM code. This option is only available for JVM instructions that support bootstrap classes and resources.

If you need to add a classpath to this field, enter each classpath entry into a table row. You do not need to add the colon or semicolon at the end of each entry.

If you need to add multiple classpaths to this field, you can use either a colon (:) or semi-colon (;), depending on which operating system the node resides, to separate these classpaths.

The only classpaths that should be added to this field are the ones that specify the location of the following items:

- An inspection or monitoring tool to your system.
- JAR files for a product that runs on top of this product.
- JVM diagnostic patches or fixes.

Processing errors might occur if you add classpaths to this field that specify the location of the following items:

- JAR files for resource providers. such as DB2. The paths to these JAR files should be added to the relevant provider class paths.
- A user JAR file that is used by one or more of the applications that you are running on the product. The path to this type of JAR file should be specified within each application that requires that JAR file, or in server-associated shared libraries.
- An extension JAR file. If you need to add an extension JAR file to your system, you should use the ws.ext.dirs JVM custom property to specify the absolute path to this JAR file. You can also place the JAR file in the `WAS_HOME/lib/ext/` directory, but using the ws.ext.dirs JVM custom property is the recommended approach for specifying the path to an extension JAR file.

Verbose class loading

Specifies whether to use verbose debug output for class loading. The default is to not enable verbose class loading.

Data type   Boolean
Default   false

Verbose garbage collection

Specifies whether to use verbose debug output for garbage collection. The default is not to enable verbose garbage collection.

Data type   Boolean
Default   false

When this field is enabled, a report is written to the output stream each time the garbage collector runs. This report should give you an indication of how the Java garbage collection process is functioning.

You can check the verboseGC report to determine:
- How much time the JVM is spending performing garbage collection.

  Ideally, you want the JVM to spend less than 5 percent of its processing time doing garbage collection. To determine the percentage of time the JVM spends in garbage collection, divide the time it took to complete the collection by the length of time since the last AF and multiply the result by 100. For example:

  `83.29/3724.32 * 100 = 2.236 percent`

If you are spending more than 5 percent of your time in garbage collection and if garbage collection is occurring frequently, you might need to increase your Java heap size.

- If the allocated heap is growing with each garbage collection occurrence.

  To determine if the allocated heap is growing, look at the percentage of the heap that is remains unallocated after each garbage collection cycle, and verify that the percentage is not continuing to decline. If the percentage of free space continues to decline you are experiencing a gradual growth in the heap size from garbage collection to garbage collection. This situation might indicate that your application has a memory leak.

Note: Version 7.0 and previous versions use the optthruput garbage collection algorithm. In Version 8.0, the default is set to the generational garbage collector. This garbage collection algorithm can increase performance. The following JVM option is added to the WebSphere Application Server startup command: -Xgcpolicy:gencon. If you prefer to use the optthruput garbage collection alogorithm, you can remove -Xgcpolicy:gencon and the default optthruput garbage collection algorithm is used.

On the z/OS platform, you can also issue the MVS console command, `modify display`, `jvmheap`, to display JVM heap information. In addition, you can check the server activity and interval SMF records. The JVM heap size is also made available to PMI and can be monitored using the Tivoli Performance Viewer.

Verbose JNI

Specifies whether to use verbose debug output for native method invocation. The default is not to enable verbose Java Native Interface (JNI) activity.

Data type	Boolean
Default	false

Initial heap size

Specifies, in megabytes, the initial heap size available to the JVM code. If this field is left blank, the default value is used.

For z/OS, the default initial heap size for the controller is 48 MB, and the default initial heap size for the servant is 128 MB. These default values apply for both 32-bit and 64-bit configurations.

bprac: These default values are sufficient for most applications.

Increasing this setting can improve startup. The number of garbage collection occurrences are reduced and a 10 percent gain in performance is achieved.

Increasing the size of the Java heap continues to improves throughput until the heap becomes too large to reside in physical memory. If the heap size exceeds the available physical memory, and paging occurs, there is a noticeable decrease in performance.

Maximum heap size

Specifies, in megabytes, the maximum heap size that is available to the JVM code. If this field is left blank, the default value is used.

The default maximum heap size is 256 MB. This default value applies for both 32-bit and 64-bit configurations.

Increasing the maximum heap size setting can improve startup. When you increase the maximum heap size, you reduce the number of garbage collection occurrences with a 10 percent gain in performance.

Increasing this setting usually improves throughput until the heap becomes too large to reside in physical memory. If the heap size exceeds the available physical memory, and paging occurs, there is a noticeable decrease in performance. Therefore, it is important that the value you specify for this property allows the heap to be contained within physical memory.

To prevent paging, specify a value for this property that allows a minimum of 256 MB of physical memory for each processor and 512 MB of physical memory for each application server. If processor utilization is low because of paging, increase the available memory, if possible, instead of increasing the maximum heap size. Increasing the maximum heap size might decrease performance rather than improving performance.

bprac: These default values are appropriate for most applications. Enable the **Verbose garbage collection** property if you think garbage collection is occurring too frequently. If garbage collection is occurring too frequently, increase the maximum size of the JVM heap.

Debug mode

Specifies whether to run the JVM in debug mode. The default is to not enable debug mode support.

If you set the **Debug mode** property to `true`, then you must specify command-line debug arguments as values for the **Debug arguments** property.

Data type	Boolean
Default	false

Debug arguments

Specifies command-line debug arguments to pass to the JVM code that starts the application server process. You can specify arguments when the **Debug mode** property is set to `true`.

If you enable debugging on multiple application servers on the same node, verify that the same value is not specified for the address argument. The address argument defines the port that is used for debugging. If two servers, for which debugging is enabled, are configured to use the same debug port, the servers might fail to start properly. For example, both servers might still be configured with the debug argument `address=7777`, which is the default value for the debug address argument.

Data type	String
Units	Java command-line arguments

Generic JVM arguments

Specifies command-line arguments to pass to the Java virtual machine code that starts the application server process.

You can enter the following optional command-line arguments in the **Generic JVM arguments** field. If you enter more than one argument, enter a space between each argument.

gotcha: If the argument states that it is only for the IBM Developer Kit only, you cannot use that argument with the JVM from another provider, such as the Microsoft or Hewlett-Packard

- **-DhotRestartSync:**

  Specify `-DhotRestartSync` if you want to enable the hot restart sync feature of the synchronization service. This feature indicates to the synchronization service that the installation is running in an environment where configuration updates are not made when the deployment manager is not active. Therefore, the service does not have to perform a complete repository comparison when the deployment manager or node agent servers restart. Enabling this feature improves the efficiency of the first synchronization operation after the deployment manager or a node agent restarts, especially for installations that include mixed release cells, use several nodes, and run several applications.

- **-Xquickstart**

Specify -`Xquickstart` if you want the initial compilation to occur at a lower optimization level than in default mode. Later, depending on sampling results, you can recompile to the level of the initial compile in default mode.

bprac: Use -`Xquickstart` for applications where early moderate speed is more important than long run throughput. In some debug scenarios, test harnesses and short-running tools, you can improve startup time between 15-20 percent.

- **-Xverify:none**

Specify -`Xverify:none` if you want to skip the class verification stage during class loading . Using **-Xverify:none** disables Java class verification, which can provide a 10-15 percent improvement in startup time. However corrupted or invalid class data is not detected when this argument is specified. If corrupt class data is loaded, the JVM might behave in an unexpected manner, or the JVM might fail.

gotcha:
- Do not use this argument if you are making bytecode modifications, because the JVM might fail if any instrumentation error occurs.
- If you experience a JVM failure or the JVM behaves in an unexpected manner while this argument is in affect, remove this argument as your first step in debugging your JVM problem.

- **-Xnoclassgc**

Specify -`Xnoclassgc` if you want to disable class garbage collection. This argument results in more class reuse and slightly improved performance. However, the resources owned by these classes remain in use even when the classes are not being called.

gotcha: The performance impact of class garbage collection is typically minimal, and turning off class garbage collection in a Java Platform, Enterprise Edition (Java EE) based system, with its heavy use of application class loaders, might effectively create a memory leak of class data, and cause the JVM to throw an Out-of-Memory Exception.

You can use the `verbose:gc` configuration setting if you want to monitor garbage collection. You can use the resulting output to determine the performance impact of reclaiming these resources.

If you specify the -Xnoclassgc argument, whenever you redeploy an application, you should always restart the application server to clear the classes and static data from the pervious version of the application.

- **-Xgcthreads**

Specify -`Xgcthreads` if you want to use several garbage collection threads at one time. This garbage collection techniques is known as *parallel garbage collection*. This argument is valid only for the IBM Developer Kit.

When entering this value in the **Generic JVM arguments** field, also enter the number of processors that are running on your machine.

Specify -`Xgcthreads` as follows:

-Xgcthreads<*number of processors*>

gotcha: Do not add a space between --`Xgcthreads` and the *n* value for the number of processors.

-Xgcthreads5 is an example of specifying -`Xgcthreads` with 5 processors.

bprac: You should use parallel garbage collection if your machine has more than one processor.

- **-Xnocompactgc**

Specify -`Xnocompactgc` if you want to disable heap compaction. Heap compaction is the most expensive garbage collection operation. If you are using the IBM Developer Kit, you should avoid heap compaction. If you disable heap compaction, you eliminate all associated overhead.

- **-Xgcpolicy**

Specify -Xgcpolicy to set the garbage collection policy. This argument is valid only for the IBM Developer Kit.**-Xgcpolicy**

Set this argument to optthruput if you want to optimize throughput and it does not create a problem if long garbage collection pauses occur. This is the default parameter, recommended setting.

Set this argument to gencon, if you are using a generational garbage collector. The generational schema attempts to achieve high throughput along with reduced garbage collection pause times. To accomplish this goal, the heap is split into new and old segments. Long lived objects are promoted to the old space while short-lived objects are garbage collected quickly in the new space. The gencon policy provides significant benefits for many applications. However, it is not suited for all applications, and is typically more difficult to tune.

Set this argument to optavgpause, if you want concurrent marking used to track application threads starting from the stack before the heap becomes full. When this parameter is specified, the garbage collector pauses become uniform and long pauses are not apparent. However, using this policy reduces throughput because threads might have to do extra work.

Set this argument to subpool, if you want to increase performance on multiprocessor systems, that commonly use more then eight processors. This policy is only available on IBM System i®, System p®, and System z® processors. The subpool policy is similar to the optthruput policy except that the heap is divided into subpools that provide improved scalability for object allocation.

- **-XX**

The Java Platform, Standard Edition 6 (Java SE 6) has generation garbage collection, which allows separate memory pools to contain objects with different ages. The garbage collection cycle collects the objects independently from one another depending on age. With additional parameters, you can set the size of the memory pools individually. To achieve better performance, set the size of the pool containing objects that have short life cycles, such that the objects in the pool are not kept through more then one garbage collection cycle. Use the NewSize and MaxNewSize parameters to specify the size of the new generation pool.

Objects that survive the first garbage collection cycle are transferred to another pool. Use the SurvivorRatio parameter to specify the size of the survivor pool. SurvivorRatio. You can use the object statistics that the Tivoli Performance Viewer collects, or include the verbose:gc argument in your configuration setting to monitor garbage collection statistics. If garbage collection becomes a bottleneck, specify the following arguments to customize the generation pool settings to better fit your environment.

```
-XX:NewSize=lower_bound
-XX:MaxNewSize=upper_bound
 -XX:SurvivorRatio=new_ratio_size
```

The default values are:
- NewSize=2m
- MaxNewSize=32m
- SurvivorRatio=32

bprac: However, if you have a JVM with more than 1 GB heap size, you should use the following values:
- -XX:NewSize=640m
- -XX:MaxNewSize=640m
- -XX:SurvivorRatio=16

Alternatively, you could set 50% to 60% of the total heap size to a new generation pool.

- **-Xminf**

Specify -Xminf if you want to change the minimum free heap size percentage. The heap grows if the free space is below the specified amount. In reset enabled mode, this argument specifies the minimum percentage of free space for the middleware and transient heaps. The valued specified for this argument is a floating point number, 0 through 1. The default is .3 (30 percent).

- **-server | -client**

Java HotSpot Technology in Java SE 6 uses an adaptive JVM containing algorithms that, over time, optimize how the byte code performs. The JVM runs in two modes, **-server** and **-client**. In most cases, use **-server** mode, which produces more efficient run-time performance over extended lengths of time.

If you use the default **-client** mode, the server startup time is quicker and a smaller memory footprint is created. However, this mode lowers extended performance. Use the **-server** mode, which improves performance, unless server startup time is of higher importance than performance. You can monitor the process size, and the server startup time to check the performance difference between using the **-client** and **-server** modes.

- **-Dcom.ibm.CORBA.RequestTimeout=***timeout_interval*

  Specify the `-Dcom.ibm.CORBA.RequestTimeout=` *timeout_interval* argument to set the timeout period for responding to requests sent from the client. This argument uses the -D option. *timeout_interval* is the timeout period in seconds. If your network experiences extreme latency, specify a large value to prevent timeouts. If you specify a value that is too small, an application server that participates in workload management can time out before it receives a response.

  Specify this argument only if your application is experiencing problems with timeouts. There are no recommended values for this argument.

- **-Dcom.ibm.server.allow.sigkill=**

  The **-Dcom.ibm.server.allow.sigkill=true** argument allows the node agent process to use the terminate method of a process when the stop method does not complete within the time interval specified for the Ping interval. This setting is useful when the node agent is monitoring an application server and loses contact with that application server.

  When the monitoring policy for the application server allows the node agent to restart the application server because automatic restart is enabled for the application server, the node agent executes the stop method on the application server process. During stop processing, the node agent monitors the application server and if the application server does not stop within the time interval specified for the Ping interval, and this argument is set to `true`, which is the default value, the node agent executes the terminate method on the application server process to stop the application server process.

  If you set this argument to `false`, the node agent continues to monitor the stop process, but does not try to restart the application server.

  To use the administrative console to disable this argument, click **System Administration > Node agents >** *nodeagent_name* **> Java & Process Management > Process Definition > Java Virtual Machine > Generic JVM Arguments**.

- **-Dcom.ibm.websphere.wlm.unusable.interval=***interval*

  This argument only applies for z/OS. Specify the `-Dcom.ibm.websphere.wlm.unusable.interval=` *timeout_interval* argument to change the value of the com.ibm.websphere.wlm.unusable.interval property when the workload management state of the client is refreshing too soon or too late. This property specifies, in seconds, the amount of time that the workload management client run time waits after it marks a server as unavailable before it attempts to contact the server again. This argument uses the -D option. . The default value is 300 seconds. If the property is set to a large value, the server is marked as unavailable for a long period of time. This prevents the workload management refresh protocol from refreshing the workload management state of the client until after the time period has ended.

- **-Dcom.ibm.ws.buffermgmt.impl.WsByteBufferPoolManagerImpl=**

  This argument only applies for z/OS. Specify the `-Dcom.ibm.ws.buffermgmt.impl.WsByteBufferPoolManagerImpl=` argument to indicate that storage for individual direct byte buffers should be released as soon as the buffer is no longer needed. The only supported value for this argument is `com.ibm.ws.buffermgmt.impl.ZOSWsByteBufferPoolManagerImpl`.

  The direct byte buffers, that the JVM creates to handle request data, are allocated in the Language Environment (LE) heap instead of in the JVM heap. Typically, even if the direct byte buffers are no longer needed, the JVM does not release this native LE storage until the next garbage collection occurs. If the server is handling large requests, LE storage might become exhausted before the JVM runs a garbage collection cycle, causing the server to abnormally terminate (abend). Configuring the JVM with the following argument prevents these abends from occurring.

```
-Dcom.ibm.ws.buffermgmt.impl.WsByteBufferPoolManagerImpl=
    com.ibm.ws.buffermgmt.impl.ZOSWsByteBufferPoolManagerImpl
```

On the z/OS platform, you also need to specify this argument if you specify the `zaioFreeInitialBuffers` custom property for a TCP channel to have the channel release the initial read buffers used on new connections as soon as these buffers are no longer needed for the connection.

- **-DisSipComplianceEnabled=true|false**

  Specifies whether SIP compliance checking is enabled in the SIP proxy server. SIP compliance checking ensures that the SIP messages conform to the Session Initiation Protocol standard. When this property is set to `true`, SIP compliance checking is enabled.

  gotcha: If you are running a proxy server in a z/OS WebSphere Application Server, Network Deployment environment, and your proxy server is not part of a cluster, you can use the isSipComplianceEnabled SIP proxy server custom property to enable or disable SIP compliance checking for that SIP proxy server. However if you are running a stand-alone application server or your proxy server is part of a cluster, you must use this generic JVM argument to enable or disable SIP compliance checking.

- **-Xshareclasses:none**

  Specify the `-Xshareclasses:none` argument to disable the share classes option for a process. The share classes option, which is available with Java SE 6, lets you share classes in a cache. Sharing classes in a cache can improve startup time and reduce memory footprint. Processes, such as application servers, node agents, and deployment managers, can use the share classes option.

  If you use this option, you should clear the cache when the process is not in use. To clear the cache, either call the app_server_root/bin/clearClassCache.bat/sh utility or stop the process and then restart the process.

  gotcha:
    – J2EE application classes running in an application server process are not added to the shared class cache.

- -**XXallowvmshutdown:false**

  Use the **-XXallowvmshutdown:false** argument to revert to a previous behavior for the JVM that is not correct. Java 5.0 SR10 and Java 6 SR5 correct issues in which the Java Virtual Machine (JVM) does not shut down correctly. If you have an application that depends on the old behavior, you can revert to the previous behavior by adding the this argument to the Generic JVM arguments section.

Data type	String
Units	Java command-line arguments

Executable JAR file name
Specifies a full path name for an executable JAR file that the JVM code uses.

Data type	String
Units	Path name

Disable JIT
Specifies whether to disable the just-in-time (JIT) compiler option of the JVM code.

If you disable the JIT compiler, throughput decreases noticeably. Therefore, for performance reasons, keep JIT enabled.

Data type	Boolean
Default	false (JIT enabled)
Recommended	JIT enabled

Operating system name
Specifies JVM settings for a given operating system.

When the process starts, the process uses the JVM settings that are specified for the node as the JVM settings for the operating system.

Configuring JVM sendRedirect calls to use context root

If the com.ibm.websphere.sendredirect.compatibility property is not set and your application servlet code has statements such as *sendRedirect("/home.html")*, your web browser might display messages such as *Error 404: No target servlet configured for uri: /home.html*.

About this task

depfeat: The com.ibm.websphere.sendredirect.compatibility property is deprecated. You should modify your applications to redirect non-relative URLs (those starting with a "/") relative to the servlet container (*web_server_root*) instead of relative to the web application context root.

To instruct the server to use the context root for that the application uses for sendRedirect() calls instead of using the document root for the web server, configure the Java Virtual Machine (JVM) by setting the com.ibm.websphere.sendredirect.compatibility property to a `true` or `false` value.

Procedure

1. Access the settings page for a property of the JVM.
    a. In the administrative console, click **Servers > Server Types > Application servers**.
    b. On the Application server page, click on the name of the server whose JVM settings you want to configure.
    c. On the settings page for the selected application server, in the Server Infrastructure section, click **Java and process management > Process definition**.
    d. Select **Control**.
    e. On the Process definition page, click **Java virtual machine**.
    f. On the Java virtual machine page, click**Custom Properties**.
    g. On the Custom properties page, click **New**.
2. On the settings page for a property, specify `com.ibm.websphere.sendredirect.compatibility` in the **Name** field, and either `true` or `false` in the **Value** field. Then click **OK**.
3. Click **Save** on the console task bar.
4. Stop the application server, and then restart the application server.

Java virtual machine custom properties

You can use the administrative console to change the values of Java virtual machine (JVM) custom properties.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log`, `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

To set custom properties, connect to the administrative console and navigate to the appropriate Java virtual machine custom properties page.

Application server	Click **Servers > Server Types.** Select either **WebSphere application servers** > *server_name* or **WebSphere proxy servers** > *server_name*, and then, under Server Infrastructure, click **Java and process management > Process definition > Control > Java virtual machine > Custom properties**.
Deployment manager	Click **System Administration > Deployment manager > Java and process management > Process definition > Control > Java virtual machine > Custom properties**
Node agent	**System Administration > Node agent >** *nodeagent_ name* **> Java and process management > Process definition > Control > Java virtual machine > Custom properties**

If the custom property is not present in the list of already defined custom properties, create a new property, and enter the property name in the Name field and a valid value in the Value field. Restart the server to complete your changes.

You can use the Custom properties page to define the following properties for use by the Java virtual machine.

- "allowDeployerRoleGenPluginCfg" on page 452
- "com.ibm.config.eclipse.wtp.enablejemtrim" on page 453
- "com.ibm.config.eclipse.wtp.enablexmltrim" on page 453
- "com.ibm.config.eclipse.wtp.jem=finer" on page 453
- "com.ibm.config.eclipse.wtp.xmltrim=finer" on page 454
- "com.ibm.eclipse.wtp.allowRootedEntries" on page 454
- "com.ibm.ejs.ras.writeSystemStreamsDirectlyToFile" on page 454
- "com.ibm.ejs.sm.server.quiesceInactiveRequestTime" on page 454
- "com.ibm.ejs.sm.server.quiesceTimeout" on page 454
- "com.ibm.websphere.application.updateapponcluster.waitforappsave" on page 454
- "com.ibm.websphere.bean.delete.sleep.time" on page 455
- "com.ibm.websphere.deletejspclasses" on page 455
- "com.ibm.websphere.deletejspclasses.delete" on page 455
- "com.ibm.websphere.deletejspclasses.update" on page 455
- "com.ibm.websphere.ejb.UseEJB61FEPScanPolicy" on page 455
- "com.ibm.websphere.ejbcontainer.expandCMPCFJNDIName" on page 456
- "com.ibm.websphere.ejbcontainer.includeRootExceptionOnRollback" on page 456
- "com.ibm.websphere.jaxrpc.stub.typemapping.per.thread" on page 456
- "com.ibm.websphere.jaxrs.server.DisableIBMJAXRSEngine" on page 457
- "com.ibm.websphere.management.application.fullupdate" on page 457
- "com.ibm.websphere.management.application.fullupdate.*application_name*" on page 457
- "com.ibm.websphere.management.application.keepExistingSharedLibraries" on page 457
- "com.ibm.websphere.management.application.persistWebContext" on page 458
- "com.ibm.websphere.management.application.sync.recycleappasv5" on page 458
- "com.ibm.websphere.management.application.updateClusterTask.serverStopWaitTimeout" on page 458
- "com.ibm.websphere.management.application.updatesync.appExpansionTimeout" on page 458
- "com.ibm.websphere.management.configservice.validatePropNames" on page 459
- "com.ibm.websphere.management.jmx.random" on page 459

- "disableWSAddressCaching" on page 486
- "DRS_THREADPOOL_ISGROWABLE" on page 486
- "DRS_THREADPOOL_MINSIZE" on page 486
- "DRS_THREADPOOL_MAXSIZE" on page 486
- "dynacache.jms.cacheInstance" on page 486
- "dynacache.jms.connRetryInterval" on page 487
- "dynacache.jms.invProcessingDelay" on page 487
- "dynacache.jms.numStoredInvalidations" on page 487
- "invocationCacheSize" on page 487
- "java.util.logging.configureByLoggingPropertiesFile" on page 487
- "jaxws.payload.highFidelity" on page 488
- "jaxws.provider.interpretNullAsOneway" on page 488
- "jaxws.runtime.restrictStaticWebmethod" on page 489
- "jaxws.soapfault.local.exceptions.disable" on page 489
- "ODCClearMessageAge" on page 489
- "ODCInit.disabled" on page 489
- "org.eclipse.jst.j2ee.commonarchivecore.FILTERBINARIES" on page 490
- "sizeThreshold" on page 490
- "threadpool.maxsize" on page 490
- "was.xcfmonitor.enabled" on page 491
- "webservices.unify.faults" on page 491
- "wink.client.readTimeout" on page 491
- "wink.client.connectTimeout" on page 491

allowDeployerRoleGenPluginCfg

Set this custom property to true to enable users with the deployer role to generate and configure the `plugin-cfg.xml` file. After you set and save this custom property, restart the application server.

If the custom property is missing or its value is set to false, the following situations occur when the user has the deployer role permissions:

- The generation and configuration processes fail.
- An error message is issued.

To disable this function, delete the custom property or set its value to false.

Also, you can set this custom property from the command line using the wsadmin tool and the following Jacl script:

```
#------------------------------------------------------------------------
# setAllowDeployer.jacl - Jacl script for setting a the allowDeployerRoleGenPluginCfg
property
# of the web server plug-in for WebSphere Application Server
#------------------------------------------------------------------------
#    This Jacl file modifies the server.xml file for an application
server. This script is designed
#    to be invoked while the AppServer is running.
#
#    Here is an example of how to invoke the script:
#    wsadmin -f setAllowDeployer.jacl &ltnodeName&gt; &ltserverName&gt; &ltenableValue&gt;

#----------------------------------------------------------------------
proc printUsageAndExit {} {
```

```
    puts " "
    puts "Usage: wsadmin -f setAllowDeployer.jacl &ltnodeName> <serverName> <boolEnable>"
    puts "Note: enableValue argument is of type boolean; valid values
are true and false."
    exit
}
if { [llength $argv] >= 3 } {
    set nodename [lindex $argv 0]
    set servername [lindex $argv 1]
    set enablevalue [lindex $argv 2]
} else {
    printUsageAndExit
}

set cellname [$AdminControl getCell]
set propname "allowDeployerRoleGenPluginCfg"
set propdesc "Allow conditional deployer role for plug-in generation
and propagation"
set required "false"

set jvm [$AdminConfig getid
/Cell:${cellname}/Node:${nodename}/Server:${servername}/JavaProcessDef:/JavaVirtualMachine:/]

$AdminConfig modify $jvm [subst {{systemProperties {{{name {$propname}}
{value {$enablevalue}}
{description {$propdesc}} {required {$required}}}}}}]
$AdminConfig save

exit 0
```

Usage:

```
wsadmin -f setAllowDeployer.jacl node_name server_name true
```

com.ibm.config.eclipse.wtp.enablejemtrim

Use this custom property to enable the pruning of intermediate DOM nodes after the XML parse of the metadata occurs for an application.

gotcha: The setting for this property should match the setting for the com.ibm.config.eclipse.wtp.enablexmltrim custom property. Either both of these properties should be left unset, set to `false`, or set to `true`

The default value for this property is false.

com.ibm.config.eclipse.wtp.enablexmltrim

Use this custom property to enable the sharing of JavaClass instances, and the conversion of expanded JavaClass and JavaMethod objects to lightweight proxies after they are used.

gotcha: The setting for this property should match the setting for the com.ibm.config.eclipse.wtp.enablejemtrim custom property. Either both of these properties should be left unset, set to `false`, or set to `true`.

The default value for this property is false.

com.ibm.config.eclipse.wtp.jem=finer

Use this custom property to generate a trace from code areas that are enabled when the com.ibm.config.eclipse.wtp.enablejemtrim custom property is set to `true`.

gotcha: This property might impact performance. Therefore, this property should only be specified if a problem occurs during the pruning of intermediate DOM nodes after the XML parse of the metadata occurs for an application, and you need to obtain additional information to diagnose the cause of that problem.

com.ibm.config.eclipse.wtp.xmltrim=finer

Use this custom property to generate a trace from code areas that are enabled when the com.ibm.config.eclipse.wtp.enablexmltrim custom property is set to `true`.

gotcha: This property might impact performance. Therefore, this property should only be specified if a problem occurs with the sharing of JavaClass instances, or the conversion of expanded JavaClass and JavaMethod objects to lightweight proxies after they are used, and you need to obtain additional information to diagnose the cause of that problem.

com.ibm.eclipse.wtp.allowRootedEntries

In previous service releases, properties files in the root directory of an enterprise archive (EAR) file are not read properly. Thus, in this service release and later, the behavior is changed so that the class path in META-INF and MANIFEST.MF files are treated as a relative URI. To revert back to the original behavior, set the com.ibm.eclipse.wtp.allowRootedEntries to **true**.

com.ibm.ejs.ras.writeSystemStreamsDirectlyToFile

Use this custom property to support JSR-47 customized logging to write to the SystemOut stream without the format of WebSphere Application Server. The format of WebSphere Application Server includes information, for example, timestamp, thread ID, and some others. An application might not want this information to appear in the SystemOut stream (or perhaps prefer the information to appear in a different format). To disable the format of WebSphere Application Server, set this custom property to *true*.

com.ibm.ejs.sm.server.quiesceInactiveRequestTime

Use this custom property to specify, in milliseconds, how fast IIOP requests through the Object Request Broker (ORB) can be received and processed. For example, if you specify a value of 5000 for this property, the server does not attempt to shutdown until incoming requests are spaced at least 5 seconds apart. If the value specified for this property is too large, when the application server is stopped from the administrative console the following error message might be issued:

```
An error occurred while stopping Server1. Check the error logs for more information.
```

The default value is 5000 (5 seconds).

If you decide to use this custom property, you can specify it as a JVM custom property for either an application server, a node agent, or a deployment manger. It is typically set as an application server JVM custom property.

com.ibm.ejs.sm.server.quiesceTimeout

Use this custom property to specify, in seconds, the overall length of the quiesce timeout. If a request is still outstanding after this number of seconds, the server might start to shut down. For example, a value of 180 would be 3 minutes.

The default value is 180.

If you decide to use this custom property, you can specify it as a JVM custom property for either an application server, a node agent, or a deployment manger. It is typically set as an application server JVM custom property.

com.ibm.websphere.application.updateapponcluster.waitforappsave

Use this custom property to specify, in seconds, the amount of time that you want the deployment manager to wait for the extension tasks of the save operation to complete before starting the updated application.

gotcha: This property is only valid if it is specified for a deployment manager.

Usually during the save operation for an application update that is being performed using the rollout update process, the extension tasks of the save operation run as a background operation in a separate thread. If the main thread of the save operation completes before the synchronization portion of the rollout update process, the updated application fails to start properly.

When you add this custom property to your deployment manager settings, if the extension tasks of the save operation do not complete within the specified amount of time, the rollout update process stops the application update process, thereby preventing the application from becoming corrupted during the synchronization portion of the rollout update process.

The default value is 180.

com.ibm.websphere.bean.delete.sleep.time

Use this property to specify the time between sweeps to check for timed out beans. The value is entered in seconds. For example, a value of 120 would be 2 minutes. This property also controls the interval in the Servant process that checks for timed out beans still visible to the enterprise bean container.

The default value is 4200 (70 minutes). The minimum value is 60 (1 minute). The value can be changed through the administrative console. To apply this property, you must specify the value in both the Control and Servant JVM Custom Properties.

com.ibm.websphere.deletejspclasses

Use this property to indicate that you want to delete JavaServer Pages classes for all applications after those applications have been deleted or updated. The default value for this property is `false`.

gotcha: In a Network Deployment environment, this property only applies for the node agent.

com.ibm.websphere.deletejspclasses.delete

Use this property to indicate that you want to delete JavaServer Pages classes for all applications after those applications have been deleted, but not after they have been updated. The default value for this property is `false`.

gotcha: In a Network Deployment environment, this property only applies for the node agent.

com.ibm.websphere.deletejspclasses.update

Use this property to indicate that you want to delete JavaServer Pages classes for all applications after those applications have been updated, but not after they have been deleted. The default value for this property is `false`.

gotcha: In a Network Deployment environment, this property only applies for the node agent.

com.ibm.websphere.ejb.UseEJB61FEPScanPolicy

Use this property to control whether the product scans pre-Java EE 5 modules for additional metadata during the application installation process or during server startup. By default, these legacy EJB modules are not scanned.

The default value for this custom property is `false`.

You must set this property to `true` for each server and administrative server that requires a change in the default value.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.ejbcontainer.expandCMPCFJNDIName

The EJB container should allow for the expansion of the CMP Connection Factor JNDI Name when a user's JNDI name contains a user defined Application Server variable. The custom property, com.ibm.websphere.ejbcontainer.expandCMPCFJNDIName, makes it possible to expand the CMP Connection Fatory JNDI Name.

If the value is **true**, which is the default, the EJB Container expands a variable when found in the CMP Connection Factory JNDI Name. If the value is set to **false**, the EJB Container does not expand a variable.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.ejbcontainer.includeRootExceptionOnRollback

Use this Enterprise JavaBeans (EJB) custom property to enable the following functionality:

- Allow the root cause of a transaction roll back to be included in a TransactionRolledbackLocalException if the transaction is issued by a local caller.
- Allow the root cause of a transaction roll back from a from the commit method to be included in a TransactionRolledbackLocalException even if the transaction is issued by a remotecaller.
- Allow Heuristic Exceptions to be returned rather than a TransactionRolledbackLocalException, for a local client, or a TransactionRolledbackLocalException, for a remote client.
- Allow a RemoteExcpetion to be returned from a remote EJB method even if that method is running in the context of the transaction of the call. For example, consider that EJB1, method m1, begins a transaction and calls EJB2, method m2, where m2 causes an unhandled exception. In this case, the EJB Specification mandates that m1receives a TransactionRolledbackException. Setting this property to true allows a RemoteException, that includes any nested exceptions to be returned instead of the a TransactionRolledbackException even though this functionality is contrary to the EJB Specification requirement.

To enable this functionality set this property to `true`. To disable, this functionality set this property to `false`.

The default is `false`.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.jaxrpc.stub.typemapping.per.thread

Use this property to indicate whether the JAX-RPC runtime should use thread specific type mapping objects.

The JAX-RPC runtime uses a single TypeMappingRegistry object for all of the JAX-RPC clients. This design is intentional, and allows you to create a JAX-RPC stub and use it on multiple threads. However the singleton TypeMappingRegistry gets contaminated if multiple JAX-RPC Web services with different mappings are invoked concurrently. Even though this situation is uncommon, if it exists on your system, you can set the com.ibm.websphere.jaxrpc.stub.typemapping.per.thread custom property to `true` to indicate to the JAX-RPC runtime that it can use thread specific type mapping objects. These separate mapping objects avoid the contamination issue, and the various web service calls will succeed.

The default value for this property is false.

gotcha: You should not use this custom property unless you encounter a situation where the singleton TypeMappingRegistry gets contaminated. Enabling this property might regress applications that are dependent on access to the same JAX-RPC stub across multiple threads.

com.ibm.websphere.jaxrs.server.DisableIBMJAXRSEngine

Use this property to disable the JAX-RS integration run time from automatically processing your JAX-RS applications. The default value for this property is `false`.

Note: Setting this property to a value of `true` also disables the IBM JAX-RS runtime integration with EJB and JCDI.

com.ibm.websphere.management.application.fullupdate

Use this property to specify that when any of your applications are updated, you want the binaries directory erased and the content of the updated EAR file completely extracted.

If this property is not specified, each changed file within an updated EAR file is individually updated and synchronized in the node. This process can be time consuming for large applications if a large number of files change.

Setting the com.ibm.websphere.management.application.fullupdate property to:
- `true` specifies that, when any of your applications are updated, you want the binaries directory erased and the content of the updated EAR file completely extracted.
- `false` specifies that, when any of your applications are updated, you only want the changed files within that EAR file updated on the node and then synchronized.

gotcha: Use the com.ibm.websphere.management.application.fullupdate.*application_name* property if you only want to do a full replacement for a specific application instead of all of your applications.

com.ibm.websphere.management.application.fullupdate.*application_name*

Use this property to specify that when the specified application is updated, you want the binaries directory for that application erased and the content of the updated EAR file completely extracted.

If this property is not specified, each changed file within the updated EAR file for the specified application is individually updated and synchronized in the node. This process can be time consuming for large applications if a large number of files change.

Setting the com.ibm.websphere.management.application.fullupdate.*application_name* property to:
- `true` specifies that when the specified application is updated, you want the binaries directory erased and the content of the updated EAR file completely extracted.
- `false` that when the specified application is updated, you only want the changed files updated on the node and then synchronized.

gotcha: Use the com.ibm.websphere.management.application.fullupdate property if you want the binaries directory erased and the content of the updated EAR file completely extracted whenever any of your applications are updated.

com.ibm.websphere.management.application.keepExistingSharedLibraries

Use this property to specify how shared library mappings are handled during application updates.

When this property is set to `false`, then the shared libraries specified during the application update operation should replace the original shared library settings. False is the default setting.

When this property is set to `true`, after an application is updated, the application and module configurations include the original shared library settings in addition to those that are specified during the update operation.

com.ibm.websphere.management.application.persistWebContext

Use this property to specify whether the context root and virtual host information for web modules is persisted in the deployment.xml file. If this property is not specified, application deployment has to rely on annotation processing to read the context root and virtual host information, which impacts the performance of application deployment

When this property is set to `true`, the context root and virtual host information for web modules is persisted in the deployment.xml file, the peristed data is used for application deployment validation, which improves the performance of application deployment.

The default value is false, which means that the context root and virtual host information for web modules is not persisted in the deployment.xml file.

com.ibm.websphere.management.application.sync.recycleappasv5

Use this property to specify that you want your application recycling behavior to work the same way as this behavior worked in versions previous to Version 6.x of the product.

In Version 6.x and higher, after an application update or edit operation occurs, depending on which files are modified, either the application or its modules are automatically recycled. This recycling process occurs for all application configuration file changes, and all non-static file changes.

However, in versions previous to Version 6.x of the product, an application is recycled only if the Enterprise Archive (EAR) file itself is updated, or if the binaries URL attribute changes. An application is not recycled if there is a change to the application configuration file.

Setting the com.ibm.websphere.management.application.sync.recycleappasv5 property to:

- `true` specifies that you want your application recycling behavior to work the same way as this behavior worked in versions previous to Version 6.x of the product.
- `false` specifies that you want your application recycling behavior to work according to the Version 6.x and higher behavior schema.

The default value for this custom property is `false`.

gotcha: You must define this property in the node agent JVM. However, when defining this property, you can specify a scope of cell if you want the setting to apply to all of the nodes within a specific cell. If this property is set at both the cell and node agent level, the node agent setting takes precedence for that particular node agent.

com.ibm.websphere.management.application.updateClusterTask.serverStopWaitTimeout

Use this property to specify, in seconds how long the deployment manager waits for a server to stop completely in the $AdminTask updateAppOnCluster task. By default, the deployment manager waits for 60 seconds. The amount of time that you specify for this property should be greater than the longest amount of time that it takes to stop a server in the cluster.

This property can only be specified if you are using Version 7.0.0.1 or higher.

gotcha: This property is only valid if it is specified for a deployment manager.

com.ibm.websphere.management.application.updatesync.appExpansionTimeout

Use the property to specify how long the deployment manager waits to start an application server following an application update. This wait time enables the binaries for the application to be expanded to their directories after the update process completes. The amount of time that you specify for this property should be the maximum amount of time that any of the applications that reside in a node, take to fully expand their binaries.

By default, the rollout update function waits for 60 seconds, for each application expansion to occur following an update to one or more applications. Because the rollout function can be used to update multiple applications at the same time, the default value for this property is *n* x 60 seconds, where *n* is the number of applications that are being updated.

The default wait time might not be sufficient for larger applications. If, after your applications are updated, one or more of these applications do not start when the server starts, you might have to specify a longer length of time for the rollout update function to wait before starting the server.

gotcha: This property is only valid if it is specified for a deployment manager.

com.ibm.websphere.management.configservice.validatePropNames

Use this property to specify whether to enforce character restrictions for custom property names, and for the name value of Property and J2EEResourceProperty configuration objects in wsadmin commands.

You can use one of the following methods to turn off character validation for custom property names, and the name value of Property and J2EEResourceProperty configuration objects in wsadmin commands.

- Set the com.ibm.websphere.management.configservice.validatePropNames Java system property to `false` in the Java virtual machine (JVM) for the deployment manager server.
- Set the com.ibm.websphere.management.configservice.validatePropNames property using the **-javaoption** parameter when you use the wsadmin tooling in the local mode.

```
wsadmin -conntype none -javaoption
"-Dcom.ibm.websphere.management.configservice.validatePropNames=false"
```

com.ibm.websphere.management.jmx.random

Use this property to enable the controller to randomly select an initial servant from the servant pool to process a Java Management Extensions (JMX) connector requests instead of automatically assigning the request to the hot servant.

By default, when multiple servants are enabled and the application server receives a JMX connection request, the application server assigns the request to the first servant, which is also referred to as the hot servant. This strategy minimizes the risk that the request is assigned to a servant that is paged out. However, if the first servant has a heavy workload, requests to that servant eventually fail. Therefore, the advantage of using the random algorithm is that the assigned servant is probably not already handling a lot of other requests. The disadvantage of using the random algorithm is that the selected servant might be paged out and have to be paged back in before it can handle the request.

Setting the com.ibm.websphere.management.jmx.random property to:

- `true` specifies that the controller will randomly select an initial servant from the servant pool to process a JMX connector requests.
- `false` specifies that the controller will assign all JMX connector requests to the hot servant.

If you decide to use this custom property, you must specify it as a JVM custom property for the controller.

com.ibm.websphere.management.nodesync.skipWebServerTarget

Use this property to control whether application binaries are distributed to target nodes in which the application only targets web servers.

By default, the custom property value is set to `false`, which causes the application server to distribute application binaries to target nodes. If you set this custom property to `true`, the application server does not distribute application binaries to target nodes that contain web server targets only.

Important: System behavior is not affected by this custom property value. Also, it is not affected by the node synchronization change that led to a change in this custom property value for target

nodes that contain application server targets. These statements are true regardless of whether web server targets are included in that same node.

The following sample output shows the isAppReady output, which is generated for web server-only target nodes when you set this custom property to `true`:

```
wsadmin>$AdminApp isAppReady SuperApp
ADMA5071I: Distribution status check started for application SuperApp.
WebSphere:cell=IBMCell10,node=IBMNode30,distribution=true,expansion=skipped
ADMA5011I: The cleanup of the temp directory for application SuperApp is complete.
ADMA5072I: Distribution status check completed for application SuperApp.
true
```

You can set this custom property for the deployment manager, but not the application server or node agent. You can use the administrative console or complete the following steps using a command-line:

1. Start the wsadmin scripting tooling from the *profile_root*/bin directory. You need to start the .wsadmin.bat or wsadmin.sh command within that directory.

2. Run one of the following commands:

   Using Jacl

   ```
   $AdminTask setJVMSystemProperties {-serverName server_name -nodeName node_name
   -propertyName com.ibm.websphere.management.nodesync.skipWebServerTarget
   -propertyValue true}
   ```

   Using Jython string

   ```
   AdminTask.setJVMSystemProperties('[-serverName server_name -nodeName node_name
   -propertyName com.ibm.websphere.management.nodesync.skipWebServerTarget
   -propertyValue true]')
   ```

   Using Jython list

   ```
   AdminTask.setJVMSystemProperties (['-serverName', 'server_name', '-nodeName',
   'node_name' '-propertyName',
   'com.ibm.websphere.management.nodesync.skipWebServerTarget', '-propertyValue',
   'true'])
   ```

   where the *server_name* and *node_name* variables are specific to your environment.

3. Restart the deployment manager.

com.ibm.websphere.management.processEmbeddedConfigGlobal

Use this property to globally enable or disable processing of the embedded configuration of enhanced application Enterprise Archive (EAR) files during deployment. An enhanced EAR file results when you export an installed application.

This custom property overrides globally the default setting for the **Process embedded configuration** (-processEmbededConfig) option. By default, **Process embedded configuration** is set to `true` (selected) for enhanced EAR files and `false` (deselected) for all other EAR files. The **Process embedded configuration** setting determines the directory to which the product expands an enhanced EAR file during deployment of the enhanced EAR file. If you exported an application from a cell other than the current cell and did not specify the $(CELL) variable for **Directory to install application** when first installing the application, setting **Process embedded configuration** to `false` during deployment of an enhanced EAR file extracts the enhanced EAR file in the *app_server_root*/profiles/installedApps/*current_cell_name* directory. Otherwise, if **Process embedded configuration** is set to `true`, the enhanced EAR file is expanded in the *app_server_root*/profiles/installedApps/*original_cell_name* directory, where *original_cell_name* is the cell on which the application was first installed. If you specified the $(CELL) variable for **Directory to install application** when you first installed the application, installation expands the enhanced EAR file in the *app_server_root*/profiles/installedApps/*current_cell_name* directory.

When this processEmbeddedConfigGlobal custom property is set to `false`, the product does not process the embedded configuration of any application, including enhanced EAR files, during deployment. After you

set processEmbeddedConfigGlobal to `false`, the product does not process the embedded configuration of enhanced EAR files. However, when deploying an individual enhanced EAR file, you can override this false setting by explicitly setting **Process embedded configuration** to `true`.

When this processEmbeddedConfigGlobal custom property is set to `true`, the product processes the embedded configuration of enhanced EAR files.

Regardless of whether this processEmbeddedConfigGlobal custom property is set to `true` or `false`, the product deploys applications that do not have embedded configurations as usual. The setting has no effect on deployment.

gotcha: To apply this property, you must specify the property and its value in both the Control and Servant JVM custom properties.

com.ibm.websphere.management.registerServerIORWithLSD
Use this property to control whether a federated server registers with the Location Service Daemon (LSD). Normally, a federated server requires the node agent to be running. To direct the server to not register with the LSD and remove its dependency on an active node agent, the com.ibm.websphere.management.registerServerIORWithLSD JVM custom property must be set to `false`, and the ORB_LISTENER_ADDRESSConfiguring inbound transports must be set to a value greater than 0 so that the ORB listens at a fixed port. The setting for this property is ignored if the ORB_LISTENER_ADDRESS property is set to 0 (zero) or is not specified, and the federated server registers with the LSD.

Set this property to `false` if you want the server to run even when the node agent is not running. When this property is set to `true`, the federated server registers with the LSD. The default value for this custom property is `true`.

After you change the value for this custom property, you must do a full synchronization before this change is reflected in the server.xml file on the node. If you cannot start the node agent for the server, you must manually edit the server.xml file to change the value of this custom property from `true` to `false`.

When you set the com.ibm.websphere.management.registerServerIORWithLSD property to false, the server does not notify the node agent when it dynamically assigns the ORB_LISTENER_ADDRESS port. There also will not be any indirect Interoperable Object References (IORs) that the node agent can resolve to a server. All of the IORs become direct, which means that the node agent can only contact that server if a static ORB_LISTENER_ADDRESS has been assigned to that server.

gotcha: If you set the com.ibm.websphere.management.registerServerIORWithLSD property to false and do not intend to use the high availability manager (HAManager) service and node agent process, then you should create a static routing table to enable static routing. Enabling static routing ensures that workload management (WLM) continues to function properly. You use com.ibm.websphere.management.registerServerIORWithLSD to prevent indirect IORs from being returned for any services being provided by the application servers (and having the node agent then map these indirect IORs to direct IORs).

The use of the node agent establishes a known port on the server side to which the client can send requests when the ports for the application servers is not known. Creating a static routing table and enabling static routing provides static ports for the application servers and make these ports available for use. With static ports, node agent processing is not necessary.

Note: With dynamic application server ports, you cannot use the static routing table. Node agent processing is required with dynamic application server ports, because the client needs to obtain the new port information when the servers are restarted.

Setting the com.ibm.websphere.management.registerServerIORWithLSD property does not affect the use of either dynamic or static ports or the routing of requests by WLM. However, you do lose a potential level of failover provided by the node agents of the HAManager.

WLM processing is retained with weighted spraying of requests.

com.ibm.websphere.metadata.ignoreDuplicateRefBindingsInWebModul

Use this property to control whether the JVM ignores instances of duplicate reference bindings in the DTD file for a web module in a Java 2 Platform, Enterprise Edition (J2EE) version 1.3 application. Typically a MetaDataException occurs if the DTD file for a web module in a Java 2 Platform, Enterprise Edition (J2EE) version 1.3 application contains duplicate references.

The standards for the DTD file for a web module specifically states that the reference bindings must have a unique name fields. Therefore, an application that contains a web module that includes duplicate reference bindings is technically a non-compliant application.

Although the standards for the DTD file for a web module forbids a user from defining duplicate reference bindings, the JVMs in versions of the product that preceded 7.0 tolerate duplicate reference bindings. If you have DTD files for web modules in Java 2 Platform, Enterprise Edition (J2EE) version 1.3 applications that contain duplicate reference bindings, you can either remove the duplicate reference, or add this property to your JVM configuration settings, and set the property to `true`.

com.ibm.websphere.network.useMultiHome

In a multihomed environment where the product is restricted to listen only on a specific IP address for Discovery and SOAP messages, set this property to false for the deployment manager, all application servers and all node agents. By default, the value of the property is true and the application server listens on all IP addresses on the host for Discovery and SOAP messages. If the property is set to false, the product only listens for Discovery and SOAP messages on the configured host name. If you set the property to false, you must also:

- Have a host name configured on the product that resolves to a specific IP address.
- Ensure that the `end point` property for the deployment manager, all application servers, and all node agents is set to this host name. For the deployment manager, the end points that must be set are CELL_DISCOVERY_ADDRESS and SOAP_CONNECTOR_ADDRESS. For the node agent and application servers, only the SOAP_CONNECTOR_ADDRESS end point must be set.

You can change the value through the administrative console. Modify the defaults by setting the value for the server, deployment manager, and node agent. In order for these changes to take place, you must restart the server.

If you cannot contact the server, check the setting for com.ibm.websphere.network.useMultihome to ensure it is correct. You can change the value through the administrative console. Modify the defaults by setting the value for the server, deployment manager, and node agent. You must restart the server before these changes take effect.

com.ibm.websphere.sib.webservices.useTypeSoapArray

You can pass messages directly to a bus destination by overriding the JAX-RPC client binding namespace and endpoint address. However:

- The default RPC-encoded web services string array message that is generated might not interoperate successfully with some target service providers.
- The string array message produced is not exactly the same as the standard JAX-RPC equivalent, which can interoperate successfully.

Here are examples of the two different messages:
- Service integration bus message:

```
<partname env:encodingStyle='http://schemas.xmlsoap.org/soap/encoding/ xsi:type='ns1:ArrayOf_xsd_string'>
  <item xsi:type='xsd:anySimpleType'>namevalue</item>
</partname>
```

- JAX-RPC client message:

```
<partname xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]">
  <item>namevalue</item>
</partname>
```

Set this property to `true` to modify the default behavior and send a string array message that is fully compatible with standard JAX-RPC. Setting this property modifies the default behavior for all outbound JMS web services invocations sent from the service integration bus.

com.ibm.websphere.webservices.attachment.tempfile.expiration

Use this property to indicate, in seconds, an expiration time for an attachment on a JAX-WS or Service Component Architecture (SCA) client or service. If an attachment is not accessed for a period of time greater than the expiration time, the web service runtime is allowed to delete the attachment.

The JAX-RPC programming model allows access to attachments from incoming Web service messages. The attachment might be accessed immediately, or might be stored for later processing. Therefore, the memory associated with the attachment might persist much longer than the lifetime of the Web service interaction. because there is no precise length of time after which the Web service runtime can safely free the attachment.

For small attachments, the memory is eventually freed by the Java garbage collector.

For large attachments, the JAX-RPC runtime stores the attachment data in a temporary file, thereby allowing the runtime to process extremely large attachments without consuming memory. If the application does not access the attachment, or if the application does not adequately close the data handler associated with the attachment, the large temporary file is not freed. Over time, these temporary attachment files might accumulate on the file system if no expiration time is specified for these files.

bprac: A setting of 600 is recommended if you need to specify an expiration time for these attachments. The default setting for this custom property is 0 seconds, which indicates that there is no expiration time for these attachments.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.attachements.maxMemCacheSize

Use this property to specify, in kilobytes, the maximum size of an attachment on the JAX-RPC client or service that can be written to memory. For example, if your web service needs to send 20 MB attachments, set the property to 20480.

When determining a value for this property, remember that the larger the maximum cache size, the more impact there is on performance, and, potentially, to the Java heap.

If you do not specify a value for this property, the maximum memory that is used to cache attachments is 32 KB, which is the default value for this property.

Note: To specify the maximum size of an attachment on the JAX-WS client or service, see the com.ibm.ws.websvcs.attachments.sizethreshold custom property.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.DisableIBMJAXWSEngine

Use this property to turn off web services annotation scanning at the server level. By default, web services annotation scanning is enabled at the server level.

To turn off annotation scanning at the application level, set the DisableIBMJAXWSEngine property in the META-INF/MANIFEST.MF of a WAR file or EJB module to true.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.http.OneWayConnectionRecycleTime

Use this property to specify, in seconds, how long the web services engine should wait before reusing a one-way connection. When a one-way connection is reused too quickly, a web service operation might fail on the client because of a timeout problem, such as a SocketTimeoutException.

When a value is specified for this property, one-way connections are not reset until the specified number of seconds elapses, starting from when the request is sent.

By default, this property is not set and one-way connections are reset immediately after the request is sent.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.http.waitingThreadsThreshold

Use this property to specify how many waiting connection requests are tolerated before releasing soft connections. A soft connection occurs when a client engine maintains connection objects for some hosts after the connection is closed. By default, after five threads are waiting for connections, the client engine releases the soft connections.

Note: If all of the connections are being used, the custom property does not have an impact. In this situation, you can increase the maximum connection limit, the maximum number of threads, or both.

The default value for this custom property is 5.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.jaxrpc.client.publishwsdl

Specifies whether a WSDL file is published for a client web module. When this property is set to true, if an application contains a client web module, a WSDL file might be published for that client. If you do not want WSDL files published for your client applications, set this property to `false`.

The default value of this property is `true`.

gotcha:

- WSDL file publication is not available for JAX-RPC applications that only contain a client, .
- This property cannot be used for JAX-WS applications.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.soap.enable.legacy.get.behavior

Note: Starting in WebSphere Application Server Version 8, the SOAP with Attachments API for Java (SAAJ) methods `SOAPMessage.getSOAPHeader` and `getSOAPBody` now throw a `SOAPException` if there is no corresponding element in the message. Previously these methods would return a null if there was no corresponding element in the message. A System property is provided to revert the behavior to return null rather than throw an exception. The property is `com.ibm.websphere.webservices.soap.enable.legacy.get.behavior`. The default value of the property is `null` which is interpreted as `false`. To revert the behavior to returning a null, set the property to the String value `true`. Note that the previous behavior of returning null is not compliant with the SAAJ specification.

com.ibm.websphere.webservices.tempAttachDir

Use this property to specify the location on a storage device where you want the web services runtime to cache a copy of any attachment, that is greater than 32KB in size, that is being sent or received as part of a SOAP message.

For performance reasons, the web services runtime caches a temporary copy of any SOAP message attachment that is greater than 32KB in size. If you do not specify a value for this property, the cached copy of the attachment is typically sent to the default temporary directory for your operating system.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.transport.jms.messageType

Use this property to control the JMS message type that is used by the web services engine for SOAP over JMS components when sending request and response messages. To specify a JMS BytesMessage (`javax.jms.BytesMessage`) object, set the property to `BYTES` to indicate the body of the message is binary data. To specify a JMS TextMessage (`javax.jms.TextMessage`) object, set the property to `TEXT` to indicate the body of the message is string data.

The default value for this custom property is `BYTES`.

To learn more about the SOAP over JMS message types, see the configuring SOAP over JMS message types information.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.transport.OPTIMIZE_HTTP_HEADERS

Prior to Version 8, a JAX-WS client application for WebSphere Application Server might send a `SAVE_CONNECTION` HTTP header in a SOAP message. This additional header ensures that proper processing occurs by the application server that is hosting the JAX-WS web service. However, this `SAVE_CONNECTION` header and the additional processing is not necessary if the application server for the client and the application server host for the web service are both using WebSphere Application Server Version 7.0 Fix Pack 3 or later.

You can set the `com.ibm.websphere.webservices.transport.OPTIMIZE_HTTP_HEADERS` custom property to `false` to enable the `SAVE_CONNECTION` header to ensure proper processing by older application server levels. By default, this custom property is set to `true`, which disables the JAX-WS client from sending the `SAVE_CONNECTION` header. If you need to change this default behavior, set the custom property to `false` on the application server that is hosting the JAX-WS client application.

Important: You must verify that the application server for the client and application server host for the web service are both using Version 7.0 Fix Pack 3 or later.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.transport.ssl.loadFromPolicyBinding

Use this property to control whether JAX-WS applications use SSL transport bindings or the system default SSL settings when the client is a managed client, and the client and the server are in different application servers.

When you create an SSL binding, this property is automatically added to the bindings file, and set to true. This setting enables SSL transport bindings to be used for JAX-WS applications when the client is a managed client, and the client and the server are in different application servers. If no bindings are attached to your JAX-WS application, set this property to `false`.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.UseWSFEP61ScanPolicy

Use this property to control whether the product scans WAR 2.4 and earlier modules for JAXWS components and semi-managed service clients. By default, these legacy WAR modules are only scans for semi-managed service clients.

The default value for this custom property is `false`.

You must set this property to `true` for each server and administrative server that requires a change in the default value.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.websphere.webservices.WSDL_Generation_Extra_Classpath

Use this property to set the location of the shared class files. The wsgen command-line tool generates the necessary artifacts that are required for Java API for XML Web Services (JAX-WS) applications when they start from Java code. However, the wsgen command-line tool might not locate the necessary class files and append the following error messages to the log file:

```
Caused by: java.lang.NoClassDefFoundError
...
at com.ibm.ws.websvcs.wsdl.WASWSDLGenerator.wsgen(WASWSDLGenerator.java:521)
at com.ibm.ws.websvcs.wsdl.WASWSDLGenerator.generateWsdl(WASWSDLGenerator.java:183)
```

Use this property to provide the fully qualified location to the missing class files. With this custom property, you can provide fully qualified paths to multiple Java archives (JAR) and directories and separate them using a semicolon (;).

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.amm.scan.context.filter.archives

Use this property to provide a list of archives, or utility JAR files, that do not contain annotations. Archives or utility JAR files specified for this property are not be scanned for annotations.

When a Java Platform, Enterprise Edition (Java EE) 5 or 6 application is deployed or updated, the Annotations Metadata Manager (AMM) facility scans all of the annotation metadata. This scanning process can negatively affect the amount of time required to deploy an application. If the application includes archives or utilities that do not contain annotations, you can list these archives and utilities as the value for this property. If the application includes Java Packages that do not contain annotations, you can list them as the value for the Ignore-Scanning-Packages property.

The values specified for this properties are case sensitive, and must be expressed as a single string with a comma followed by a space used to separate the names of the archives or utility JAR files. Wildcards and REGEX expressions are not permitted.

As an alternative to using this custom property, you can add the Ignore-Scanning-Archives property to one of the following files or modules, and specify the archives and utilities that you do not want scanned as the value of that property:

- The amm.filter.properties file that is located in the was_home/properties directory.
- The amm.filter.properties file that is located in the profile_home/properties directory
- The manifest file of an application, META-INF/MANIFEST.MF
- The manifest of a web or Enterprise JavaBeans (EJB) module within an application

Values specified in the amm.filter.properties files are merged with those found in this custom property to form a server scoped set of filters. This merged set of filters applies to all of the applications that are deployed on that server.

Values specified in the manifest file of an application are merged with the server scoped set of filters to form a module scoped superset that applies to all modules within that application.

Values specified in the manifest file of a web or Enterprise JavaBeans (EJB) module are merged with the module scoped set of filters. This merged set of filters only applies to that module.

gotcha: Exercise caution if you update a manifest file. Manifest files have line length limitations, and other constraints that must be adhered to.

Example:

```
Ignore-Scanning-Archives : ant.jar,  avalon-framework-4.2.0.jar, axis.jar, CICS.jar, xerces.jar
```

com.ibm.ws.amm.scan.context.filter.packages.
Use this property to provide a list of Java Packages that do not contain annotations. The Java classes specified for this property are not scanned for annotations.

When a Java Platform, Enterprise Edition (Java EE) 5 or 6 application is deployed or updated, the Annotations Metadata Manager (AMM) facility scans all of the annotation metadata. This scanning process can negatively affect the amount of time required to deploy an application. If the application includes Java Packages that do not contain annotations, you can list them as the value for this property. If the application includes archives or utilities that do not contain annotations, you can list them as the value for the Ignore-Scanning-Archives property.

The value specified for this property are case sensitive and must be expressed as a single string with a comma followed by a space used to separate the names of the Java Packages. Wildcards and REGEX expressions are not permitted.

As an alternative to using this custom property, you can add the Ignore-Scanning-Packages property to one of the following files or modules, and specify the archives and utilities that you do not want scanned as the value of that property:

- The amm.filter.properties file that is located in the was_home/properties directory
- The amm.filter.properties file that is located in the profile_home/properties directory
- The manifest file of an application, META-INF/MANIFEST.MF
- The manifest of a web or Enterprise JavaBeans (EJB) module within an application

Values specified in the amm.filter.properties files are merged with those found in this custom property to form a server scoped set of filters. This merged set of filters applies to all of the applications that are deployed on that server.

Values specified in the manifest file of an application are merged with the server scoped set of filters to form a module scoped superset that applies to all modules within that application.

Values specified in the manifest file of a web or Enterprise JavaBeans (EJB) module are merged with the module scoped set of filters. This merged set of filters only applies to that module.

gotcha: The following example is properties file centric and cannot be used as is for a manifest file. Manifest files have a 72 byte line length limit, as well as other constraints that must be adhered to.

Example:

```
Ignore-Scanning-Packages : org.apache.avalon, org.apache.batik, org.apache.commons
```

com.ibm.ws.application.enhancedScanning

Use this property to disable several optimizations that decreases the time to deploy and start enterprise applications. The optimizations primarily involve Java Platform, Enterprise Edition (Java EE 5)-enabled applications. When you set this property to false, the following updates are disabled:

- A new cache for modules files
- A new cache for module classloading
- Alternate code paths for annotation processing.

If you set this property to `false`, you might experience decreases in performance. Thus, by default, this property value is set to `true`.

com.ibm.ws.cache.CacheConfig.alwaysSetSurrogateControlHdr

Use this property to force the surrogate-control header from the dynamic cache service to always be set on the response. The surrogate-control header contains the metadata that the Edge Side Include (ESI) processing needs to correctly generate, and invalidate the cached content in the ESI cache.

The default value is `false`, which means that the surrogate-control header might not be set on the response.

If you decide to use this custom property, specify it as an application server JVM custom property unless otherwise indicated within the context of a specific task.

com.ibm.ws.cache.CacheConfig.cascadeCachespecProperties

Use this property to enable child pages or fragments to inherit the cascade of save-attributes, and store-cookies properties from their parent pages or fragments.. The default value is false.

The default behavior of the dynamic cache service is to store the request attributes for a child page or fragment, if not explicitly overridden in the cache specification. An application server can run into an Out-Of-Memory condition in scenarios where these request attributes get too large. If the attributes saved by default are not serializable, then the disk offload of these cache entries results in java.io.NotSerializableExceptions.

If you decide to use this custom property, specify it as an application server JVM custom property unless otherwise indicated within the context of a specific task.

com.ibm.ws.cache.CacheConfig.filteredStatusCodes

Use this property to indicate error situations in which you do not want the dynamic cache service to cache the servlet output.

The value specified for this property is a space delimited list of HTTP response error codes. If the status code returned from a cache miss matches one of the listed response error codes, the dynamic cache service does not cache the data that was obtained in response to an HTTP request.

If you decide to use this custom property, specify it as an application server JVM custom property unless otherwise indicated within the context of a specific task.

com.ibm.ws.CacheConfig.alwaysTriggerCommandInvalidations

Use this property to ensure that command invalidations are triggered regardless of the skipCache attribute.

When a request object contains the `<previewRequest>` attribute the dynamic cache sets the skipCache attribute to true. When the skipCache attribute is true, commands are not always invalidated. Set the com.ibm.ws.CacheConfig.alwaysTriggerCommandInvalidations custom property to true to ensure that command invalidations are triggered regardless of the skipCache attribute. When you set this custom property, it affects all cache instances. The default value for this property is false.

This custom property is set on the application server level only.

If you decide to use this custom property, specify it as an application server JVM custom property unless otherwise indicated within the context of a specific task.

com.ibm.ws.classloader.allowDisposedClassLoad

Use this property to specify whether the JVM should fully dispose of an application class loader when the application is stopped. If the JVM does not fully dispose of an application class loader, classes can still be loaded from that class loader.

When this property is set to a value of `true`, the JVM does not fully dispose of an application class loader when the application is stopped.

The default value for this property is false.

transition: The default value for this property was true in previous versions of the product.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.classloader.strict

Use this property to enable the WebSphere Application Server application class loader to provide access to the META-INF directory through a getResources call even if the META-INF directory path does not include a trailing slash

The WebSphere Application Server application class loader does not, by default, provide access to the META-INF directory through a getResources call unless a trailing slash is specified at the end of the META-INF directory path. If you want the WebSphere Application Server application class loader to provide access to the META-INF directory through a getResources call even if a trailing slash is not specified at the end of the META-INF directory path, set this property to `true`.

The default value for this property is `false`.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.classloader.zipFileCacheSize

Use this property to specify the maximum number of application JAR files that can be held open for resource and class loading. Reducing the number of times JAR files must be opened, improves the performance of applications that are resource or class loading intensive.

When the specified limit of open JAR files is reached, the class loader starts to close and remove JAR files based on the last time they were accessed. The most recently accessed JAR files are kept open. The value specified for this property should be based on the total number of application JAR files that are frequently accessed.

The default value for this property is `8`. Specifying a value of `0` disables the cache and prevents application JAR files from being held open.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.el.reuseEvaluationContext

Use this property to indicate that the same EvaluationContext object can be reused on a per thread basis.

Typically, during the evaluation of expressions the Unified EL code creates a new org.apache.el.lang.EvaluationContext object for each call that is made. Because these objects are subsequently made available for garbage collection, as the number of objects created increases, memory consumption and garbage collection also increases. Setting the com.ibm.ws.el.reuseEvaluationContext property to `true` enables the same EvaluationContext object to be reused on a per thread basis., thereby decreasing memory consumption and the amount of garbage collection that needs to occur.

The default value is `false`.

com.ibm.ws.iiop.channel.disableOnewayLocateRequiredMessage

Use this property to control whether the CWZGB0005I message displays whenever a oneway GIOP request against an object requires further locate-forward addressing. If the property is set to `false`, the CWZGB0005I message is displayed. If the property is set to `true`, the CWZGB0005I message is not displayed.

The default value for this property is false.

com.ibm.ws.management.connector.soap.logClientInfo

Use this property to indicate whether you want to log the host, port, and username of SOAP client requests. When this property is set to `true`, SOAP client details are logged in SystemOut.log. These details are also added to trace.log if the trace level for the SOAP connector is set to `all`.

The default value for this property is `false`.

com.ibm.ws.management.repository.tempFileKeepTimeMinutes

Use this property to specify, in minutes, how long a file is kept in the configuration repository temporary directory before the configuration repository temporary directory cleanup task can delete that file from the directory. The default value for this property is 1440 minutes, which is equal to 24 hours. In previous versions of the product, a file was kept for 60 minutes.

The default value is typically sufficient for performing needed cleanup without deleting files that are in use. However, there might be situations where you need to specify a larger, or smaller value. You can specify a minimum value of 60 minutes for this property. However, it is recommended that you specify a value that is equivalent to several hours to account for situations where very large files are being transferred or synchronized, or where a network is slow, and file transfer operations are taking a long time. In these situations, if too short a time period is specified, it is possible for a file to be deleted while it is still being transferred.

If the com.ibm.ws.management.repository.tempFileSweepIntervalMinutes property is set to `0`, the cleanup function is disabled, and any files left behind after a server process failure, remain in the configuration repository temporary directory until they are manually removed, or the cleanup function is enabled.

gotcha: If an invalid value is specified for this property, the default value is used.

com.ibm.ws.management.repository.tempFileSweepIntervalMinutes

Use this property to specify, in minutes, how frequently the configuration repository temporary directory cleanup task runs. This task removes files from the configuration repository temporary directory that were not properly removed because of a server process failure.

The cleanup task always runs when the server starts, and then again after the time length specified for this property expires. The default value for this property is 720 minutes, which is equivalent to 12 hours. This length of time is typically sufficient for the configuration repository temporary directory cleanup task to successfully complete the cleanup process. You can disable this cleanup function by setting this property to `0`.

In previous versions of the product, the cleanup task ran when the server started, and then ran again every 30 minutes.

gotcha: If an invalid value is specified for this property, the default value is used.

com.ibm.ws.odr.plugincfg.config.ASDisableNagle

Use this property to specify whether you want to disable the Nagle algorithm for the connection between the plug-in and the proxy server.

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`, which means that the Nagle algorithm is enabled for the connection between the plug-in and the proxy server.

com.ibm.ws.odr.plugincfg.config.AcceptAllContent

Use this property to specify whether you can include content in POST, PUT, GET, and HEAD requests when a Content-Length or Transfer-encoding header is contained in the request header. You can specify one of the following values for this attribute:

- `True` if content is to be expected and read for all requests.
- `False` if content only is only to be expected and read for POST and PUT requests.

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`.

com.ibm.ws.odr.plugincfg.config.AppServerPortPreference

Use this property to specify which port number is used to build URI's for a sendRedirect.

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `HostHeader`.

com.ibm.ws.odr.plugincfg.config.ChunkedResponse

Use this property to specify whether the plug-in groups the response to the client when a Transfer-Encoding : Chunked response header is present in the response.

You can specify one of the following values for this attribute:

- `True` if the plug-in is to chunk the response to the client when a Transfer-Encoding : Chunked response header is present in the response.

- `False` if the response is not to be chunked.

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`.

com.ibm.ws.odr.plugincfg.config.IISDisableNagle

Use this property to specify whether you want to disable the nagle algorithm.

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`.

com.ibm.ws.odr.plugincfg.config.IISPluginPriority

Use this property to specify the priority in which the Web server loads the plug-in. You can specify one of the following values for this attribute:

- High
- Medium
- Low

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `High`.

com.ibm.ws.odr.plugincfg.config.IgnoreDNSFailures

Use this property to specify whether the plug-in is to ignore DNS failures within a configuration when started.

When this property is set to true, the plug-in ignores DNS failures within a configuration and starts successfully if at least one server in each ServerCluster resolves the host name. Any server for which the host name is not resolved is marked unavailable for the life of the configuration. The host name is not resolved later during the routing of requests. If a DNS failure occurs, a log message is written to the plug-in log file and the plug-in continues initializing instead of the Web server not starting.

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`.

com.ibm.ws.odr.plugincfg.config.RefreshInterval

Use this property to specify, in seconds, how frequently the plug-in should check the configuration file for updates or changes. The plug-in checks the file for any modifications that occur since the plug-in configuration was loaded.

In a development environment where frequent changes occur, set the time interval to less than 60 seconds.

In a production environment, you should set a higher value than the default value, because updates to the configuration do not occur as frequently.

If the plug-in reload is not successful, the plug-in log file contains an error message, and the previous configuration is used until the plug-in configuration file successfully reloads. Refer to the plug-in log file for more information if an error occurs.

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `60`.

com.ibm.ws.odr.plugincfg.config.ResponseChunkSize
Use this property to specify, in kilobytes, the maximum chunk size the plug-in should use when reading the response body. For example, Config ResponseChunkSize="*N*">, where *N* equals the chunk size.

By default, the plug-in reads the response body in 64k chunks until all of the response data is read. This process might cause a performance problem for requests where the response body contains large amounts of data. If the content length of the response body is unknown, a buffer size of *N* kilobytes is allocated and the body is read in *N* kilobyte size chunks, until the entire body is read. If the content length is known, then a buffer size of either content length or *N* is used to read the response body.

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `64`.

com.ibm.ws.odr.plugincfg.config.VHostMatchingCompat
Use this property to specify whether the plug-in should use the port number for virtual host matching. The following values can be specified:
- True for physically matching by using the port number for which the request is received.
- False for logically matching by using the port number contained in the host header.

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`.

com.ibm.ws.odr.plugincfg.config.TrustedProxyEnable
Use this property to specify whether the plug-in is to allow the inclusion of trusted proxies. The following values can be specified:
- True if you want to allow the inclusion of trusted proxies.
- False if you do not want to allow the inclusion of trusted proxies.

The trusted proxies are collected from the defined trusted security proxies.

This property is only valid for a proxy server, and applies to the Config element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`.

com.ibm.ws.odr.plugincfg.log.Name
Use this property to specify the fully qualified path to the log file to which the plug-in writes error messages.

This property is only valid for a proxy server, and applies to the Log element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `profileRoot/logs/http_plugin.log`.

com.ibm.ws.odr.plugincfg.log.LogLevel

Use this property to specify the level of detail of the log messages that the plug-in writes to the log. You can specify one of the following values for this attribute:

- Trace. All of the steps in the request process are logged in detail.
- Stats. The server selected for each request and other load balancing information relating to request handling is logged.
- Warn. All warning and error messages resulting from abnormal request processing are logged.
- Error. Only error messages resulting from abnormal request processing are logged.
- Debug. All of the critical steps performed in processing requests are logged.
- Detail. All of the information about requests and responses are logged.

This property is only valid for a proxy server, and applies to the Log element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `Error`.

gotcha: A lot of messages are logged at the trace level, which can cause the file system to fill up very quickly. Never use a trace setting in a normally functioning environment as it adversely affects performance.

com.ibm.ws.odr.plugincfg.cluster.CloneSeparatorChange

Use this property to indicate to the plug-in that the plus character (+) can be used as the clone separator.

Some pervasive devices cannot handle the colon character (:) that is used to separate clone IDs in conjunction with session affinity.

gotcha: If you use this custom property, you must change the proxy server configurations such that the proxy server separates clone IDs with the plus character instead the colon character.

This property is only valid for a proxy server, and applies to the ServerCluster element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`.

com.ibm.ws.odr.plugincfg.cluster.LoadBalance

Use this property to specify the appropriate load balancing option: `Round Robin` or `Random`.

The Round Robin implementation has a random starting point. The first proxy server is picked randomly. Round Robin is then used to pick proxy servers from that point forward. This implementation ensures that in multiple process based Web servers, all of the processes don't start up by sending the first request to the same proxy server.

The Random implementation also has a random starting point. However with this implementation all subsequent proxy servers are also randomly selected. Therefore, the same proxy server might get selected repeatedly while other proxy servers remain idle.

The default value is `Round Robin`.

com.ibm.ws.odr.plugincfg.cluster.PostSizeLimit

Use this property to specify, in bytes, the maximum number of bytes of request content that the plug-in is allowed to attempt to send to a server. If a request is received that is greater than the specified value, the plug-in ends the request

This property is only valid for a proxy server, and applies to the ServerCluster element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is -1, which indicates that there is no limit to the size of a request.

com.ibm.ws.odr.plugincfg.cluster.RemoveSpecialHeaders

Use this property to whether the plug-in is to add special headers to a request before it is forwarded to the server. These headers store information about the request that the application then uses. By default, the plug-in removes these headers from incoming requests before adding the required headers.

If you set this property to false, you introduce a potential security exposure headers from incoming requests are not removed.

This property is only valid for a proxy server, and applies to the ServerCluster element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `true`.

com.ibm.ws.odr.plugincfg.cluster.RetryInterval

Use this property to specify, in seconds, the amount of time that elapses between when a proxy server is marked down and when the plug-in reattempts to make a connection.

This property is only valid for a proxy server, and applies to the ServerCluster element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `60`.

com.ibm.ws.odr.plugincfg.odrIncludeStopped

Use this property to specify whether the plug-in is to allow the inclusion of stopped proxy servers. The following values can be specified:
- True if you want to allow the inclusion of stopped proxy servers.
- False if you do not want to allow the inclusion of stopped proxy servers.

This property is only valid for a proxy server.

The default value is `false`.

com.ibm.ws.odr.plugincfg.server.ConnectTimeout

Use this property to specify, in seconds, the amount of time the plug-in waits for a successful connection

Specifying a value for th property enables the plug-in to perform non-blocking connections with the proxy server. Such connections are beneficial when the plug-in is unable to contact the destination to determine if the port is available or unavailable.

When a value greater than `0` is specified, and a connection does not occur after that time interval elapses, the plug-in marks the proxy server unavailable, and proceeds with one of the other proxy servers defined in the cluster.

If no value is specified for this property, the plug-in performs a blocking connect in which the plug-in waits until an operating system times out and allows the plug-in to mark the proxy server unavailable.

This property is only valid for a proxy server, and applies to the Server element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `0`.

com.ibm.ws.odr.plugincfg.server.ExtendedHandShake

Use this property to indicate to the plug-in that it must ensure the availability of a proxy server before sending a request to that proxy server.

Typically, the plug-in marks a proxy server as stopped when a connect() ends. However, when a proxy firewall is between the plug-in and the proxy server, the connect() succeeds, even though the back-end proxy server is stopped. This situation causes the plug-in to not failover correctly to other proxy servers.

This property is only valid for a proxy server, and applies to the Server element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`.

com.ibm.ws.odr.plugincfg.server.MaxConnections

Use this property to specify the maximum number of pending connections to a proxy server that can flow through a Web server process at any point in time.

This property is only valid for a proxy server, and applies to the Server element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `-1`, which indicates that there is no maximum number for the number of pending connections to a proxy server that can flow through a Web server process at any point in time.

com.ibm.ws.odr.plugincfg.cluster.WaitForContinue

Use this property to specify whether to use the HTTP 1.1 100 Continue support before sending the request content to the proxy server.

Typically, the plug-in does not wait for the 100 Continue response from the proxy server before sending the request content. You should use HTTP 1.1 100 Continue support when configuring the plug-in to work with certain types of proxy firewalls.

This property is ignored for POST requests to prevent a failure from occurring if the proxy server closes a connection because of a time-out.

This property is only valid for a proxy server, and applies to the Server element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`.

com.ibm.ws.odr.plugincfg.property.ESIEnable

Use this property to enable or disable the Edge Side Include (ESI) processor. If the ESI processor is disabled, the other ESI elements in the file are ignored.

This property is only valid for a proxy server, and applies to the Property element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `true`.

com.ibm.ws.odr.plugincfg.property.ESIMaxCacheSize

Use this property to specify, in 1K byte units, the maximum size of the cache. The default maximum size of the cache is 1024K bytes (1 megabyte). If the cache is full, the first entry to be evicted from the cache is the entry that is closest to its expiration time.

This property is only valid for a proxy server, and applies to the Property element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `1024`.

com.ibm.ws.odr.plugincfg.property.ESIInvalidationMonitor

Use this property to specify whether or not the ESI processor receives invalidations from the proxy server.

This property is only valid for a proxy server, and applies to the Property element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `false`.

com.ibm.ws.odr.plugincfg.property.https.keyring
Use this property to specify the directory location of the SAF keyring when the protocol of the transport is set to HTTPS.

This property is only valid for a proxy server, and applies to the Property element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `profileRoot/etc/plugin-key.kdb`.

com.ibm.ws.odr.plugincfg.property.https.stashfile
Use this property to specify the location of the stashfile.

This property is only valid for a proxy server, and applies to the Property element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `profileRoot/node/etc/plugin-key.sth` .

com.ibm.ws.odr.plugincfg.property.PluginInstallRoot
Use this property to specify the installation path for the plug-in.

You must set this property, to the fully qualified path of the plug-in installation root. If you use the default value, the property does not display in the plugin-cfg.xml file.

This property is only valid for a proxy server, and applies to the Property element in the plugin-cfg.xml file that the proxy server automatically generates.

The default value is `""`.

com.ibm.ws.pm.checkingDBconnection
Use this property to specify whether the persistence manager is to continue checking the availability of a database, that was previously marked as unavailable, until a connection with that database is successfully established.

If a database service is down when the persistent manager attempts to establish a connection to that database, the database is marked as unavailable. Typically, the persistent manager does not re-attempt to establish a connection after a database is marked as unavailable. If you sent this property to `true`, the persistence manager continues to check the availability of the database until it is able to successfully establish a connection to that database.

The default value for this property is `false`.

com.ibm.ws.runtime.component.ResourceMgr.postBindNotify
Use this property to make the Connection Factory MBeans available when a resource adapter starts. Typically, when a resource adapter starts, the Connection Factory MBeans are not available for the resource adapter to query. However, certain resource adapters, such as the IMS DB Resource Adapter, require this functionality for initialization.

If you are not using a resource adapter that requires the availability of Connection Factory MBeans at initialization, add this property to your JVM settings and set the value to `false`.

The default value for this property is `true`.

com.ibm.ws.scripting.apptimeout

Use this property to specify, in seconds, the length of time that can elapse before an application installation, or an application update times out. The default value is 86400, which is equivalent to 24 hours.

Specifying a reasonable value for this property prevents the installation, or update process from continuing indefinitely when a situation occurs that prevents the installation, or update script from completing. For example, you might have a JACL script that updates an EAR file that cannot complete because the deployment manager that the script is connected to stops.

com.ibm.ws.sib.webservices.useSOAPJMSTextMessages

By default on WebSphere Application Server Version 6 or later, a SOAP over JMS web service message sent by the web services gateway is sent as a JmsBytesMessage.

Set this property to `true` to modify the default behavior and send a compatible JmsTextMessage. Setting this property modifies the default behavior for all outbound JMS web services invocations sent from the service integration bus.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.use602RequiredAttrCompatibility

Use the com.ibm.ws.use602RequiredAttrCompatibility custom property to specify whether the `<required>` attribute is evaluated prior to other attributes in the `cachespec.xml` file.

In Version 6.0.2, if you set the `<required>` attribute to false, then all of the other attributes within the `cachespec.xml` file are ignored and a cache ID is generated.

Note: In later versions, by default, the `<required>` attribute is evaluated along with all of the other attributes to determine if a cache ID is generated.

Note: If you set the com.ibm.ws.use602RequiredAttrCompatibility custom property to true, then the behavior of the `cachespec.xml` file is reverted back to the behavior in Version 6.0.2. The `<required>` attribute is evaluated prior to other attributes in the `cachespec.xml` file. The default value for this custom property is false. When you set this JVM custom property, which only applies to the application server level, it affects all of the dynamic cache users.

If you decide to use this custom property, specify it as an application server JVM custom property unless otherwise indicated within the context of a specific task.

com.ibm.ws.webservices.allowNoSOAPActionHeader

Use this property to enable the web services engine to tolerate an incoming web service request that **does not** contain a SOAPAction header. This property must be set at the application server level.

The SOAP specification states that an HTTP request message must contain a SOAPAction HTTP header field with a quoted empty string value, if in the corresponding WSDL description, the soapAction of soapbind:operation is either not present, or present with an empty string as its value. However, if you want the web services engine to handle requests that do not contain a SOAPACTION header, add this property to the application server settings and set it to `true`.

When this property is not specified, or is not set to true, if an incoming SOAP request message does not contain a SOAPAction header, a SOAP Fault is returned to the client

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.allowStatusCode202OneWay

Use this property to allow a JAX-RPC one-way service to send a 202 status code instead of a 200 status code.

A JAX-RPC one-way service deployed on WebSphere Application Server normally returns a 200 HTTP status code. Some JAX-RPC implementations cannot tolerate a 200 status code, preferring a 202 instead. According to the Basic Profile Version 1.1, both 200 and 202 are valid status codes for one-way services.

If the property is set to true, then the JAX-RPC one-way service returns a 202 status code.

The default value is false.

This property only applies to the application server JVM.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.appendRootCauseToWSF

If you are a JAX-RPC user, use this property to loop though all the exception causes and concatenate the details into the Fault details in the response.

The Fault details in the response typically does not contain any information about the original exception. This lack of information can make problem determination more difficult if the developer does not have access to the logs from the service provider.

The default value is false.

This property only applies to the application server JVM.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.contentTransferEncoding

Use this property to specify a range of bits for which .XML-encoding is disabled. Typically any integer that is greater than 127 is XML-encoded. When you specify this property:
- Web services disables encoding for integers that fall within the specified range.
- The HTTP transport message contains a ContentTransferEncoding header that is set to the value that is specified for this custom property.

Specify `7bit`, if you only want integers greater than 127 encoded. Specify `8bit`, if you only want integers greater than 255 encoded. Specify `binary`, if you want encoding disabled for all integers.

The default value is `7bit`.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.disableSOAPElementLazyParse

Use this property to disable lazy parsing of SOAPElements. Lazy parsing is designed for situations where the client is not parsing the SOAPElement. If a client is parsing the SOAPElement with SAAJ, it is better to not delay parsing by the web services component.

You can set this property as a JVM custom property at either the server or client level. When this property is set at either the server or client level, the setting applies to all applications on the JVM. The default value for this property is `false`.

You can also use an application assembly tool to specify this property as a new web service description binding entry for the port component binding, if you want to disable lazy parsing of SOAPElements on an application-by-application basis for a particular server, instead of for all of the servers that are managed by the deployment manager.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.engine.transport.jms.propagateOneWaySystemExceptions

Use this property to enable exceptions that occur during the processing of a one-way JMS Web service to be propagated back to the EJB container. This propagation makes normal error recovery possible.

If this property is set to `false`, an exception is wrapped in a WebServicesFault message and sent back to the client. Because the Web service is not aware of the exception, no recovery is attempted.

The default value for this property is `false`.

gotcha: This property does not apply to a one-way HTTP Web service, or to two-way JMS requests.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.HttpRedirectWithProxy

Set this property to `true` to allow HTTP redirect requests to be sent through the proxy server. When you set this property to true, you change the default behavior for all outbound HTTP redirect requests sent from the JAX-RPC runtime. When this property is set to `false`, a redirect request is sent to a remote server directly even though a proxy server is configured.

The default value for this property is false.

If you decide to use this custom property, you must specify it as an proxy server JVM custom property.

com.ibm.ws.webservices.ignoreUnknownElements

Use this property to control whether clients can ignore extra XML elements that are sometimes found within literal SOAP operation responses.

Setting this property to `true` provides you with the flexibility of being able to update your server code to include additional response information, without having to immediately update your client code to process this additional information. However, when this functionality is enabled, the checking of SOAP message against the expected message structure is more relaxed than when this property is set to `false`.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.jaxrpc.parse.tolerate.invalid.namespace

Use this property to enable the JAX-RPC engine to use a more tolerant algorithm when determining whether to accept an incoming JAX-RPC message.

Typically, if an incoming JAX-RPC message uses an invalid namespace for a body element, the JAX-RPC engine rejects the message. If you set this property to `true`, the JAX-RPC engine uses a more tolerant algorithm that ignores the namespace mismatch.

The default value is `false`

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.resolveXMLSchemaDTD

Use this custom property to enable a JAX-RPC application to properly start even if the schema or WSDL file that is represented in the _AbsoluteImportResolver class also references the http://www.w3.org/2001/XMLSchema.dtd DTD.

When you run on a host that is not connected to the Internet, a JAX-RPC application that is packaged with the _AbsoluteImportResolver class might not start properly. The following error might exist in the log files:

```
WSDDPort      W com.ibm.ws.webservices.engine.deployment.wsdd.WSDDPort expand
WSWS3114E: Error: Internal error.
java.net.UnknownHostException: www.w3.org
```

Setting this custom property to `true` enables a JAX-RPC application to properly start even if the schema or WSDL file that is represented in the _AbsoluteImportResolver class also references the http://www.w3.org/2001/XMLSchema.dtd DTD.

The default value is `false`

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.searchForAppServer

Use this property to control whether `DualMetaDataLoaderImpl E loadWebContainerPorts could not find any http or https ports` messages are sent to the system log.

Depending on your system configuration, if a web services application is installed across both a web server and an application server, your system might issue this message, indicating that an error occurred even though this is a valid configuration. Therefore if you install any of your web services applications across both a web server and an application server, you might not want these messages sent to the system log.

If the com.ibm.ws.webservices.searchForAppServer property is set to `true`, any `DualMetaDataLoaderImpl E loadWebContainerPorts could not find any http or https ports` messages that are issued are not sent to the system log. If this property is not specified or is set to `false`, these messages are sent to the system log.

The default value for this property is false.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.serialize.2DimArray.asArrays

Use this property to cause the JAX-RPC runtime to serialize two-dimensional XML arrays as a series of arrays.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

The default value for this property is false.

The following message snippet illustrates a series of elements, which is a valid format for representing two-dimensional XML arrays when this property is set to `false`.

```
<p565:sayHelloResponse xmlns:p565="http://ibm.com">
    <sayHelloReturn xsi:type="soapenc:Array"
soapenc:arrayType="xsd:string[2,3]">
      <item xsi:type="xsd:string">array1 element1</item>
      <item xsi:type="xsd:string">array1 element2</item>
      <item xsi:type="xsd:string">array1 element3</item>
```

```
        <item xsi:type="xsd:string">array2 element1</item>
        <item xsi:type="xsd:string">array2 element2</item>
        <item xsi:type="xsd:string">array2 element3</item>
    </sayHelloReturn>
  </p565:sayHelloResponse>
```

The following message snippet illustrates an array of two arrays, with each array containing three elements, which is a valid format for representing two-dimensional XML arrays when this property is set to `true`.

```
<p565:sayHelloResponse xmlns:p565="http://ibm.com">
    <sayHelloReturn xsi:type="soapenc:Array"
soapenc:arrayType="xsd:string[][2]">
        <item soapenc:arrayType="xsd:string[3]">
          <item>array1 element1</item>
          <item>array1 element2</item>
          <item>array1 element3</item>
        </item>
        <item soapenc:arrayType="xsd:string[3]">
          <item>array2 element1</item>
          <item>array2 element2</item>
          <item>array2 element3</item>
        </item>
    </sayHelloReturn>
  </p565:sayHelloResponse>
```

com.ibm.ws.webservices.serializeDetailElementUsingDefaultNamespace

Use this property to specify whether the application server uses an actual prefix name to locate the namespace that defines the Fault detail, or uses a default namespace to define the Fault detail.

When a JAX-RPC Web service responds with a SOAP Fault, an actual prefix name is typically used to locate the namespace that defines the contents of the Fault detail. Following is an example of the message that the application server typically issues in this situation:

```
<soapenv:Fault
      xmlns:soapenv=
      "http://schemas.xmlsoap.org/soap/envelope/">
  <faultcode xmlns="http://sample">
      sampleFault
  </faultcode>
  <faultstring>sample text</faultstring>
  <detail encodingStyle="">
    <sampleFault
      xmlns="http://sample">
      ...
    </sampleFault>
  </detail>
<soapenv:Fault>
```

If your application server needs to communicate with .Net clients, and these .Net clients require the use of a default namespace to define the contents of the Fault detail, set this property to `true`. When this property is set to `true`, the message that the application server issues is similar to the message that was sent from a version previous to a Version 6.x application server.

The default value for this property is false.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.webservices.suppressHTTPRequestPortSuffix

Use this property to control whether a port number can be left in an HTTP POST request that sends a SOAP message.

Some web service implementations do not properly tolerate the presence of a port number within the HTTP POST request that sends the SOAP message. If you have a web service client that needs to inter-operate with web service that cannot tolerate a port number within an HTTP POST request that sends a SOAP message, set this custom property to `true`.

When you set this property to `true`, the port number is removed from the HTTP POST request before it is sent.

gotcha: You must restart the server before this configuration setting takes affect.

The default value for this custom property is `false`.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.websvcs.attachments.sizethreshold

Use this property to specify, in bytes, the maximum size of an attachment on the JAX-WS client or service that can be written to memory. By default, the maximum attachment size is set to 102400 bytes. With this value, if an attachment exceeds 100 KBs, it is cached to the file system instead of written to memory. When you use this custom property, as you increase the maximum cache size, there is a greater impact on performance and, potentially, to the Java heap.

Note: To specify the maximum size of an attachment on the JAX-RPC client or service, see the com.ibm.websphere.webservices.attachements.maxMemCacheSize custom property.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.websvcs.suppressHTTPRequestPortSuffix

Use this property to prevent the JAX-WS runtime from appending the port number to the HTTP Host header value to a request.

A JAX-WS client might receive a java.io.IOException in response to a request, especially if there is a non-IBM web server located between the client and the web service the client is trying to call. This intermediary server might not understand where to route the request because the JAX-WS runtime has appended the port number to the HTTP Host header value. For example, JAX-WS runtime might have changed the header value from the endpoint URL lilygirl.austin.mycompany.com to the URL lilygirl.austin.mycompany.com:80, which includes the port number.

To prevent the JAX-WS runtime from appending the port number to the HTTP Host header value, add this custom property to your JVM settings, and set it to `true`. When this property is set to `true`, the Host header only contains the hostname of the endpoint URL; for example, Host: lilygirl.austin.mycompany.com

The default value for this property is `false`, which means that, the port number is appended to a Host header value.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.websvcs.unmanaged.client.dontUseOverriddenEndpointUri

You might want a request by an unmanaged JAX-WS client service to be sent to the endpoint URL that is specified in the **Overridden endpoint URL** field on the administrative console. The value of this managed field, which is set as part of the web services client port configuration, overwrites the endpoint that is specified in the WSDL file. For more information on this field, read about the web services client port.

Note: If you have either all managed clients or a mixture of both managed and unmanaged clients, you can edit the **Overridden endpoint URL** field in the administrative console. However, if you do not have any managed clients, you cannot edit the field.

Normally, you do not want an unmanaged JAX-WS client service to access this managed client service function. However, you might depend on unmanaged JAX-WS client services accessing this URL. By default, the com.ibm.ws.websvcs.unmanaged.client.dontUseOverriddenEndpointUri custom property is set to false to allow unmanaged JAX-WS client services to access the endpoint URL that overwrites the endpoint in the WSDL file.

This custom property is set on the application server level where a JAX-WS client is installed or a Java EE client exists if you run the launchClient.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.ws.ws.wsba.protocolmessages.twoway

Use this property to improve the performance of an application server that is handling requests for Web Services Business Activities (WS-BA). Specifying true for this custom property improves application server performance when WS-BA protocol messages are sent between two application servers. The default value for this property is true.

gotcha: If you decide to use this custom property, the property must be set on the application server that initiates the requests. It does not have to be set on the application server that receives the requests.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

com.ibm.wsspi.amm.merge.ignoreValidationExceptions

Use this custom property to indicate to the JVM that it should ignore validation exceptions that might occur during EJB processing. When an application is configured with necessary classes defined in shared libraries during EJB processing, incomplete information may be generated. As a result, a validation exception might occur and the following exception message may appear:

```
AnnotativeMetadataManagerImpl merge caught exception while merging com.ibm.wsspi.amm.validate.ValidationException:
the interface com.xyz.app.myappRemote does not define a
valid remote business interface; the method mygetMethod does not
conform to RMI rules.
```

Set this property to `true` if you want the JVM to ignore these validation exceptions.

The default value is false.

If you decide to use this custom property, you must specify it as a JVM custom property for the application server.

com.ibm.xml.xlxp.jaxb.opti.level

Use the com.ibm.xml.xlxp.jaxb.opti.level custom property to control whether optimization methods are enabled for Java Architecture for XML Binding (JAXB) unmarshalling (deserialization) and marshalling (serialization). The following table lists the supported values for this custom property and their effect on applications and web services.

Table 56. Supported values for the custom property. The table includes the custom property value and the effect of the custom property on applications and web services.

Custom property value	Effect
com.ibm.xml.xlxp.jaxb.opti.level=0	Optimization methods are not enabled.

Table 56. Supported values for the custom property (continued).   The table includes the custom property value and the effect of the custom property on applications and web services.

Custom property value	Effect
com.ibm.xml.xlxp.jaxb.opti.level=1	Only unmarshalling optimization methods are enabled.
com.ibm.xml.xlxp.jaxb.opti.level=2	Only marshalling optimization methods are enabled.
com.ibm.xml.xlxp.jaxb.opti.level=3	Both unmarshalling and marshalling optimization methods are enabled, which is the default value.

For optimum performance, set the custom property value to 3. This value increases throughput for web services and applications that use JAXB directly. If you are experiencing issues with optimization after setting this value, change the value to 0 as a temporary workaround.

You can set this custom property on the application server level only.

config_consistency_check

Use this property to optionally turn off the default workspace consistency process. The deployment manager maintains a master configuration repository for the entire cell. By default, when the configuration changes, the product compares the configuration in the workspace with the master repository to maintain workspace consistency. However, the consistency verification process can cause an increase in the amount of time to save a configuration change or to deploy a large number of applications. The following factors influence how much time is required:

- The more application servers or clusters there are defined in cell, the longer it takes to save a configuration change.
- The more applications there are deployed in a cell, the longer it takes to save a configuration change.

If the amount of time required to change a configuration change is unsatisfactory, you can add the `config_consistency_check` custom property to your JVM settings and set the value of this property to false.

Note: The `config_consistency_check` custom property affects the deployment manager process only. It does not affect other processes including the node agent and application server processes. The consistency check is not performed on these processes. However, within the `SystemOut.log` files for these processes, you might see a note that the consistency check is disabled. For these non-deployment manager processes, you can ignore this message.

deactivateWildCardURIMapping

Use this property to enable the plugin-cfg.xml file generator to recognize the URI patterns specified on the file.serving.patterns.allow attribute in the ibm-web-ext.xmi file for a web application.

The plugin-cfg.xml file generator only recognizes the URI patterns specified on the file.serving.patterns.allow attribute if the FileServingEnabled attribute in that ibm-web-ext.xmi file is set to `true`. However, when the the FileServingEnabled attribute is set to true, the plugin-cfg.xml file generator automatically adds the wildcard URI mapping, /* , to the plugin-cfg.xml file, which negates the usefulness of defining unique file serving patterns.

Setting the the deactivateWildCardURIMapping property to `true` prevents the plugin-cfg.xml file generator from adding the /* to the plugin-cfg.xml file, and enables the plugin-cfg.xml file generator to recognizes the URI patterns specified on the file.serving.patterns.allow attribute. If this property is not added to the JVM settings, or is set to `false` , the /* is automatically added to the plugin-cfg.xml file.

This property is set at the deployment manager level.

disableWSAddressCaching

Use this property to disable address caching for web services. If your system typically runs with lots of client threads, and you encounter lock contention on the wsAddrCache cache, you can set this custom property to `true`, to prevent caching of the web services data.

The default value for this property is `false`.

DRS_THREADPOOL_MINSIZE

Specifies the minimum number of threads to allow in the data replication service (DRS) thread pool.

When an application server starts, threads are not initially assigned to the thread pool. Threads are added to the thread pool as the workload that is assigned to the application server requires them and until the number of threads in the pool equals the number of threads that are specified by this custom property. After this point in time, additional threads are added and removed as the workload changes. However, the number of threads in the pool never decreases below the number that is specified by this custom property, even if some of the threads are idle.

The default value for this custom property is 40 threads.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

DRS_THREADPOOL_MAXSIZE

Specifies the maximum number of threads to maintain in the DRS thread pool.

The default value for this custom property is 100 threads.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

DRS_THREADPOOL_ISGROWABLE

Specifies whether the number of threads can increase beyond the maximum size that is configured for the DRS thread pool.

The maximum number of threads that can be created is constrained only within the limits of the Java virtual machine and the operating system. When a thread pool, that is allowed to grow, expands beyond the maximum size, the additional threads are not reused and are discarded from the pool after processing the work items for which they were created is completed.

The default value for this custom property is false.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

dynacache.jms.cacheInstance

When dynamic cache service multi-cell and multi-core group invalidation is enabled, use this property to specify the cache instance to use for processing invalidations. Setting this property to "*" causes invalidation IDs to be processed on all cache instances. This property should be set on the service integration bus servers.

By default, the baseCache cache instance is used. You can use the administrative console to determine the names of the cache instances that are defined for your system.

If you decide to use this custom property, specify it as an application server JVM custom property unless otherwise indicated within the context of a specific task.

dynacache.jms.connRetryInterval

When dynamic cache service multi-cell and multi-core group invalidation is enabled, use this property to specify the number of seconds that a cluster member waits before attempting to reconnect to a service integration bus server.

The default value for this property is `30`.

If you decide to use this custom property, specify it as an application server JVM custom property unless otherwise indicated within the context of a specific task.

dynacache.jms.invProcessingDelay

When dynamic cache service multi-cell and multi-core group invalidation is enabled, use this property to specify the number of seconds the service integration bus server queues invalidation IDs before processing them.

The default value for this property is `20`.

If you decide to use this custom property, specify it as an application server JVM custom property unless otherwise indicated within the context of a specific task.

dynacache.jms.numStoredInvalidations

When dynamic cache service multi-cell and multi-core group invalidation is enabled, use this property to specify the maximum number of invalidation IDs a cluster member can store while waiting for a service integration bus server to become available. After the threshold is reached, the oldest invalidation IDs are removed as new IDs are added.

The default value for this property is `10000`.

If you decide to use this custom property, specify it as an application server JVM custom property unless otherwise indicated within the context of a specific task.

invocationCacheSize

Use this property to control the size of the invocation cache. The invocation cache holds information for mapping request URLs to servlet resources. A cache of the requested size is created for each worker thread that is available to process a request. The default size of the invocation cache is 50. If more than 50 unique URLs are actively being used (each JavaServer Page is a unique URL), you should increase the size of the invocation cache.

A larger cache uses more of the Java heap, so you might also need to increase the maximum Java heap size. For example, if each cache entry requires 2KB, maximum thread size is set to 25, and the URL invocation cache size is 100; then 5MB of Java heap are required.

You can specify any number higher than 0 for the cache size. Setting the value to zero disables the invocation cache.

java.util.logging.configureByLoggingPropertiesFile

Use this custom property to specify whether the JVM uses the logging.properties file to configure JSR-47 logging.

If this property is not added to the JVM configuration settings, or is set to `false`, the configuration settings contained in the logging.properties file are not picked up because the product overrides the base JSR47 logging configuration with the java.util.logging.manager=com.ibm.ws.bootstrap.WsLogManager system property setting. In this situation, only logging settings that can be changed programmatically, such as the addition of handlers, and formatters, can be modified.

When this property is set to `true`, the JVM still configures the WsLogManager as the LogManager, but during server startup, the logging configuration for applications using JSR-47 logging is initialized based on settings in the logging.properties file. Refer to the Java Utility Logging API documentation for valid logging properties and format that can be specified in the logging.properties configuration file.

gotcha: Do not assign java.util.logging.ConsoleHandler to any of the loggers because this assignment can cause an infinite loop as mentioned in the Java logging topic.

The `logging.properties` file is located in the `<<WAS_install>>`/java/J*/lib/logging.properties directory, and can be customized as needed.

The default setting for this property is false.

jaxws.ignore.extraWSDLOps

Use this property if there are more operations in the WSDL than built into the client.

Default client behavior is to validate the operations built into the client against the operations in WSDL and fail if they do not match. Set this property to true if there are more operations in the WSDL than built into the client and the WSDL validation will succeed and the client can be invoked.

The default value is false.

jaxws.payload.highFidelity

Use this property to enable lossless transformations. When this property is set to `true`, the Web Service runtime guarantees that the incoming message and the message at the SOAPHandler boundary are the same.

Typically, the SOAP message received by a JAX-WS SOAPHandler is not exactly the same as the inbound SOAP message. For example, the message received by the JAX-WS SOAPHandler might use different xml prefixes than the original inbound message. These subtle changes do not affect the logical interpretation of the message. However, you must add this property to your JVM settings and set the property to true if messages at the SOAPHandler boundary must be exactly the same as the incoming messages. For example, the Canonicalization Specification (C14N) requires that the prefix names are preserved.

bprac: You should only use this property if your SOAP requests access the contents of a soapenv:Body element within your SOAPHandlers. Setting the property to `true` might degrade Web Service runtime performance.

The default value for this property is `false`.

jaxws.provider.interpretNullAsOneway

If you have a JAX-WS web service that defines a Provider-based endpoint using the `javax.xml.ws.Provider` annotation and a WSDL file is not specified, you can use this custom property to control how the JAX-WS runtime environment behaves when the Provider returns a null value from the `invoke()` method. By default, the runtime environment will send back a response that consists of a SOAPEnvelope that contains an empty SOAPBody element.

If this property is set to `true`, whenever the Provider implementation returns a null value and a WSDL file is not defined, the runtime environment interprets the null value returned from the Provider implementation as a request-only operation so that no response is returned. As with all request-only operations, some qualities of services, such as WS-Transactions, will not be available.

If the `javax.xml.ws.WebServiceProvider` annotation specifies a WSDL value and the WSDL defines a request and response operation, the JAX-WS runtime environment always returns a response that consists of a SOAPEnvelope that contains an empty SOAPBody, regardless of the setting of this property.

The default value for this custom property is `false`.

You must set this property to `true` for each server that requires a change in the default value.

jaxws.runtime.restrictStaticWebmethod

Use this property to prevent exposure of static operations. When this property is set to `true`, the JAX-WS runtime prevents the exposure of static operations.

The default value is false.

jaxws.soapfault.local.exceptions.disable

Note: Use the `jaxws.soapfault.local.exceptions.disable` property to prevent locally occurring exceptions on a JAX-WS client from being treated as a `SOAPFault`. By default, if a JAX-WS client encounters a local exception, a `SOAPFault` is created for the exception. An example of a local exception is a `ConnectException` caused by an invalid host or port. The relevant JAX-WS application handlers `handleFault` methods are called with the `SOAPFault`, then a `SOAPFaultException` is thrown back through the JAX-WS client's invoked method.

By setting this property to `true`, local exceptions create an empty message. The relevant JAX-WS application handlers `handleMessage` methods are called with the empty message, then a `WebServiceException` is thrown back through the JAX-WS client's invoked method. This was the behavior in previous releases.

The default value for this property is `false`.

ODCClearMessageAge

Use this property to establish a length of time, specified in milliseconds, after which an ODC message is removed from the bulletin board, even if the receiver has not acknowledged the message. Specifying a value for this property helps prevent the build up of messages that, for some reason, do not get acknowledged.

You can specify any positive integer as a value for this property, but a value of 300000 (5 minutes) or higher is recommended to avoid premature removal of messages.

The default value is 300000 milliseconds.

ODCInit.disabled

Set this property to `true` if you want to disable the communication between processes for the On Demand Configuration (ODC) component, and for all local ODC processing.

The on demand configuration component is used when deploying Web services-based applications, and when using a WebSphere Application Server Proxy Server to handle requests. The on demand configuration component is enabled or disabled on a cell-wide basis. Therefore, if your topology contains any proxy servers, or any web services based applications, you should not disable the on demand configuration service.

If you are running in a large topology environment where Web services-based applications are not deployed, or WebSphere Application Server Proxy Servers are not used to handle requests, the on demand configuration component is not utilized, and you can set this property to `true`. Setting this property to `true` disables the on demand configuration component, which will reduce network bandwidth and CPU utilization.

The default value is false.

org.eclipse.jst.j2ee.commonarchivecore.disableZip

Use this custom property to allow ZIP archives to be processed as simple files.

Set this property to true to allow ZIP archives to be processed as simple files when scanning the files of a deployed application.

The default value is false.

This property must be set as a custom property for the IBM WebSphere Application Server process which runs applications for which ZIP files are to be ignored.

org.eclipse.jst.j2ee.commonarchivecore.FILTERBINARIES

Use this custom property to allow certain application files from being listed during runtime processing.

Because of new JavaEE5 annotations processing requirements, more application files are typically listed during runtime processing than are listed in previous versions of the product. The additional listing might cause applications that are migrated from previous versions of the product to start more slowly, as additional time is spent listing application files.

The default value is not set.

If you decide to use this custom property, you must specify it as an application server JVM custom property.

sizeThreshold

Use this property when you want to control the algorithm for caching attachments in the JAX-WS runtime environment. When SOAP messages are processed by the JAX-WS runtime environment, the runtime environment stores small attachments in memory and stores large attachments in a file on a disk.

Use this property to specify, in kilobytes, the maximum size of an attachment that can be written to memory. If you do not specify a value for this property, the default value is 32. This default value specifies that any attachment that is less than 32 KB is stored in memory.

If the value of this property is increased, larger attachments are stored in memory. Increased values might increase the performance of the web service; however, this increased value causes the Java heap to grow. Setting the value too high might cause OutOfMemoryError errors to occur.

When determining a value for this property, remember that the larger the maximum cache size, the greater the impact on performance, and, potentially, to the Java heap. Use this property only for Java and performance tuning.

This property does not affect the logical processing of JAX-WS web services. Your JAX-WS web services will successfully process SOAP messages containing both large and small attachments, regardless of the setting of this property.

threadpool.maxsize

The JVM custom property, **threadpool.maxsize**, is a dedicated string used to start each application server. **threadpool.maxsize** controls the number of application servers that are started in parallel. You add **threadpool.maxsize** on the Node Agent to provide thread pool size as follows:

System Administration > Node agent > *nodeagent_ name* > Configuration > Java and Process Management > Process Definition > Java Virtual Machine > Custom Properties > New. Then you supply the name, **threadpool.maxsize** and a value.

If the value you supply is greater than 0, then a threadpool is created with that value as its maximum value. For example, if the value you supply is 3, a maximum of 3 application servers will be started in parallel. If the value you supply is 0 or less than 0, then the Node Agent behaves by launching application servers sequentially.

There is no default value. If you choose to use **threadpool.maxsize**, you must supply a value.

Note: When using the administrative console, you cannot leave the **threadpool.maxsize** value field blank. If you pass a blank value to the property from the command line then it will be considered as an illegal value, and the default behavior of the Node Agent is restored.

There is no maximum value for **threadpool.maxsize**. The maximum threads created will be equal to the number Application Servers. You cannot configure this value.

was.xcfmonitor.enabled

Use this property to enable the use of the z/OS Cross-system Coupling Facility (XCF) for monitoring application server status. When this property is set to `true`, node agents can use XCF to monitor their application servers, application servers can use XCF to monitor their node agents, and an administrative agent server can use XCF to monitor its registered application servers. When this property is set to `false`, the TCP/IP pinging method is used to monitor application server and node agent status.

The default value is true.

webservices.unify.faults

Note: Use the `webservices.unify.faults` property to disable SOAP Fault unification for JAX-WS and JAX-RPC. By default, the web service runtime environments (both JAX-WS and JAX-RPC) unify all faults generated by the runtime environment to a single type of fault containing a `faultcode` of `Server` and a `faultstring` of `Internal Error`. The faults do not contain any additional information identifying the actual cause of the fault. The unification of faults results in a more secure system, preventing detailed information regarding why inbound message processing failed from being returned to message senders.

The default value for this property is `true`, which causes faults to be unified. If your applications require fault details, then you can set this property to `false` to disable fault unification, allowing detailed information to be returned in faults. Note that regardless of the property setting, checked exceptions defined in the WSDL and thrown by a service provider method implementation are not unified. Additionally, detailed information regarding the cause of the fault are logged if trace is enabled, regardless of the setting of this property.

This property and the associated behavior is new in Version 8 of the product.

wink.client.readTimeout

Use this property to specify how long the RestClient object waits (in milliseconds) for a response to requests before timing out. A value of zero (0) means that the client waits for an unlimited amount of time and will not timeout.

The default value is `60,000` milliseconds.

wink.client.connectTimeout

Use this property to specify how long the RestClient object waits (in milliseconds) before timing out when attempting to connect to the target resource. A value of zero (0) means that the client waits for an unlimited amount of time and will not timeout.

The default value is `60,000` milliseconds.

Tuning application servers

The product contains interrelated components that must be harmoniously tuned to support the custom needs of your end-to-end e-business application.

About this task

The following steps describe various tuning tasks that may improve your application server performance. You can choose to implement any of these application server settings. These steps can be performed in any order.

Procedure

1. Run the applyPerfTuningTemplate.py, as the starting point for improving the performance of an application server.

   You can use the python-based tuning script, applyPerfTuningTemplate.py, along with one of its template files, to apply recommended performance tuning settings. The script, and these template files are located in the <*WAS_HOME*>/scriptLibraries/perfTuning/V70 directory.

2. **Tune the object request broker.** An Object Request Broker (ORB) manages the interaction between clients and servers, using the Internet InterORB Protocol (IIOP). It supports client requests and responses received from servers in a network-distributed environment. You can use the following parameters to tune the ORB:
   - Set **Pass by reference (com.ibm.CORBA.iiop.noLocalCopies)** as described in the *Tuning guide* PDF.
   - Set the **com.ibm.CORBA.FragmentSize** as described in the information about Object Request Broker custom properties.*Administering applications and their environment* PDF.

3. **Tune the XML parser definitions.**
   - **Description:** Facilitates server startup by adding XML parser definitions to the jaxp.properties and xerces.properties files in the ${*app_server_root*}/jre/lib directory. The XMLParserConfiguration value might change as new versions of Xerces are provided.
   - **How to view or set:** Insert the following lines in both files:

   ```
   javax.xml.parsers.SAXParserFactory=org.apache.xerces.jaxp.SAXParserFactoryImpl
   javax.xml.parsers.DocumentBuildFactory=org.apache.xerces.jaxp.
              DocumentBuilderFactoryImpl
   org.apache.xerces.xni.parser.XMLParserConfiguration=org.apache.xerces.parsers.XIncludeAwareParserConfiguration
   ```

   You can also consult with the jre/lib/jaxp.properties and jre/lib/xerces.properties files that come with the JDK installation. These sample files always contain the recommended settings.
   - **Default value:** None
   - **Recommended value:** None

4. **Tune the dynamic cache service.**

   Using the dynamic cache service can improve performance. See the *Administering applications and their environment* PDF for information about using the dynamic cache service and how it can affect your application server performance.

5. **Tune the EJB container.** An Enterprise JavaBeans (EJB) container is automatically created when you create an application server. After the EJB container is deployed, you can use the following parameters to make adjustments that improve performance.
   - Set the **Cleanup interval** and the **Cache size** as described in the *Administering applications and their environment* PDF.
   - **Break CMP enterprise beans into several enterprise bean modules** while assembling EJB modules.

   See also the *Tuning guide* PDF.

6. **Tune the session management.**

   The installed default settings for session management are optimal for performance. See the *Tuning guide* PDF for more information about tuning session management.

7. **Tune the data sources and associated connection pools.** A data source is used to access data from the database; it is associated with a pool of connections to that database.

8. **Tune the URL invocation cache.**

Each JavaServer Page is a unique URL. If you have more than 50 unique URLs that are actively being used, increase the value specified for the invocationCacheSize JVM custom property. This property controls the size of the URL invocation cache.

Each JavaServer Page is a unique URL. If you have more than 50 unique URLs that are actively being used, increase the value specified for the invocationCacheSize JVM custom property. This property controls the size of the URL invocation cache. See the *Administering applications and their environment* PDF for more information on how to change this property.

9. **Change how frequently the recovery log service attempts to compress any logstreams that application components are using.**

The Transaction Service RLS_LOGSTREAM_COMPRESS_INTERVAL custom property can be set to a value larger then the default value if the Transaction Service is the only application component using a logstream. If none of your components are configured to use a logstream, you can set this property to 0 (zero) to disable this function.

Tuning the application server using pre-defined tuning templates

You can use the python-based tuning script, applyPerfTuningTemplate.py, along with one of its template files, to apply pre-defined performance tuning templates to your application server or cluster. The script, and these property-based template files are located in the <*WAS_HOME*>/scriptLibraries/perfTuning/ V70 directory.

Before you begin

bprac: The configuration settings applied by this script and the associated tuning templates should be viewed as potential performance tuning options for you to explore or use as a starting point for additional tuning. The configuration settings that each of the pre-defined templates applies are geared towards optimizing common application server environments or scenarios. Typically, these settings improve performance for many applications.

Because optimizing for performance often involves trade-offs with features, capabilities, or functional behavior, some of these settings might impact application correctness, while other settings might be inappropriate for your environment. Please review the documentation below and consider the impact of these settings to your application inventory and infrastructure.

As with any performance tuning exercise, the settings configured by the predefined templates should be evaluated in a controlled preproduction test environment. You can then create a customized template to refine the tuning settings to meet the specific needs of your applications and production environment.

Note: This topic references one or more of the application server log files. Beginning in WebSphere Application Server Version 8.0 you can configure the server to use the High Performance Extensible Logging (HPEL) log and trace infrastructure instead of using `SystemOut.log` , `SystemErr.log`, `trace.log`, and `activity.log` files or native z/OS logging facilities. If you are using HPEL, you can access all of your log and trace information using the LogViewer command-line tool from your server profile bin directory. See the information about using HPEL to troubleshoot applications for more information on using HPEL.

Typically, when you run the applyPerfTuningTemplate.py script, you will specify either the production.props template file or the development.props template file to apply against the target server or cluster.

- If you specify the production.props template file when you run the applyPerfTuningTemplate.py script, the script applies configuration settings that are appropriate for a production environment where application changes are rare and optimal runtime performance is important.
- If you specify the development.props template file when you run the applyPerfTuningTemplate.py script, the script applies configuration settings that are appropriate for a development environment where frequent application updates are performed and system resources are at a minimum.

In addition to these two common templates, a third template file, default.props, is provided to enable you to revert the server configuration settings back to the out-of-the-box defaults settings.

You can also create your own custom tuning template. To create a custom tuning template, copy one of the existing templates, modify the configuration settings to better fit the needs of your applications and environment, and then use the applyPerfTuningTemplate.py script to apply these customized settings. The script and properties files leverage the property file configuration management features that wsadmin provides, and can easily be augmented to tune additional server components. See the topic Using properties files to manage system configuration for more information.

About this task

Review the following table to see the configuration changes that occur based on the template file that you specify when you run the applyPerfTuningTemplate.py script. A blank cell in this table indicates that the listed parameter is not configured, or is configured back to the default settings for the server defaults.

Table 57. Tuning parameters and their template values.   The table includes the tuning parameter and its value for the default template, the production template and the development template.

Parameter	Server default (default.props template file)	Production environment (production.props template file)	Development environment (development.props template file)
JVM Heap Size (MB) See the topic Tuning the IBM virtual machine for Java for more information about this setting.	50 min / 256 max	512 min / 512 max	256 min / 512 max
Verbose GC See the topic Tuning the IBM virtual machine for Java for more information about this setting.	disabled	enabled	enabled
JVM Diagnostic Trace (Generic JVM Arguments) See the topic Tuning the IBM virtual machine for Java for more information about this setting.		-Dcom.ibm.xml.xlxp.jaxb .opti.level=3	-Dcom.ibm.xml.xlxp.jaxb .opti.level=3
HTTP (9080) and HTTPS (9443) Channel maxKeepAliveRequests See the topic HTTP transport custom properties for more information about this setting.	100	10000	10000
TCP Channel maxOpenConnections	20000	500	500
TCP Channel listenBacklog	511	128	128
Development Mode See the topic Application server settings for more information about this setting.	disabled		enabled

Table 57. Tuning parameters and their template values (continued).   The table includes the tuning parameter and its value for the default template, the production template and the development template.

Parameter	Server default (default.props template file)	Production environment (production.props template file)	Development environment (development.props template file)
Server Component Provisioning See the topic Application server settings for more information about this setting.	disabled	enabled	enabled
PMI Statistic Set See the topic Enabling PMI data collection for more information about this setting.	basic	none	none
Authentication Cache Timeout See the topic Authentication cache settings for more information about this setting.	10 minutes	60 minutes	60 minutes
Data Source Connection Pool Size* See the topic Connection pool settings for more information about this setting.	1 min / 10 max	10 min / 50 max	
Data Source Prepared Statement Cache Size* See the topic WebSphere Application Server data source properties for more information about this setting.	10	50	
ORB Pass-by-Reference** See the topic Request Broker service settings for more information about this setting.	disabled	enabled	enabled
Web Server Plug-in ServerIOTimeout	900	900	900
Thread Pools (Web Container, ORB, Default) See the topic Thread pool settings for more information about this setting.	50 min / 50 max, 10 min / 50 max, 20 min / 20 max		5 min / 10 max

Table 57. Tuning parameters and their template values (continued).   The table includes the tuning parameter and its value for the default template, the production template and the development template.

Parameter	Server default (default.props template file)	Production environment (production.props template file)	Development environment (development.props template file)
Table notes:			
* Indicates items that are tuned only if they exist in the configuration. For example, a data source connection pool typically does not exist until an application is installed on the application server. If these items are created after your run the script, they are given the standard server default values unless you specify other settings.			
** Enabling ORB Pass-By-Reference can cause incorrect application behavior in some cases, because the Java EE standard assumes pass-by-value semantics. However, enabling this option can improve performance up to 50% or more if the EJB client and server are installed in the same instance, and your application is written to take advantage of these feature. The topic Object Request Broker service settings can help you determine if this setting is appropriate for your environment.			

Following are a few subtle platform-specific tuning differences:

z/OS platform
> The default JVM heap sizes are different than those on the other platforms:
> - Default minimum heap size: 256 MB
> - Default maximum heap size: 512 MB

Procedure
- Start the wsadmin tool if it is not already running, and then complete one of the following actions to tune an application server or all of the application servers in a cluster.
- Run the applyPerfTuningTemplate.py script to tune a specific server or cluster of servers running in a production environment.

  ```
  wsadmin -f applyPerfTuningTemplate.py
  [-nodeName node_name -serverName server_name][clusterName cluster_name] -templateFile production.props
  ```
- Run the applyPerfTuningTemplate.py script to tune a specific server or cluster of servers running in a development environment.

  ```
  wsadmin -f applyPerfTuningTemplate.py
  [-nodeName node_name -serverName server_name][clusterName cluster_name] -templateFile development.props
  ```
- Run the applyPerfTuningTemplate.py script to change the settings for a server or a cluster back to the standard out-of-the-box default configuration settings.

  ```
  wsadmin -f applyPerfTuningTemplate.py
  [-nodeName node_name -serverName server_name][clusterName cluster_name] -templateFile default.props
  ```

What to do next

Conduct a performance evaluation, and tuning exercise to determine if you should further fine tune the server for your specific applications.

Web services client to web container optimized communication

To improve performance, there is an optimized communication path between a web services client application and a web container that are located in the same application server process. Requests from the web services client that are normally sent to the web container using a network connection are delivered directly to the web container using an optimized local path. The local path is available because the web services client application and the web container are running in the same process.

This direct communication eliminates the need for clients and web containers that are in the same process to communicate over the network. For example, a web services client might be running in an application server. Instead of accessing the network to communicate with the web container, the web services client

can communicate with the web container using the optimized local path. This optimized local path improves the performance of the application server by enabling web services clients and web containers to communicate without using network transports.

In a clustered environment, there is typically an HTTP server (such as IBM HTTP server) that handles incoming client requests, distributing them to the correct application server in the cluster. The HTTP server uses information about the requested application and the defined virtual hosts to determine which application server receives the request. The web services client also uses the defined virtual host information to determine whether the request can be served by the local web container. You must define unique values for the host and port on each application server. You cannot define the values of host and port as wild cards denoted by the asterisk symbol (*) when you enable the optimized communication between the web services application and the web container. Using wild cards indicate that the local web container can handle web services requests for all destinations.

The optimized local communication path is disabled by default. You can enable the local communication path with the enableInProcessConnections custom property. Before configuring this custom property, make sure that you are not using wild cards for host names in your web container end points. Set this property to **true** in the web container to enabled the optimized local communication path. When disabled, the web services client and the web container communicate using network transports.

For information about how to configure the enableInProcessConnections custom property, see the *Administering applications and their environment* PDF.

When the optimized local communication path is enabled, logging of requests through the local path uses the same log attributes as the network channel chain for the web container. To use a different log file for in process requests than the log file for network requests, use a custom property on the HTTP Inbound Channel in the transport chain. Use the localLogFilenamePrefix custom property to specify a string that is added to the beginning of the network log file name to create a file name that is unique. Requests through the local process path are logged to this specified file. For example, if the log filename is `../httpaccess.log` for a network chain, and the localLogFilenamePrefix custom property is set to "local" on the HTTP channel in that transport chain, the local log file name for requests to the host associated with that chain is `/localhttpaccess.log`.

Important: If you specify a value for the localLogFilenamePrefix custom property, you must also set the accessLogFileName HTTP channel custom property to the fully qualified name of the log file you want to use for in process requests. You cannot specify a variable, such as $(SERVER_LOG_ROOT), as the value for this custom property.

Chapter 5. Balancing workloads

You should use server clusters and cluster members to monitor and manage the workloads of application servers.

Before you begin

You should understand your options for configuring application servers. To assist you in understanding how to configure and use clusters for workload management, consider this scenario. Client requests are distributed among the cluster members on a single machine. A *client* refers to any servlet, Java application, or other program or component that connects the end user and the application server that is being accessed.

In more complex workload management scenarios, you can distribute cluster members within the same sysplex.

About this task

Perform the following steps if you decide to use clusters to balance your workload.

Procedure

1. Decide which application server you want to cluster.
2. Decide whether you want to replicate data. Replication is a service that transfers data, objects, or events among application servers.

   You can create a replication domain when creating a cluster.
3. Deploy the application onto the application server.
4. Create a cluster.

   After configuring the application server and the application components exactly as you want them to be, create a cluster. The original server instance becomes a cluster member that is administered through the cluster.
5. Create one or more cluster members.
6. Start the cluster.

   When you start the cluster, all of the application servers that are members of that cluster start. Workload management automatically begins after the cluster members start.
7. After the cluster is running, you can perform the following tasks:
   - Stop the cluster.
   - Upgrade the applications that are installed on the cluster members.
   - Detect and handle problems with server clusters and their workloads.
   - Change how frequently the workload management state of the client refreshes.

     The default timeout value for the com.ibm.CORBA.RequestTimeout JVM property is 0, which means wait forever. This default value is not a good setting to have for failover situations. Therefore, if your application is experiencing problems with timeouts, or if you have configured your system for failover situations, use the -CCD option on the LaunchClient command to set an appropriate non-zero value for this property.

     If the workload management state of the client refreshes too soon or too late, change the interval setting of the JVM custom property com.ibm.websphere.wlm.unusable.interval.

Clusters and workload management

Clusters are sets of servers that are managed together and participate in workload management. Clusters enable enterprise applications to scale beyond the amount of throughput capable of being achieved with a single application server. Clusters also enable enterprise applications to be highly available because requests are automatically routed to the running servers in the event of a failure. The servers that are members of a cluster can be on different host machines. In contrast, servers that are part of the same node must be located on the same host machine. A cell can include no clusters, one cluster, or multiple clusters.

Servers that belong to a cluster are *members* of that cluster set and must all have identical application components deployed on them. Other than the applications configured to run on them, cluster members do not have to share any other configuration data. One cluster member might be running on a huge multi-processor enterprise server system, while another member of that same cluster might be running on a smaller system. The server configuration settings for each of these two cluster members are very different, except in the area of application components assigned to them. In that area of configuration, they are identical. This allows client work to be distributed across all the members of a cluster instead of all workload being handled by a single application server.

When you create a cluster, you make copies of an existing application server template. The template is most likely an application server that you have previously configured. You are offered the option of making that server a member of the cluster. However, it is recommended that you keep the server available only as a template, because the only way to remove a cluster member is to delete the application server. When you delete a cluster, you also delete any application servers that were members of that cluster. There is no way to preserve any member of a cluster. Keeping the original template intact allows you to reuse the template if you need to rebuild the configuration.

A *vertical cluster* has cluster members on the same node, or physical machine. A *horizontal cluster* has cluster members on multiple nodes across many machines in a cell. You can configure either type of cluster, or have a combination of vertical and horizontal clusters.

You can use the administrative console to specify a weight for a cluster member. The weight you assign to a cluster member should be based on its approximate, proportional ability to do work. The weight value specified for a specific member is only meaningful in the context of the weights you specify for the other members within a cluster. The weight values do not indicate absolute capability. If a cluster member is unavailable, the web server plug-in temporarily routes requests around that cluster member.

For example, if you have a cluster that consists of two members, assigning weights of 1 and 2 causes the first member to get approximately 1/3 of the workload and the second member to get approximately 2/3 of the workload. However, if you add a third member to the cluster, and assign the new member a weight of 1, approximately 1/4 of the workload now goes to the first member, approximately 1/2 of the workload goes to the second member, and approximately 1/4 of the workload goes to the third member. If the first cluster member becomes unavailable, the second member gets approximately 2/3 of the workload and third member gets approximately 1/3 of the workload.

The weight values only approximate your load balance objectives. There are other application dependencies, such as thread concurrency, local setting preferences, affinity, and resource availability that are also factors in determining where a specific request is sent. Therefore, do not use the exact pattern of requests to determine the weight assignment for specific cluster members.

Workload management for EJB containers can be performed by configuring the web container and EJB containers on separate application servers. Multiple application servers can be clustered with the EJB containers, enabling the distribution of enterprise bean requests between EJB containers on different application servers.

In this configuration, EJB client requests are routed to available EJB containers in a round robin fashion based on assigned server weights. The EJB clients can be servlets operating within a web container, stand-alone Java programs using RMI/IIOP, or other EJBs.

The server weighted round robin routing policy ensures a balanced routing distribution based on the set of server weights that have been assigned to the members of a cluster. For example, if all servers in the cluster have the same weight, the expected distribution for the cluster is that all servers receive the same number of requests. If the weights for the servers are not equal, the distribution mechanism sends more requests to the higher weight value servers than the lower weight value servers. The policy ensures the desired distribution, based on the weights assigned to the cluster members.

You can set up workload management to balance the tasks between different clusters.

You can choose to have requests sent to the node on which the client resides as the preferred routing. In this case, only cluster members on that node are chosen (using the round robin weight method). Cluster members on remote nodes are chosen only if a local server is not available.

Multiple servers that can service the same client request form the basis for failover support. If a server fails while processing a client request, the failed request can be rerouted to any of the remaining cluster members. Even if several servers fail, as long as at least one cluster member is running, client requests continue to be serviced.

Techniques for managing state

Multiple machine scaling techniques rely on using multiple copies of an application server; multiple consecutive requests from various clients can be serviced by different servers. If each client request is completely independent of every other client request, it does not matter if consecutive requests are processed on the same server. However, in practice, client requests are not independent. A client often makes a request, waits for the result, then makes one or more subsequent requests that depend on the results received from the earlier requests.

This sequence of operations on behalf of a client falls into two categories:

Stateless
A server processes requests based solely on information provided with each request and does not rely on information from earlier requests. The server does not need to maintain state information between requests.

Stateful
A server processes requests based on both the information provided with each request and information stored from earlier requests. The server needs to access and maintain state information generated during the processing of an earlier request.

For stateless interactions, it does not matter whether different requests are processed by different servers. However, for stateful interactions, the server that processes a request needs access to the state information necessary to service that request. Either the same server can process all requests that are associated with the same state information, or the state information can be shared by all servers that require it. In the latter case, accessing the shared state information from the same server minimizes the processing overhead associated with accessing the shared state information from multiple servers.

The load distribution facilities in the product use several different techniques for maintaining state information between client requests:

- Session affinity, where the load distribution facility recognizes the existence of a client session and attempts to direct all requests within that session to the same server.

- Transaction affinity, where the load distribution facility recognizes the existence of a transaction and attempts to direct all requests within the scope of that transaction to the same server.
- Server affinity, where the load distribution facility recognizes that although multiple servers might be acceptable for a given client request, a particular server is best suited for processing that request.

Workload management (WLM) for z/OS

Workload management optimizes the distribution of incoming work requests to the application servers, enterprise beans, servlets, and other objects that can most effectively process the requests. Workload management also provides failover when servers are not available, improving application availability.

For details on workload management, see *z/OS MVS Planning: Workload Management*, which is available on the z/OS Internet Library website. You might also find *z/OS MVS Programming: Workload Management Services* helpful.

When you are using workload management on z/OS, you can define workload management policies for your application servers. To get started, you do not need to define special classification rules and work qualifiers, but you might want to define them for your production system.

Workload management provides the following benefits to applications that are running on an application server:
- It balances server workloads, allowing processing tasks to be distributed according to the capacities of the different machines in the sysplex.
- It provides failover capability by redirecting client requests if one or more servers is unable to process them. This improves the availability of applications and administrative services.
- It enables systems to be scaled up to serve a higher client load than provided by the basic configuration. With clustering, additional instances of servers, servlets, and other objects can easily be added to the configuration.
- It enables servers to be transparently maintained and upgraded while applications remain available for users.
- It centralizes the administration of servers and other objects.

In the product environment, you implement workload management by using clusters, transports, and replication domains.

Connection optimization

Characteristics of a configuration in which the Domain Name Server cooperates with workload management (WLM) to route client requests throughout a cell are:

- The domain name server (DNS) is replicated by setting up a secondary DNS on more than one system in the cell.
- The client must know the host name and port of the name server to connect to WebSphere Application Server for z/OS.
- Each system in the cell has the same location service daemon IP name. Workload management and the domain name server determine the actual system that receives client requests. The client sees the cell as a single system, though its requests might be balanced across systems in the cell.
- As part of workload balancing and maximizing performance goals, workload management also routes work requests to systems in the cell. This function is possible because WebSphere Application Server for z/OS cooperates with workload management. Because the system references that a client sees are indirect, even requests from that same client might be answered by differing systems in the cell.
- The implication for clients is that they should not cache IP addresses unless they can recover from failed connections. That is, if a connection fails, a client should be able to reissue a request, but, because the IP address is an indirect address, a reissue of the request can be answered by another system in the cell.

For additional details on setting up servers for connection optimization, see *z/OS Communications Server: IP Configuration Reference*.

Sysplex routing of work requests

The product uses the domain name server (DNS) to route work requests within a cell. You can use the DNS, instead of a sysplex distributor to distribute workload and balance requests for the same hostname across multiple IP addresses (one per daemon).

The DNS accepts a generic hostname from the client and maps the name to a specific system. The DNS works with workload management (WLM) to select the best available system. Workload management analyzes the current state of the cell and considers a number of factors, such as CPU, memory, and I/O utilization, when it determines the best system to handle new work. The DNS then routes the client request to that system. This use of workload management and the DNS is optional. However, using workload management and the DNS eliminates a single point of failure.

Each system in a cell has the product runtime. All the systems contain a location service daemon, node agent, and business application servers. One system acts as the deployment manager for the cell. The client uses the CORBA General Inter-ORB Protocol (GIOP) to send requests to the product. The location service daemon acts as a location service agent. It accepts locate requests with object keys in the requests. The location service daemon extracts the server cluster name from the object key, and then gives the server name to workload management. Workload management chooses the optimal server in the cell to handle the request. The location service daemon merges specific Interoperable Object Reference (IOR) information that is related to the chosen server with object key information stored in the original IOR. The result of this merging is a direct IOR that gets returned to the client. The client ORB uses this returned reference to establish the IOR connection to the server holding the object of interest.

The transport mechanism that the product uses depends on whether the client is local or remote. A client that is not running on the same z/OS system as the application server, is called a remote client, and requires a TCP/IP transport. If the client is local, the transport is through a program call. Local transport is faster because it does not require a physical trip over the network, eliminates data transforms, simplifies the marshalling of requests, and uses optimized Resource Access Control Facility (RACF) facilities for security rather than having to invoke Kerberos or the Secure Sockets Layer (SSL).

Address space management for work requests

The product propagates the performance context of work requests by using workload management (WLM) enclaves. Each transaction has its own enclave and is managed according to its service class.

The controller of a server, which workload management views as a queue manager, uses the enclave associated with a client request to manage the priority of the work. If the work has a high priority, workload management can direct the work to a high-priority servant in the server. If the work has a low priority, workload management can direct the work to a low-priority servant. The effect is to partition the work according to priority within the same server.

Enclaves can originate in several ways:
- The product uses its own set of rules to create an enclave for a client request from the network.
- Some subsystems, such as the IBM HTTP Sever, create enclaves and pass them to the application server, which, in turn, passes the enclaves on.
- The product treats batch jobs as if they were remote clients.

To communicate the performance context to workload management, you must classify the workloads in your system according to the following work qualifiers.

Table 58. WLM work qualifiers and corresponding product entities.   The following table describes WLM work qualifiers and corresponding product entities.

Work qualifier abbreviation	Work qualifier	Corresponding product entity
CN	Collection name	Cluster name
UI	User ID	User ID under which work is running

For more information about classification rules and workload qualifiers, refer to the topic *Classifying z/OS workload* and the z/OS publication *z/OS MVS Planning: Workload Management*.

In addition to client workloads, you must consider the performance of the product run-time servers and your business application servers. In general, server controllers act as work routers, so they must have high priority. Because workload management starts and stops servants dynamically, servants also need high priority to be initialized quickly. After the servants are initialized, they run work according to the priority of the client enclave, so the servant priority that you assign has no significance after initialization.

In summary, use the following table to set the performance goals for each class:

Table 59. Workload management rules.   The following table describes workload management rules.

If you are classifying the ...	... assign it to:	Explanation
Location service daemon	SYSSTC or a high velocity, high importance STC	The system treats it as a started task, and it must route work requests quickly.
Controller	SYSSTC or a high velocity, high importance STC	A controller must route work quickly, but you must balance the priority of your business application server with other work in the system.
Servant	A lower velocity and importance STC than the controller	The servant should be assigned a lower goal than the controller because the servant is less important than the controller. **gotcha:** This goal only applies during servant startup. After a server starts processing requests, it is managed according to the goals of the work being processed in the servant.
Application environment	Use the CB classification rules, the percentage response time goal, for example 80% of transactions complete in .25 seconds.	
Client applications	Assuming a long-running application, a velocity goal should be used that is relative to other work on the system.	

WLM dynamic application environment operator commands

The dynamic application environments are displayed and controlled separately from static application environments. In order to control the dynamic environments, you must set the Resource Access Facility (RACF) server class profiles to give you the proper permission to issue MVS console commands.

If you have the proper permission, you can issue the following commands from the MVS console:

Display a specific dynamic application environment
>   `D WLM,DYNAPPL=appl_env_name` (appl_env_name is the short cluster name)

Display all dynamic application environments
```
D WLM,DYNAPPL=*
```

Restart a specific dynamic application environment
```
V WLM,DYNAPPL=appl_env_name,RESUME
```

Quiesce a specific dynamic application environment
```
V WLM,DYNAPPL=appl_env_name,QUIESCE
```

Setting up a highly available sysplex environment

Setting up a highly available sysplex environment enables you to control application rollout and workload routing.

Before you begin

- A highly available sysplex environment must include at least two logical partitions (LPARs). These LPARs should be on separate hardware instances to eliminate hardware single points of failure (SPOFs).
- There must be a network path redundancy leading up to the web servers and Applications Servers in your sysplex.
- If you are using HTTP sessions, session state must be shared between cluster member using the data replication service (DRS), or your session data must be stored in DB2. If you are using stateful session Enterprise JavaBeans (EJBs), the stateful session persistent store must be configured on a shared HFS. It is not recommended that you use stateful session Enterprise JavaBeans.

About this task

Complete the following actions if you want to set up a highly available sysplex environment.

Procedure

1. Configure a node on each LPAR that is configured in the Network Deployment cell. The deployment manager Server, which is required, must be configured on its own node. It can be configured on either LPAR or on a separate LPAR.
2. Use the administrative console to verify that a location service daemon has been defined on each LPAR that has one or more nodes in the same cell.
3. Define an application server on each node, and form all of the application servers into a cluster.

    See the topic *Adding members to a cluster* for more information on how to add application servers to a cluster.
4. Define the following dynamic virtual IP addresses (DVIPAs) through the z/OS Operating System's Sysplex Distributor.
    - A dynamic virtual IP address as the daemon's IP name for the cell. This IP address enables WLM-balanced workload routing and fail over between the LPARs for IIOP requests.
    - A dynamic virtual IP address as the HTTP transport channel name for the cell. This IP address enables WLM-balanced routing and fail over between the LPARs for sessionless HTTP requests.

    See the *z/OS Communications Server IP Configuration Guide* for your version of the z/OS operating system for a description of how to define IP addresses through the z/OS Sysplex Distributor. This publication is available at http://www.ibm.com/servers/eserver/zseries/zos/bkserv/v1r4books.html.
5. Define a static IP address for each node as an auxiliary HTTP transport channel name for the cell. This IP address enables directed HTTP routing for sessional HTTP requests.
6. Configure web server plug-ins in each of the web servers. Configure the plug-ins to use the HTTP DVIPA for sessionless requests and the static IP addresses for sessional requests.

Sysplex Distributor

The IBM-recommended implementation, if you are running in a sysplex, is to set up your TCP/IP network with Sysplex Distributor. This makes use of dynamic virtual IP addresses (DVIPAs), which increase availability and aid in workload balancing.

The following are recommended environment considerations for Sysplex Distributor:
- You need only basic sysplex functionality to utilize DVIPAs and Sysplex Distributor because these functions do not rely on data stored permanently in the coupling facility.
- Set up your system such that each HTTP request connection results in no saved state or the HTTP and application servers are configured to share a persistent state.

When doing this, HTTP server plug-ins send no-affinity connections to Sysplex Distributor (a secondary connection load balancer) with more information to make a better distribution decision.

Note: As long as the HTTP catcher itself is not bound to any particular IP address, the application-specific DVIPA can be used when affinities dictate a particular server. This allows use of the Sysplex Distributor server address for requests that are not tied to a server, covering the same set of servers in the sysplex.

Since the client connection terminates at the plug-in/proxy, and the secondary connection is established by the plug-in itself, there is no need for network address translation.

Requests to the node agent do not require any affinity, and each request is independent of other requests. Sysplex Distributor can be used to balance work requests among node agents, with the added benefit that Sysplex Distributor knows which nodes are available. Therefore, it will never route a work request to a node that is not listening for new connection requests.

Note: If you are running z/OS Version 1.2 or earlier, Sysplex Distributor is limited to distribution on only four ports for a particular distributed DVIPA. You may configure multiple DVIPAs when more than four ports exist, but this is a configuration burden.

Setting up the Server Runtime on multiple systems in a sysplex

If your applications require workload balance, and high availability, and you want the ability to easily add new systems to meet demand as your workload grows, you might want to migrate your Application Server runtime and associated application servers from a monoplex to a sysplex configuration.

Before you begin

After you install the server runtime and associated business application servers on a monoplex, you should evaluate whether you want to migrate the server runtime and associated application servers to a sysplex configuration. If you decide to set up a sysplex environment that consists of several z/OS images and workload management (WLM), you must run the BBOWWPFA job in the z/OS image on which you are starting WebSphere Application Server.

The BBOWWPFA job is one of the jobs that are automatically generated when you initially configure and customization WebSphere Application Server. You run this batch of jobs to set up your z/OS environment. Typically these JCL jobs are executed one by one in the first z/OS image that is available on the sysplex. However, the BBOWWPFA job, which sets up the runtime (configuration) file system for the product must run in the same z/OS image as where you plan to start the product. To ensure that the BBOWWPFA job runs in the proper image, add the following JCL statement, after the BBOWWPFA job statement, within the batch of automatically generated JCL statements, before you run the BBOWWPFA job.

```
/*JOBPARM SYSAFF=sxx
```

where *sxx* is the name of z/OS image in which you are going to run the product.

About this task

A sysplex environment enables you to:
*   Balance the workload across multiple systems, thus providing better performance management for your applications.
*   Add new systems to meet demand as your workload grows. This capability provides a scalable solution to your processing needs.
*   Replicate the runtime and associated business application servers. This capability ensures that if a failure occurs on one system, other systems are available to handle user requests.
*   Upgrade the Application Server from one release or service level to another without interrupting service to your users.

gotcha: If you are using a global resource serialization (GRS) ring to attach one or more monoplexes to a sysplex environment, the cell name of any servers running in any of the monoplexes must be unique within the entire GRS environment. This requirement means that the cell name of a server running in any of the monoplexes:

*   Must be different than the cell name of any servers running in the sysplex
*   Must be different than the cell name of any servers running in another monoplex that is attached to the sysplex

If you have servers with duplicate cell names within the GRS environment, WebSphere Application Server cannot differentiate between the sysplex cell and the monoplex cell, and treats both servers as part of the same cell, This inaccurate cell association typically causes unpredictable processing results.

Perform the following tasks to setup the Application Server in a sysplex configuration.

Procedure

1. Set up a sysplex environment if one is not already available.

    The z/OS publication *z/OS MVS Setting Up a Sysplex* describes how to set up a z/OS sysplex. The directory you set up should be similar to the following directory structure:

Figure 7. Directory structure for two Application Servers running in a Sysplex

2. Configure the Server runtime for a sysplex environment.
    a. Decide whether you want a single-system view of the error log. If you want a single-system view of the error log, and initially you set up the error log in the system logger and used DASD for logging, you must now configure the error log in the coupling facility.
    b. Decide how you will share application executables in the cell.
    c. Set up ARM. This release does not support cross-system restart, so you must set up your ARM policy accordingly. Make sure you specify TARGET_SYSTEM for the system on which each element runs (if you take the default TARGET_SYSTEM=*, you get cross-system restart).
    d. Decide whether you will run all of the run-time servers on every system in the cell.

        Recommendations: The following table provides recommendations and requirements for running servers in a cell.

Table 60. Running servers in a cell.   The following table provides recommendations and requirements for running servers in a cell.

Server	Recommendations and requirements for running servers in a cell
location service daemon and node agent	• You must run both the location service daemon and the node agent on each system in the sysplex in which you want the server runtime to run. If the server runtime is not installed on some of the systems in your sysplex, you do not have to run a location service daemon and node agent on those systems. • If a server indicates that PassTickets are desirable for interaction with a client, you must run the location service daemon and node agent on the system where the z/OS client resides.
deployment manager	Make sure you follow the correct steps to configure a deployment manager cell.

3. Prepare your security system.

4. Set up data sharing. Refer to the *DB2 Data Sharing: Planning and Administration* publication for the version of DB2 that is running on your z/OS system.

5. Customize z/OS functions on the other systems in the sysplex to match the customization you performed as part of the initial server runtime installation.

   Complete customization information for additional systems is contained in the generated customization instructions.

6. Change the TCP/IP settings. Each system in a sysplex contains a location service daemon, node agent, and business application servers. The location service daemon acts as a location service agent and accepts locate requests with object keys in the requests. Therefore it is important that the TCP/IP domain name server (DNS) entries, and TCP/IP profile for each system in the cell include the port for the location service daemon, and that port is associated with the name of the new location service daemon server.

   a. Change DNS entries.

      If you use a DNS implementation that allows the use of generic IP names that dynamically resolve to like-configured servers, you must adjust the IP names in your DNS. You must keep the generic IP name of the location service daemon, but add a new IP name for the second and subsequent location service daemon servers. The additional IP names enable the DNS to direct work to other servers if a failure occurs.

   b. In the TCP/IP profile for each additional system in the cell, add a port for the location service daemon, and associate that port with a new location service daemon server name.

      By default, the server runtime uses port 5655 for the location service daemon. The server runtime also names the first location service daemon server DAEMON01 and increments the suffix on that name for each new location service daemon server; for example, DAEMON02, DAEMON03, and so forth. Therefore, for the second system in the sysplex, you must add a port and associate it with DAEMON02.

      Example:

      ```
      5655
          TCP     DAEMON02
      ```

      Follow the same pattern for the third and subsequent systems in the sysplexl.

7. Define new application server clusters in the sysplex.

8. Optional: Create deployment manager cells.

   a. Install the default application server on each node in your sysplex.

   b. Install a deployment manager cell on one node in your sysplex.

   c. Add default server nodes to the deployment manager cell.

Results

You can take advantage of all of the benefits of running your applications on multiple systems in a sysplex.

What to do next

Migrate your applications to the sysplex.

Enabling multiple servants on z/OS

Use this task to enable multiple servants on your system. Enabling multiple servants improves the performance of the product on z/OS..

Before you begin

Review the limitations of enabling multiple servants.

About this task

Workload Management (WLM), enables you to manage the performance and the number of servants that are running.. WLM manages the response time and throughput of transactions according to their assigned service class, the associated performance objectives, and the availability of system resources. To meet these goals, WLM sometimes needs to control or override the number of servants that are active. With this task, you can enable WLM to start additional servants to meet performance goals.

Procedure

1. In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*.
2. Under Server Infrastructure, click **Java and process management > Server instance**.
3. Select the Multiple instances enabled field.
4. Click **Apply** to finish the Server Instance changes.
5. Click **OK**, and then click **Save** to save your configuration changes.

Results

Workload Management (WLM) can start additional servant regions to meet performance goals, based on the importance of its work compared to other work in the system, the availability of system resources that are needed to satisfy those objectives, and a determination by WLM of whether starting more address spaces might help achieve the objectives. It is also important to make the goals reasonable.

Multiple servants

With workload management (WLM), you can control the number of servants that are running and how they are performing. WLM manages the response time and throughput of transactions according to their assigned service class, the associated performance objectives, and the availability of system resources. To meet these goals, WLM sometimes needs to control or override the number of servants that are active.

Product applications are deployed within a generic *server*. One or more server instances must be defined on one or more systems within a *node*. Each server instance consists of a *controller* and one or more *servants*. The controllers are started by MVS as *started tasks*, and servants are started by WLM, as they are needed.

WLM dynamic application environments can be enabled on your z/OS system if you are running on z/OS Release 2, with the service update for APAR OW54622 installed, or if you are running on a higher level of z/OS. If you have WLM dynamic application environments enabled on your system, WLM honors the specifications for the number of servants. If you are using static application environments, which is a setting that is specified through the WLM ISPF panels, then you must also enable multiple servants by indicating No limit in the WLM ISPF panels.

gotcha: If you use multiple servants, remember that:
- The administrative console and the deployment manager no longer have serialization requirements that only work in a single Java virtual machine (JVM). Therefore, you can run these applications in a server with multiple servants.
- If you specify a maximum number of instances, workload management (WLM) cannot start more than the specified number of servants for this server instance.
- The maximum number of servants should be at least as large as the number of different service classes that might be used by transactions that are run in the server. The number must also account for the *default CB-type service class* and enclaves that might originate outside of the product servers and are classified by other classification rules, such as the IBM HTTP Server (IHS).

Controlling the minimum and maximum number of servants

You can control the minimum or maximum number of servants for a server using the administrative console. The minimum value is useful for starting up a basic number of servants before your work arrives. This can reduce delays while waiting for the workload manager (WLM) to start up additional servants.

About this task

The maximum value is useful for *capping* the number of address spaces that are started by WLM for each *server instance*, if you determine that excessive servant regions are contributing to service degradation.

Procedure

1. Start the administrative console.
2. Click **Servers > Server Types > WebSphere application servers >** *server_name*.
3. Under Server Infrastructure, click **Java and process management > Server instance**
4. Type a value into the **Minimum Number of Instances** and **Maximum Number of Instances** fields, or leave these fields blank to allow WLM to determine the numbers.
5. Click **Apply** to finish the Server Instance changes.
6. Click **Save** to save your changes.

Classifying z/OS workload

You can use a common workload classification document to classify inbound HTTP, IIOP, Session Initiation Protocol (SIP), optimized local adapter, and message-driven bean (MDB) work requests for the z/OS workload manager.

Before you begin

You should use workload management on a z/OS system. See "Workload management (WLM) for z/OS" on page 502 for more information.

About this task

A workload classification document file is an XML file in which you classify incoming HTTP, IIOP, Session Initiation Protocol (SIP), optimized local adapter, and message-driven bean (MDB) work requests and assign them to a transaction class (TCLASS). The TCLASS value, if it is assigned, is passed to the MVS Workload Manager. WLM uses the TCLASS value to classify the inbound work requests and to assign a service class or a report service class to each request.

The common workload classification document is the method you should use to classify work requests in a z/OS environment. Support for other WebSphere Application Server mechanisms for classifying work in a z/OS environment is deprecated and you should no longer use those mechanisms.

If you want to classify work for message-driven beans deployed against JCA 1.5 resources with the default messaging provider, or you want to classify mediation work for use with service integration buses, you need to define a Classification element that uses SibClassification elements. You must also perform z/OS Workload Manager actions that are required to use the TCLASS value "SIBUS". If you replace any listener port with a JMS activation specification for use by MDB applications with the Version 6 default messaging provider, you should replace any related InboundClassification type="mdb" classifications with SibClassifications type="jmsra" classifications.

If you want to classify work for message-driven beans deployed against a WebSphere MQ messaging provider activation specification, you need to define a Classification element that uses WMQRAClassification elements. You must also perform z/OS Workload Manager actions that are required to use the TCLASS value "WMQRA". If you replace any listener port with a JMS activation specification for use by MDB applications with the WebSphere MQ messaging provider, you should replace any related InboundClassification type="mdb" classifications with WMQRAClassification classifications.

Procedure

1. Develop the workload classification document. Use the information in the "Workload classification file" on page 514 topic to create the document. The topic contains examples of the workload classification document, with and without RAS attributes. Use one workload classification document whether you are using it to classify z/OS workload or to implement Reliability Availability and Serviceability (RAS) granularity.

2. If you create the document on a z/OS system in code page IBM-1047, the normal code page for files that exist in the HFS, convert the file to ASCII before you use the file. Use one of the following options to convert a working document into a document that can be used by the server:

   - native2ascii

     This is a utility in the Java SDK that can convert a file from the native code page to the ASCII code page. For example, if you are working on an XML document called x5sr02.classification.ebcdic.xml and you want to create a document called x5sr02.classification.xml, use the following command:

     ```
     /u/userid -> native2ascii \
     x5sr02.classification.ebcdic.xml > x5sr02.classification.xml
     ```

     The command line is split with the backslash (\) character to the next line for publication purposes.

   - iconv

     This is a z/OS utility that can convert files from one designated code page to a different designated code page. For example, if you are working on an XML document called x5sr02.classification.ebcdic.xml and you want to create a document called x5sr02.classification.xml, use the following command:

     ```
     /u/userid -> iconv -f IBM-1047 -t UTF-8 \
     x5sr02.classification.ebcdic.xml >x5sr02.classification.xml
     ```

     The command line is split with the backslash (\) character to the next line for publication purposes.

   - Create the document on your workstation and then FTP the file to the correct location on the z/OS system in binary format. By using this option, you can also create the `Classification.dtd` file in the same directory as the workload classification document. Then, you can perform an XML validity check on the document before installing it on a server. Use any type of validating parser, for example, you can use WebSphere Application Developer workbench to construct and validate the workload classification document.

3. Specify the location of the workload classification document in the administrative console. Use the `wlm_classification_file` variable to specify the XML file that contains the classification information. In the administrative console, click **Environment > WebSphere variables > New**. You can set the variable at cell, node, or server instance level. If you specify the variable at the cell or node level, the information must be accessible and applicable to all the servers that inherit the specification from the node or cell.

4. Perform z/OS Workload Manager actions that are required to use the TCLASS values. Each TCLASS must be assigned a service class, report service class, or both to the enclave under which the work runs. The `CB` classification rules must be updated.

If you want to classify work for message-driven beans deployed against JCA 1.5 resources with the default messaging provider, or you want to classify mediation work for use with service integration buses, you need to perform z/OS Workload Manager actions that are required to use the TCLASS value "SIBUS".

Transaction classes are used as sub-rules in establishing service classes and transaction. The TCLASS values are not used as level one rules. If you decide to use TCLASS as a level one rule rather than a sub-rule, you must be careful in ordering the rules. The first level one rule that applies to the work is used, so more specific rules should be first, followed by the broad rules. For example, consider the following two examples of CB classification rules:

```
Subsystem-Type Xref Notes Options Help
--------------------------------------------------------------------------
Modify Rules for the Subsystem Type     Row 1 to 17 of 17
Command ===> _____  SCROLL ===> CSR
Subsystem Type . : CB  Fold qualifier names? Y (Y or N)
Description . . . CB Class'n w/WLM Trans. CLASSes
Action codes: A=After C=Copy M=Move I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
More ===>
        --------Qualifier--------              -------Class--------
Action   Type   Name   Start                   Service   Report
                DEFAULTS:  CBCLASS RWASDEF
____ 1   CN    P5SR01*  1                      CBCLASS RTP5CLUS
____ 1   TC     A0      ___                     CBHUTCH RP5A0
____ 1   TC     A1      ___                     CBHUTCH RP5A1
____ 1   TC     A1B     ___                     CBHUTCH RP5A1B
____ 1   CN    WSIVP2*  ___                     CBSLOW  RWSIVP
____ 1   CN    T%SERV*  1                       CBFAST  RTSMIGT
____ 1   CN     B4*     ___                     CBFAST  _____
```

In the preceding example, the TCLASS assignments that are made for enclaves running in the server P5SR01x are never used by the workload manager. When the following rule is run, no further searching of the classification table is done:

```
____ 1   CN     P5SR01*  1               CBCLASS
```

The TCLASS assignments are not used. All of the enclaves that run in the P5SR01x servers are assigned to the CBCLASS service class and the RTP5CLUS report service class.

```
Subsystem-Type Xref Notes Options Help
--------------------------------------------------------------------------
Modify Rules for the Subsystem Type     Row 1 to 17 of 17
Command ===> _____  SCROLL ===> CSR
Subsystem Type . : CB  Fold qualifier names? Y (Y or N)
Description . . . CB Class'n w/WLM Trans. CLASSes
Action codes: A=After C=Copy M=Move I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
More ===>
        --------Qualifier--------              -------Class--------
Action   Type   Name   Start                   Service      Report
                            DEFAULTS: CBCLASS     RWASDEF
____ 1   TC     A0      ___                     CBHUTCH      RP5A0
____ 1   TC     A1      ___                     CBHUTCH      RP5A1
____ 1   TC     A1B     ___                     CBHUTCH      RP5A1B
____ 1   CN    P5SR01*  1                       CBCLASS      RTP5CLUS
____ 1   CN    WSIVP2*  ___                     CBSLOW       RWSIVP
____ 1   CN    T%SERV*  1                       CBFAST       RTSMIGT
____ 1   CN     B4*     ___                     CBFAST       _____
```

In the preceding example, if a TCLASS value of A0, A1, or A1B are provided in the classification, they are used regardless of which server is running the work. In this case, the server name is used only if these three TCLASS values are not present.

5. Implement the changes to the file. You can restart the application server or reload the workload classification document without having to restart the server:

   - Restart the application server.
   - Reload the workload classification document by issuing the following command:

     ```
     MODIFY|F <servername>, RECLASSIFY,FILE='/path/to/newfile.xml'
     ```

   If the workload classification document is not a well formed, valid XML document it is ignored by the application server and the following message is displayed:

   ```
   BBOJ0085E PROBLEMS ENCOUNTERED PARSING WLM CLASSIFICATION XML FILE (0)
   ```

6. Use the DISPLAY WORK operator command to display classification information. Use this command to determine if your classification scheme is classifying the work as you intended. Issue the following command to display the IIOP, HTTP, internal, SIP, MDB, and optimized local adapter classification information:

   ```
   MODIFY|F <servername>, DISPLAY,WORK,CLINFO
   ```

   Issue this command against each application server.

   The following example shows a possible result of issuing the new operator command:

   ```
   00- SY1  f bbos001,display,work,clinfo
        SY1  BBOJ0129I: The /tmp/wlm4.class.xml workload classification file was loaded at
        2009/07/14 19:33:35.297 (GMT).
        SY1  BB000281I CLASSIFICATION COUNTERS FOR IIOP WORK
        SY1  BB000282I CHECKED 0, MATCHED 0, USED 0, COST 2, DESC: IIOP root
        SY1  BB000282I CHECKED 0, MATCHED 0, USED 0, COST 4, DESC: leotag
        SY1  BB000282I CHECKED 0, MATCHED 0, USED 0, COST 3, DESC: byetag
        SY1  BB000282I CHECKED 0, MATCHED 0, USED 0, COST 4, DESC: hellotag
        SY1  BB000283I FOR IIOP WORK: TOTAL CLASSIFIED 0, WEIGHTED TOTAL COST 0
        SY1  BB000281I CLASSIFICATION COUNTERS FOR HTTP WORK
        SY1  BB000282I CHECKED 2, MATCHED 2, USED 0, COST 2, DESC: HTTP root
        SY1  BB000282I CHECKED 2, MATCHED 2, USED 0, COST 4, DESC: plantta4
        SY1  BB000282I CHECKED 2, MATCHED 1, USED 1, COST 3, DESC: giftag4
        SY1  BB000282I CHECKED 1, MATCHED 1, USED 1, COST 4, DESC: jpgtag4
        SY1  BB000283I FOR HTTP WORK: TOTAL CLASSIFIED 2, WEIGHTED TOTAL COST 7
        SY1  BB000188I END OF OUTPUT FOR COMMAND DISPLAY,WORK,CLINFO
   ```

   An explanation of the command output follows:

   BBOJ0129I: The *file* **workload classification file was loaded at** *time*.
   The message indicates the workload classification file currently active and the time that it was loaded.

   BB000281I CLASSIFICATION COUNTERS FOR *type* **WORK**
   The header message for messages that display the usage of the workload classification rules. The value of *type* can be HTTP, IIOP, INTERNAL, SIP, OLA, or MDB.

   BB000282I CHECKED *n1***, MATCHED** *n2***, USED** *n3***, COST** *n4***, DESC:** *text*
   This message displays information about a particular rule in the workload classification. This message displays the following information:

   - *n1* - The number of times the rule has been examined.
   - *n2* - The number of times that this rule has been matched by the request.
   - *n3* - The number of times that this rule has been used.
   - *n4* - The cost of using the rule, or the number of compares that are required to determine if this rule is the correct rule to use.
   - *text* - The descriptive text from the classification rule so that you can tell which classification rule is being displayed.

   BB000283I FOR *type* **WORK: TOTAL CLASSIFIED** *n1***, WEIGHTED TOTAL COST** *n2*
   This message shows the summary information for the IIOP, HTTP, INTERNAL, SIP, MDB, or optimized local adapter work classification. This message displays the following information:

- *type* - The type of work that is being displayed. The value must be IIOP, HTTP, INTERNAL, SIP, MDB, or OLA.
- *n1* - The number of requests that were classified using the classification rules.
- *n2* - The weighted total cost, calculated by taking the number of times that each rule was used multiplied by the cost, or number of rule compares that were done, of using the rule and adding those up across all the rules.

  The total cost *n2* divided by the total number of requests classified *n1* equals the cost of using the table. The closer that the value is to one, the lower the cost of using the defined rules. A value of 1 indicates that there is just the default classification, so no requests match it.

7. Repeat these steps until you achieve your optimal workload distribution and costs.

Results

You have used the workload classification document to classify inbound requests.

Workload classification file

The workload classification document is a common XML file that classifies inbound HTTP, IIOP, message-driven bean (MDB), Session Initiation Protocol (SIP), optimized local adapter, and mediation work for the z/OS workload manager.

Usage notes

This topic contains examples of the workload classification file with and without Reliability Availability and Serviceability (RAS) attributes. RAS attributes allow you to achieve request-level RAS granularity for HTTP requests, IIOP requests, MDB requests, and optimized local adapter requests. You specify these attributes on the http_classification_info element, the iiop_classification_info element, the classificationentry element, the sib_classification_info element, the wmqra_classification_info element, and the ola_classification_info element in the workload classification file.

You use the workload classification file when you complete the tasks for classifying z/OS workload or enabling request-level RAS granularity.

Required elements

<?xml version="1.0" encoding="UTF-8"?>
>   Indicates that the workload classification document must be saved in ASCII to be processed by the application server. This statement is required.

<!DOCTYPE Classification SYSTEM "Classifications">
>   Provides the XML parser with the name of the DTD document that is provided with the product, and that is used to validate the workload classification document. The workload classification document that you write must follow the rules that are described in this DTD. You must add this statement to the workload classification document.

Classification
>    ```
>    <Classification schema_version="1.0">
>    ```
>
>   Indicates the root of the workload classification document. Every workload classification document must begin and end with this element. The schema_version attribute is required. The only supported schema_version is 1.0. The Classification element contains one or more InboundClassification elements. For inbound service integration work, the Classification element can also contain up to two SibClassification elements. If classifying inbound messages for delivery to message-driven beans using WebSphere MQ messaging provider activation specifications, the Classification element can contain one or more WMQRAClassification elements.

InboundClassification

```
<InboundClassification type="iiop | http | mdb |internal | sip | ola"
schema_version="1.0" default_transaction_class="value">
```

Use the following rules when using the InboundClassification element:

- The **type** attribute is required. The value must be `internal`, `iiop`, `http`, `mdb`, `sip`, or `ola`. Only one occurrence of an InboundClassification element can occur in the document for each type. There can be up to five InboundClassification elements in a document. The types do not have to be specified in a certain order in your classification document.

- The **schema_version** attribute is required. The value must be set to 1.0.

- The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " ").

- The InboundClassification elements cannot be nested. Each InboundClassification element must end before the next InboundClassification element or SibClassification element can begin.

SibClassification

```
<SibClassification type="jmsra | destinationmediation" schema_version="1.0"
default_transaction_class="value">
```

Use the following rules when using the SibClassification element:

- The **type** attribute is required. The value must be jmsra or destinationmediation. There can be at most one SibClassification element in the document for each type. The types do not have to be specified in a certain order in your classification document.

- The **schema_version** attribute is required. The value must be set to 1.0.

- The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " ").

- The SibClassification elements cannot be nested. Each SibClassification element must end before the next InboundClassification element or SibClassification element can begin.

WMQRAClassification

```
<WMQRAClassification schema_version="1.0" default_transaction_class="value">
```

The following rules apply to the WMQRAClassification element:

- The **schema_version** attribute is required. The value must be set to 1.0.

- The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class.

- The WMQRAClassification elements cannot be nested. Each WMQRAClassification element must end before any other classification elements can begin.

The rules and XML statements for classifying different types of work are similar, but there is slightly different syntax for each type. For more information about the syntax for each type of work, see the following sections:

InboundClassification

- "Optimized local adapter classification" on page 528
- "SIP Classification" on page 532

SibClassification
- "JMS RA classification" on page 532
- "Mediation classification" on page 535

WMQRAClassification
- "WebSphere MQ messaging provider classification" on page 539

Internal Classification

The InboundClassification element with the attribute type="internal" defines the section of the document that is applicable to internal work, such as requests that are dispatched in a servant, that originate in the owning controller. An example of this element follows:

```
<InboundClassification  type="internal" schema_version="1.0"
                        default_transaction_class="value1">
```

If an InboundClassification element with the type="internal" attribute is not specified, internal work is classified using the rules that are specified for IIOP work.

IIOP Classification

The InboundClassification element with the attribute type="iiop" defines the section of the document that is applicable to IIOP classification. An example of this element follows:

```
<InboundClassification  type="iiop" schema_version="1.0"
                        default_transaction_class="value1">
```

You can classify IIOP work based on the following Java Platform, Enterprise Edition (Java EE) application artifacts:

- Application name

  The name of the application that contains the enterprise beans. It is the display name of the application, which might not be the name of the .ear file that contains all the artifacts.

- Module name

  The name of the Enterprise JavaBeans(EJB) .jar file that contains one or more enterprise beans. There can be multiple EJB .jar files in an .ear file.

- Component name

  The name of the EJB that is contained in a module (or EJB .jar file). There can be one or more enterprise beans contained in a .jar file.

- Method name

  The name of a remote method on an EJB.

Classify IIOP work in various applications at any of these levels by using the iiop_classification_info element.

iiop_classification_info

```
<iiop_classification_info transaction_class="value1"
                        application_name="value2"
                        module_name="value3"
                        component_name="value4"
                        method_name="value5"
                        description="value6"
                        dispatch_timeout="value7"
                        queue_timeout_percent ="value8"
                        request_timeout="value9"
                        stalled_thread_dump_action="traceback"
                        cputimeused_limit="value11"
                        cputimeused_dump_action="traceback"
                        dpm_interval="value13"
                        dpm_dump_action="traceback"
```

```
SMF_request_activity_enabled="1"
SMF_request_activity_timestamps="1"
SMF_request_activity_security="1"
SMF_request_activity_CPU_detail="1"
classification_only_trace="1"
message_tag="value20">
```

With the iiop_classification_info element, you can build filters based on the application, module, component, and method names to assign TCLASS values, RAS attributes, or both to inbound requests. Use the following rules when using the iiop_classification_info element:

transaction_class

The transaction_class attribute is optional. If the attribute is not defined, it inherits the transaction class of its parent. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " "). By specifying a null or blank string, you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.

application_name, module_name, component_name, and method_name

The attributes application_name, module_name, component_name, and method_name can be used as you need them. These attributes act as selectors or filters that either assign a transaction class or allow a nested iiop_classification_info element to assign the transaction class. You can specify the values of these attributes in the following ways:

- The exact name of the application, module, component, or method.
- A wild-card value. You can place an asterisk (*) anywhere in a string to indicate that any string that starts with the string preceding the asterisk and ends with the string that follows the asterisk is considered a match. If the asterisk is at the end of the string, any string that starts with the string preceding the asterisk is considered a match.

  Examples:

  - The string `Mar*61`, matches `Mar61`, `March61`, and `Mar20early61`, but does not match `March81` or MAR61.
  - The string `MAR*` matches `MARCH`, `MAR61`, and `MARS`, but does not match `Mar61` or `MAY61`.

  gotcha: The value comparisons that are performed are case sensitive.

You can use any combination of these attributes to make a classification filter. However, use only the granularity that is required. For example, if there is only one application on the application server, the classification rules do not need to specify the application_name attribute.

RAS attributes

You can specify the following RAS attributes on the iiop_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the control_region_wlm_dispatch_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_iiop_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_iiop_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to `1`, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Classification filters

- The iiop_classification_info elements can be nested in a hierarchical manner. By nesting the elements, you can create classification filters that are based on the attribute values. The following filter classifies requests on the EJB1 and EJB2 enterprise beans in the MyAPP1 application:

```
<iiop_classification_info transaction_class="FAST"
                          application_name="MyAPP1"
                          component_name="EJB1" />
<iiop_classification_info transaction_class="SLOW"
                          application_name="MyAPP1"
                          component_name="EJB2" />
```

> The following filter also classifies requests on EJB1 and EJB2 in the MyAPP1 application, but
> also classifies requests on any other EJB in the application:

```
<iiop_classification_info transaction_class="MEDIUM"
                          application_name="MyAPP1">
    <iiop_classification_info transaction_class="FAST"
                          component_name="EJB1" />
    <iiop_classification_info transaction_class="SLOW"
                          component_name="EJB2" />
</iiop_classification_info>
```

- If you specify an attribute value that conflicts with the attribute value of the parent element, the lower-level filter is negated. An example of a child value that conflicts with the attribute value of the parent element follows:

```
<iiop_classification_info transaction_class="FAST"
                          application_name="MyAPP1">
    <iiop_classification_info transaction_class="SLOW"
                          application_name="MyAPP2" />
</iiop_classification_info>
```

> In this example, EJB Requests in MyAPP2 would never be assigned to transaction class
> "SLOW" because the higher level filter only allows IIOP requests for
> application_name="MyAPP1" to be passed through to the lower-level filter.

- The first filter at a specific level that matches the attributes of the request is used, not the best or most restrictive filter. Therefore, the order that you specify filters is important.

```
<iiop_classification_info transaction_class="FAST"
                          application_name="MyAPP" />
    <iiop_classification_info transaction_class="SLOW"
                          component_name="*" />
    <iiop_classification_info transaction_class="MEDIUM"
                          component_name="MySSB" />
</iiop_classification_info>
```

> In the preceding example, all the IIOP requests that are processed by enterprise beans in the
> MyAPP application are assigned a TCLASS value of SLOW. This assignment is done for any
> requests to the MySSB enterprise as well. Even though MySSB is assigned a transaction class,
> the filter is not applied because the first filter was applied and was assigned a TCLASS value of
> SLOW. The remaining list of filters at the same level is ignored.

- The description field is optional. However, you should use a description on all iiop_classification_info elements. The description string prints as part of the operator command support so you can identify the classification rules that are being used. Keep your descriptions reasonably short because they are displayed in the MVS console.

HTTP classification

The InboundClassification element with the attribute type="http" defines the section of the document that is applicable to HTTP classification. An example of this element follows:

```
<InboundClassification   type="http"
                         schema_version="1.0"
                         default_transaction_class="value1">
```

HTTP work can be classified based on the following J2EE artifacts:

- Virtual host name

  Specifies the host name in the HTTP header to which the inbound request is being sent.

- Port number

  Specifies the port on which the HTTP catcher is listening.

- URI (Uniform Resource Identifier)

  The string that identifies the web application.

You can classify HTTP work in various applications at any of these levels by using the http_classification_info element.

```
<http_classification_info transaction_class="value1"
                          host="value2"
                          port="value3"
                          uri="value4"
                          description="value5"
                          dispatch_timeout="value6"
                          queue_timeout_percent ="value7"
                          request_timeout="value8"
                          stalled_thread_dump_action="traceback"
                          cputimeused_limit="value10"
                          cputimeused_dump_action="traceback"
                          dpm_interval="value12"
                          dpm_dump_action="traceback"
                          SMF_request_activity_enabled="1"
                          SMF_request_activity_timestamps="1"
                          SMF_request_activity_security="1"
                          SMF_request_activity_CPU_detail="1"
                          classification_only_trace="1"
                          message_tag="value19"
                          timeout_recovery="value20">
```

With the http_classification_info element, you can build filters based on the host, port, and URI to assign TCLASS values, RAS attributes, or both to inbound requests. Use the following rules when you use the http_classification_info element:

transaction_class

> The transaction_class attribute is optional. If the attribute is not defined, it inherits the transaction class of its parent. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " "). By specifying a null or blank string, you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.

host, port, and uri

> The attributes of host, port, and uri can be used as you need them. These attributes act as selectors or filters that either assign a transaction class or allow a nested http_classification_info element to assign the transaction class. You can specify the values of these attributes in the following ways:
>
> - The exact name of the host, port, or URI.
> - Any value. To specify a match to any value, use the asterisk (*) symbol.
> - A wild card value. You can place an asterisk (*) anywhere in a string to indicate that any string that starts with the string preceding the asterisk and ends with the string that follows the asterisk is considered a match. If the asterisk is at the end of the string, any string that starts with the string preceding the asterisk is considered a match.
>
>   Examples:
>
>   – The string `Mar*61`, matches `Mar61`, `March61`, and `Mar20early61`, but does not match `March81`.
>   – The string `MAR*` matches `MARCH`, `MAR61`, and `MARS`, but does not match `Mar61` or `MAY61`.
>
>   **gotcha:** The value comparisons that are performed are case sensitive.
>
> Use any or all these attributes to make a classification filter. Only use the granularity that is required. For example, if there is only one application on the application server, the classification rules do not need to specify the uri attribute.

RAS attributes

You can specify the following RAS attributes on the http_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the following server properties:

HTTP   protocol_http_timeout_output

HTTPS
        protocol_https_timeout_output

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the following server properties:

HTTP   control_region_http_queue_timeout_percent

HTTPS
        control_region_https_queue_timeout_percent

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the following server properties:

HTTP   server_region_http_stalled_thread_dump_action

HTTPS
        server_region_https_stalled_thread_dump_action

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to `1`, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

timeout_recovery

Specifies the action that the server takes when a request that the classification element classified exceeds its dispatch timeout. Specify the dispatch timeout on the dispatch_timeout attribute.

Valid values are `servant` and `session`.

When the attribute is set to `servant`, the servant address space processing the request terminates with an ABEND EC3 and a reason code 04130007. The controller address space sends a failure response to the client, then closes the socket associated with the request.

When the attribute is set to `session`, the controller address space sends a failure response to the client, then closes the socket associated with the request. The Servant address space is not terminated. The request is not disrupted, but instead is permitted to run to completion.

The attribute overrides the following server properties:

HTTP   protocol_http_timeout_output_recovery

HTTPS
          protocol_https_timeout_output_recovery

Classification filters

- You can nest the http_classification_info elements in a hierarchical manner. You can construct filters based on attribute names. Consider the two following filters:

```
<http_classification_info transaction_class="FAST"
                          host="MyVHost1.com"
                          uri="/MyWebApp1/*" />
<http_classification_info transaction_class="SLOW"
                          host="MyVHost2.com"
                          uri="/MyWebApp2/*" />

<http_classification_info transaction_class="MEDIUM"
                          host="MyVHost1.com">
    <http_classification_info transaction_class="FAST"
                          uri="/MyWebApp1/*" />
    <http_classification_info transaction_class="SLOW"
                          uri="/MyWebApp2/*" />
</http_classification_info>
```

Both filters classify requests to web applications that are identified by context roots /MyWebApp1 and /MyWebApp2 in the application server that is hosting web applications for virtual host MyVHost1.com. However, the second filter also classifies requests on any other context root in the application server.

- Specifying an attribute name that is different from the attribute value of the parent element effectively negates the lower-level filter. For example,

```
<http_classification_info transaction_class="FAST"
                          uri="/MyWebApp1/*">
    <http_classification_info transaction_class="SLOW"
                              uri="/MyWebApp2">
    </http_classification_info>
</http_classification_info>
```

This example would never result in web applications with a context root of /MyWebApp2 being assigned to the transaction class SLOW. The high-level filter only allows HTTP requests with a context root of /MyWebApp1/* to be passed to a lower-level filter.

- The first filter that is at a specific level is used, not the best or most restrictive filter. Therefore, the order of the filters at each level is important. For example:

```
<http_classification_info transaction_class="FAST"
                          host="MyVHost.com" />
    <http_classification_info transaction_class="SLOW"
                             uri="*" />
    <http_classification_info transaction_class="MEDIUM"
                             uri="/MyWebAppX/*" />
</http_classification_info>
```

In this example, HTTP requests processed by the application server by the virtual host "MyVHost.com" are assigned a TCLASS value of SLOW. Even requests to the web application with context root /MyWebAppX are assigned a TCLASS value of SLOW because the filter was not applied. The first filter that matches is used for the TCLASS assignment, and the remainder of the filters at the same level is ignored.

- The description field is optional, however, you should use it on all the http_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

MDB classification

The InboundClassification element with the attribute type="mdb" defines the section of the document that applies to work for EJB 2.0 message-driven beans (MDBs) deployed with listener ports. An example of this element follows:

```
<InboundClassification  type="mdb"
                        schema_version="1.0"
                        default_transaction_class="qrs">
```

Each InboundClassification element can contain one or more endpoint elements with a messagelistenerport type defined. Define one endpoint element for each listener port that is defined in the server that you want to associate transaction classes with the message-driven bean. An example of the endpoint element follows:

```
<endpoint  type="messagelistenerport"
           name="IPVListenerPort"
           defaultclassification="MDBX"
           description="ABC">
```

Use the following rules when defining your endpoint elements:
- The type attribute must always equal messagelistenerport.
- The name attribute corresponds to the listener for your endpoint element. The value of the name attribute must be the name of the listener port that is specified in the administration console for the server.
- The defaultclassification element is the default transaction class that is associated with the message-driven beans. The value of this attribute overrides the default transaction classification value.
- The description field is optional, however, you should use it on all the endpoint elements. The description is displayed when you monitor the transaction classes in the MVS console.

Each endpoint element can have zero, one, or more classificationentry elements. An example of a classification entry element follows:

```
<classificationentry  selector="Location=&apos;East&apos;"
                      classification="MDB2"
                      description="XYZ"
                      dispatch_timeout="value1"
                      queue_timeout_percent ="value2"
                      request_timeout="value3"
                      stalled_thread_dump_action="traceback"
                      cputimeused_limit="value5"
                      cputimeused_dump_action="traceback"
                      dpm_interval="value7"
                      dpm_dump_action="traceback"
                      SMF_request_activity_enabled="1"
                      SMF_request_activity_timestamps="1"
                      SMF_request_activity_security="1"
                      SMF_request_activity_CPU_detail="1"
                      classification_only_trace="1"
                      message_tag="value14"/>
```

selector

Use the selector attribute of the classificationentry element to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor. Use the following rules when defining your classificationentry elements:

- The value of the selector attribute must match exactly to the selector clause in the MDB deployment descriptor.
- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < symbol with the `<` entity reference and the and > symbol with the `>` entity reference. Similarly, if you use an apostrophe or quotation mark, use the `'` and `"` entity references.

classification

The classification attribute is optional. If the attribute is not defined, it inherits the classification of its parent. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " "). By specifying a null or blank string, you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.

RAS attributes

You can specify the following RAS attributes on the classificationentry element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the control_region_mdb_request_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_mdb_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump, javacore, heapdump, traceback, javatdump,` and `none`.

The attribute overrides the server_region_mdb_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to 1, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Optimized local adapter classification

The InboundClassification element with the attribute type="ola" defines the section of the document that is applicable to optimized local adapter classification. An example of this element follows:

```
<InboundClassification  type="ola" schema_version="1.0"
                        default_transaction_class="value1"
>
```

You can classify optimized local adapter work by adding a section for each EJB application that uses the service name or Java Naming and Directory Interface (JNDI) home name. You can use a wildcard for the JNDI home name.

Classify optimized local adapter work in various applications at any of these levels by using the ola_classification_info element.

ola_classification_info

```
<ola_classification_info  transaction_class="value1"
                          propagate_transaction_name="value2"
                          service_name="value3"
                          description="value4"
                          dispatch_timeout="value5"
                          queue_timeout_percent ="value6"
                          request_timeout="value7"
                          stalled_thread_dump_action="traceback"
                          cputimeused_limit="value9"
                          cputimeused_dump_action="traceback"
                          dpm_interval="value11"
                          dpm_dump_action="traceback"
                          SMF_request_activity_enabled="1"
                          SMF_request_activity_timestamps="1"
                          SMF_request_activity_security="1"
                          SMF_request_activity_CPU_detail="1"
                          classification_only_trace="1"
                          message_tag="value18">
```

With the ola_classification_info element, you can build filters based on the service or JNDI name. Use the name to assign TCLASS values, RAS attributes, or both to inbound requests. Use the following rules when using the ola_classification_info element:

transaction_class

The transaction_class attribute is optional. If the attribute is not defined, it inherits the transaction class of its parent. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " "). By

specifying a null or blank string, you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.

propagate_transaction_name

The propagate_transaction_name attribute is optional. If the attribute is specified, the string value must be `true` or `false`. By specifying a value of `true`, the WLM service class from Customer Information Control System (CICS) is propagated to the application server on each request or on each matching request if the service_name filter is specified. The work dispatched in the application server through the optimized local adapter runs under the same service class as the client request.

service_name

The service_name attribute is optional. This attribute acts as a selector or filter that either assigns a transaction class or allows a nested ola_classification_info element to assign the transaction class. You can specify the value of this attribute in the following ways:

- The exact service name or JNDI home name of the EJB application to be driven.
- A wild-card value. You can place an asterisk (*) anywhere in a string to indicate that any string that starts with the string preceding the asterisk and ends with the string that follows the asterisk is considered a match. If the asterisk is at the end of the string, any string that starts with the string preceding the asterisk is considered a match.

  Examples
  - `service_name="ejb/mySecondBean"`
  - `service_name="ejb/my*Bean"`
  - `service_name="ejb/my*"`
  - `service_name="ejb/security/*"`

  CAUTION: The value comparisons that the application server performs are case sensitive.

RAS attributes

You can specify the following RAS attributes on the ola_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the control_region_wlm_dispatch_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_iiop_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_iiop_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to 1, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Classification filters

- The ola_classification_info elements can be nested in a hierarchical manner. By nesting the elements, you can create classification filters that are based on the attribute values.

```
<ola_classification_info transaction_class="MEDIUM"
                    service_name="ejb/my*Bean">
    <ola_classification_info transaction_class="FAST"
                        service_name="ejb/myFastBean" />
    <ola_classification_info transaction_class="SLOW"
                        service_name="ejb/mySlowBean" />
</ola_classification_info>
```

- The description field is optional. However, use a description on all ola_classification_info elements. The description string prints as part of the operator command support so you can identify the classification rules that are being used. Keep your descriptions reasonably short because they are displayed in the MVS console.

SIP Classification

The InboundClassification element with the attribute type="sip" defines the section of the document that sets the default transaction class for Session Initiation Protocol (SIP) requests. An example of this element follows:

```
<InboundClassification  type="sip" schema_version="1.0"
                        default_transaction_class="value1">
```

JMS RA classification

The SibClassification element with the attribute type="jmsra" defines the section of the document that applies to work for message-driven beans (MDBs) deployed against JCA 1.5-compliant resources for use with the JCA resource adapter (RA) of the default messaging provider. An example of this element follows:

```
<SibClassification  type="jmsra"
                    schema_version="1.0"
                    default_transaction_class="a">
```

Each SibClassification element can contain one or more sib_classification_info elements. An example of a classification entry element follows:

```
<sib_classification_info
                    transaction_class="sibb"
                    selector="user.Location=&apos;East&apos;"
                    bus="bigrred"
                    destination="abusqueue"
                    description="Some words"
                    discriminator="XPath Expression"
                    dispatch_timeout="value1"
                    queue_timeout_percent ="value2"
                    request_timeout="value3"
                    stalled_thread_dump_action="traceback"
                    cputimeused_limit="value5"
                    cputimeused_dump_action="traceback"
                    dpm_interval="value7"
                    dpm_dump_action="traceback"
                    SMF_request_activity_enabled="1"
                    SMF_request_activity_timestamps="1"
                    SMF_request_activity_security="1"
                    SMF_request_activity_CPU_detail="1"
                    classification_only_trace="1"
                    message_tag="value14"/>
```

selector
> Use the selector attribute of the sib_classification_info element to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor. Use the following rules when defining your sib_classification_info elements:
>
> • The value of the selector attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification, but it can operate on SIMessage messages (more than JMS messages). The syntax can select on system properties (including JMS headers, JMSX properties, and JMS_IBM_properties) and user properties (which must be prefixed by ".user" - for example, for the user property "Location", the selector would specify "user.Location" as shown in the preceding example). For more information, see the topic on working with message properties.
>
> • The value of the selector attribute must have the correct syntax for an XML document. You must replace the < symbol with the `<` entity reference and the > symbol with the `>` entity reference. Similarly, if you use an apostrophe or quotation mark, use the `'` and `"` entity references.

bus   The name of the service integration bus on which the target destination is assigned. The classification applies to the bus named by this property, or to any bus if you do not specify this property. The destinations to which the classification applies depend on your use of the destination property.

destination
> The name of the target bus destination to which the message has been delivered. This is the name of a queue or topic space. The classification applies to the destination named by this

property, or any destination if you do not specify this property. The service integration buses to which the classification applies depend on your use of the bus property.

discriminator

The property applies only when the destination property names a topic space. This discriminator value is then an XPath expression that selects one or more topics within the topic space.

description

Although the description field is optional, you should use it on all the sib_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

RAS attributes

You can specify the following RAS attributes on the sib_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the control_region_wlm_dispatch_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_iiop_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_iiop_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to `1`, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Each sib_classification_info element can contain one or more of these properties as needed to classify the work for a message. A sib_classification_info element cannot contain more than one instance of each property.

If a message matches several sib_classification_info elements, the element that appears first is used. For example, consider the following specifications:

```
<sib_classification_info bus="MyBus" transaction_class="a" />
<sib_classification_info destination="MyDest" transaction_class="b" />
```

A message that arrives at destination MyDest from the service integration bus MyBus is assigned the classification "a". A message that arrives at MyDest from another bus is assigned the classification "b".

If a message does not match any sib_classification_info element in an enclosing SibClassification element, the message is assigned the default classification from the SibClassification element.

If a message does not match any sib_classification_info element in any SibClassification element, or if no SibClassification elements are defined, all work receives a built-in default classification with the value "SIBUS". You must perform z/OS Workload Manager actions that are required to use the TCLASS value "SIBUS", as described in "Classifying z/OS workload" on page 510.

Mediation classification

The SibClassification element with the attribute type="destinationmediation" defines the section of the document that applies to work for mediations assigned to destinations on a service integration bus. An example of this element follows:

```
<SibClassification type="destinationmediation"
                schema_version="1.0"
                default_transaction_class="b">
```

Each SibClassification element can contain one or more sib_classification_info elements. An example of a classification entry element follows:

```
<sib_classification_info
                transaction_class="e"
                selector="user.Location=&apos;East&apos;"
                bus="bigrred"
                destination="themoon"
                discriminator="sides/dark"
                description="n"
                dispatch_timeout="value1"
                queue_timeout_percent ="value2"
                request_timeout="value3"
                stalled_thread_dump_action="traceback"
                cputimeused_limit="value5"
                cputimeused_dump_action="traceback"
                dpm_interval="value7"
                dpm_dump_action="traceback"
                SMF_request_activity_enabled="1"
                SMF_request_activity_timestamps="1"
```

```
SMF_request_activity_security="1"
SMF_request_activity_CPU_detail="1"
classification_only_trace="1"
message_tag="value14"/>
```

selector

Use the selector attribute of the sib_classification_info element to assign a transaction class to a mediation that has a selector clause in its deployment descriptor. Use the following rules when defining your sib_classification_info elements:

- The value of the selector attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification, but it can operate on SIMessage messages (more than JMS messages). The syntax can select on system properties (including JMS headers, JMSX properties, and JMS_IBM_properties) and user properties (which must be prefixed by ".user" - for example, for the user property "Location", the selector would specify "user.Location" as shown in the preceding example).

- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < symbol with the < entity reference and the > symbol with the > entity reference. Similarly, if you use an apostrophe or quotation mark, use the ' and " entity references.

bus   The name of the service integration bus on which the target destination is assigned. The classification applies to the bus named by this property, or to any bus if you do not specify this property. The destinations to which the classification applies depend on your use of the destination property.

destination

The name of the target bus destination to which the message has been delivered. This is the name of a queue or topic space. The classification applies to the destination named by this property, or any destination if you do not specify this property. The service integration buses to which the classification applies depend on your use of the bus property.

discriminator

The property applies only when the destination property names a topic space. This discriminator value is then an XPath expression that selects one or more topics within the topic space.

description

Although the description field is optional, you should use it on all the sib_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

RAS attributes

You can specify the following RAS attributes on the sib_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the control_region_wlm_dispatch_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_iiop_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_iiop_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are 0 (no) and 1 (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to 1, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Each sib_classification_info element can contain one or more of these properties as needed to classify the work for a message. A sib_classification_info element cannot contain more than one instance of each property.

If a message matches several sib_classification_info elements, the element that appears first is used. For example, consider the following specifications:

```
<sib_classification_info transaction_class="e" destination="themoon" description="n" />
<sib_classification_info transaction_class="f" description="n" />
```

A message that arrives at the mediated destination themoon is assigned the classification "e". A message that arrives at another mediated destination is assigned the classification "f".

If a message does not match any sib_classification_info element in an enclosing SibClassification element, the message is assigned the default classification from the SibClassification element.

If a message does not match any sib_classification_info element in *any* SibClassification element, or if *no* SibClassification elements are defined, all work receives a built-in default classification with the value "SIBUS". You must perform z/OS Workload Manager actions that are required to use the TCLASS value "SIBUS", as described in "Classifying z/OS workload" on page 510.

WebSphere MQ messaging provider classification

The WMQRAClassification element defines the section of the document that applies to work for message-driven beans (MDBs) deployed against WebSphere MQ messaging provider activation specifications. An example of this element follows:

```
<WMQRAClassification default_transaction_class="TC99" schema_version="1.0">
```

A WMQRAClassification element can contain one or more wmqra_classification_info elements. Two examples of wmqra_classification_info elements follow:

```
<wmqra_classification_info transaction_class="TC_4"
                    destination="topic://a/b/*"
                    description="Any topics starting with a/b/ map to TC_4"/>

<wmqra_classification_info transaction_class="TC_3"
                    selector="JMSPriority&gt;3 AND JMSPriority&lt;8"
                    destination="queue://QMGR1/Q1"
                    queue_manager="QMGR1"
                    description="medium priorities with a queue manager name of QMGR1 and
                                 a queue name of Q1 map to TC_3"
                    dispatch_timeout="value1"
                    queue_timeout_percent ="value2"
                    request_timeout="value3"
                    stalled_thread_dump_action="traceback"
                    cputimeused_limit="value5"
                    cputimeused_dump_action="traceback"
                    dpm_interval="value7"
                    dpm_dump_action="traceback"
                    SMF_request_activity_enabled="1"
                    SMF_request_activity_timestamps="1"
                    SMF_request_activity_security="1"
                    SMF_request_activity_CPU_detail="1"
                    classification_only_trace="1"
                    message_tag="value14"/>
```

selector

Use the selector attribute of the wmqra_classification_info element to assign a transaction class to a message based on its properties. This attribute can also be used to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor:

- The value of the selector attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification.

- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < symbol with the < entity reference and the > symbol with the > entity reference. Similarly, if you use an apostrophe or quotation mark, use the ' and " entity references.

destination

A URI representing the WebSphere MQ destination to which the message has been delivered. The classification applies to the destination named by this property, or to any destination if you do not specify this property. If the URI represents a queue type destination it can optionally include a queue manager name, but that name is ignored and is not be used for classification. If the URI represents a topic type destination, it can make use of wildcards. For more information on wildcard support with WebSphere MQ see the WebSphere MQ information center.

queue_manager

The name of the WebSphere MQ queue manager to which the message has been delivered. The classification applies to the queue manager named by this property, or to any queue manager if you do not specify this property. The queue manager name must conform to WebSphere MQ naming conventions.

Note that this field must not be set to the name of a WebSphere MQ queue sharing group. Instead, you must either create a wmqra_classification_info element for every queue manager in the queue sharing group, or base the classification on something else such as the destination attribute.

description

Although the description field is optional, you must use it on all the wmqra_classification_info elements.

RAS attributes

You can specify the following RAS attributes on the wmqra_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the control_region_wlm_dispatch_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_iiop_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_iiop_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to `1`, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Each wmqra_classification_info element can contain one or more of these properties as needed to classify the work for a message. A wmqra_classification_info element cannot contain more than one instance of each property.

If a message matches several wmqra_classification_info elements, the element that appears first is used. For example, consider the following specifications:

```
<wmqra_classification_info queue_manager="QMGR1" transaction_class="TC_1" />
<wmqra_classification_info destination="queue:///Q1" transaction_class="TC_2" />
```

A message that arrives at destination Q1 on queue manager QMGR1 is assigned the classification "TC_1". A message that arrives at Q1 from another queue manager is assigned the classification "TC_2".

If a message does not match any wmqra_classification_info element in an enclosing WMQRAClassification element, the message is assigned the default classification from the WMQRAClassification element. If there are multiple WMQRAClassification elements, the default transaction class from the first WMQRAClassification element is used.

If no WMQRAClassification elements are defined, all work receives the default classification "WMQRA". You must perform z/OS Workload Manager actions that are required to use the TCLASS value "WMQRA", as described in "Classifying z/OS workload" on page 510.

Sample z/OS workload classification document without RAS attributes

The sample z/OS workload classification document contains attributes for classifying inbound HTTP, IIOP, Session Initiation Protocol (SIP), and message-driven bean (MDB) work requests for the z/OS workload manager. This sample does not contain RAS attributes.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Classification SYSTEM "Classification.dtd" >
<Classification schema_version="1.0">
<!--
     Internal Classification Rules
-->
  <InboundClassification  type="internal"
                     schema_version="1.0"
                     default_transaction_class="value1"/>

<!--
     IIOP Classification Rules
-->
  <InboundClassification type="iiop"
                     schema_version="1.0"
                     default_transaction_class="A0">
    <iiop_classification_info transaction_class="A1"
                         application_name="IIOPStatelessSampleApp"
                         module_name="StatelessSample.jar"
                         component_name="Sample20"
                         description="Sample20 EJB Classification">
    <iiop_classification_info transaction_class=""
                         method_name="echo"
                         description="No TCLASS for echo()" />
    <iiop_classification_info  transaction_class="A1B"
                         method_name="ping"
                         description="Ping method" />
```

```xml
          </iiop_classification_info>
          <iiop_classification_info application_name="*"
                                    module_name="*"
                                    component_name="*"
                                    transaction_class="A2"
                                    description="TCLASS the rest to A2">
          <iiop_classification_info transaction_class="A2A"
                                    method_name="resetFilter"
                                    description="Sp1 case resetFilter()" />
          </iiop_classification_info>
      </InboundClassification>

<!--
        HTTP Classification Rules
-->
      <InboundClassification type="http"
                             schema_version="1.0"
                             default_transaction_class="M">
        <http_classification_info transaction_class="N"
                                  host="yourhost.yourcompany.com"
                                  description="Virtual Host yourhost">
        <http_classification_info transaction_class="O"
                                  port="9080"
                                  description="Def yourhost HTTP reqs">
        <http_classification_info transaction_class="Q"
                                  uri="/gcs/admin"
                                  description = "Gcs" />
        <http_classification_info  transaction_class="S"
                                  uri="/gcs/admin/1*"
                                  description="GCS login" />
        <http_classification_info transaction_class="P"
                                  port="9081"
                                  description=" Def yourhost HTTPS reqs "/>
        <http_classification_info transaction_class=""
                                  uri="/gcss/mgr/*"
                                  description="GCSS Mgr" />
        </http_classification_info>
        </http_classification_info>
      </InboundClassification>

<!--
        SIP Classification Rules
-->
      <InboundClassification type="sip"
                             schema_version="1.0"
                             default_transaction_class="value1"/>

<!--
        MDB Classification Rules
-->
      <InboundClassification type="mdb"
                             schema_version="1.0"
                             default_transaction_class="qrs">
        <endpoint type="messagelistenerport"
                  name="IVPListenerPort"
                  defaultclassification="MDBX"
                  description="ABC">
          <classificationentry  selector="Location=&apos;East&apos;"
                                classification="MDB1"
                                description="DEF"/>
          <classificationentry  selector="Location=&lt;&gt;&apos;East&apos;"
                                classification="MDB2"
                                description="XYZ" />
        </endpoint>
        <endpoint  type="messagelistenerport"
                   name="SimpleMDBListenerPort"
                   defaultclassification="MDBX"
                   description="GHI" />

      </InboundClassification>

      <SibClassification type="jmsra" schema_version="1.0"
            default_transaction_class="a">
        <sib_classification_info transaction_class="b"
            selector="user.Location=&apos;East&apos;" bus="magic"
            destination="nowhere" description="n" />
        <sib_classification_info transaction_class="c"
            selector="user.Location=&apos;West&apos;" bus="omni"
description="n" />
      </SibClassification>
      <SibClassification type="destinationmediation" schema_version="1.0"
            default_transaction_class="b">
        <sib_classification_info transaction_class="e"
            selector="user.Location=&apos;East&apos;" destination="themoon"
            discriminator="sides/dark" description="n" />
        <sib_classification_info transaction_class="f"
            selector="user.Location=&apos;West&apos;" description="n"
/>
      </SibClassification>
```

```
        <WMQRAClassification default_transaction_class="TC99" schema_version="1.0">
            <wmqra_classification_info transaction_class="TC_1"
                                    queue_manager="GOLD"
                                    description="gold queue manager maps
to TC_1"/>
            <wmqra_classification_info transaction_class="TC_2"
                                    selector="JMSPriority&gt;7"
                                    description="high priorities maps to
TC_2"/>
            <wmqra_classification_info transaction_class="TC_3"
                                    selector="JMSPriority&gt;3 AND
JMSPriority&lt;8"
                                    description="medium priorities maps
to TC_3"/>
        </WMQRAClassification>

<!--
      OLA Classification Rules
-->
    <InboundClassification type="ola"
                        schema_version="1.0"
                        default_transaction_class="A0">
      <ola_classification_info transaction_class="FAST1"
                        service_name="ejb/InteractiveTransactionBean"
                        description="EJB classification for quick turnaround"/>
      <ola_classification_info transaction_class="SLOW1"
                        service_name="ejb/BackgroundBean"
                        description="EJB classification for low priority" />
      <ola_classification_info propagate_transaction_name="true"
                        service_name="ejb/CalledFromCICSBean"
                        description="use same service class as client" />
    </InboundClassification>

<!--
   Workload Classification Document for P5SR01x servers
   Change History
   _____
   Activity                Date            Author
   Created                 01-28-2005      IPL

-->

</Classification>
```

Sample z/OS workload classification document containing RAS attributes

The sample z/OS workload classification document contains attributes for classifying inbound HTTP, IIOP, and MDB work requests for the z/OS workload manager. This sample contains RAS attributes.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Classification SYSTEM "Classification.dtd" >
<Classification schema_version="1.0">

<!-- IIOP Classification Rules -->
<InboundClassification
  type="iiop"
  schema_version="1.0"
  default_transaction_class="TC">

  <!--
      IIOP classification info for app "EJBApp1".  The classification element defines a
      transaction_class of "TC1".  This value overrides the default_transaction_class of
      the InboundClassification element ("TC").  The classification element also
      specifies a message_tag, which is applied to all requests that are classified
      underneath this classification element.                                  -->
  <iiop_classification_info
    application_name="EJBApp1"
    transaction_class="TC1"
    message_tag="EJBApp1">

    <!--
        EJBApp1 contains a jar module named "MyEJB.jar" with an EJB named "MyEJBClass".
        The transaction_class is not defined for this element; thus the element
        inherits the transaction_class from its parent node, "TC1".  This element also
        inherits the message_tag attribute from its parent node, "EJBApp1".  In general,
        a classification element inherits all the RAS attributes from all its ancestor
        nodes, with nearer ancestor nodes (for example, direct parents) taking precedence over
        ancestor nodes further up the chain (for example, grandparents).          -->
    <iiop_classification_info
      module_name="MyEJB.jar"
      component_name="MyEJBClass">

      <!--
          MyEJBClass contains methods named "helloWorld" and "goodbyeWorld".  helloWorld
          is assigned a dispatch_timeout of 30 seconds and a queue_timeout_percent of 90,
          meaning that the queue timeout value is 90% of the dispatch_timeout value.  The
          classification element also specifies SMF_request_activity_enabled=1, meaning
```

```
          SMF 120 subtype 9 records are collected for all requests targeted against
          the helloWorld method.  Also note that this classification element does not
          define a transaction_class; therefore it inherits the transaction_class from
          the nearest ancestor element that defines one.  In this case, the nearest
          ancestor element that defines a transaction_class is the grandparent element,
          "TC1".  Note: if no ancestor elements define a transaction_class, then the
          classification element inherits the default_transaction_class of the
          InboundClassification element.  The default_transaction_class on the
          InboundClassification is required.
          This classification element also inherits the message_tag attribute from its
          grandparent element, "EJBApp1".                                            -->
    <iiop_classification_info
      method_name="helloWorld"
      dispatch_timeout="30"
      queue_timeout_percent="90"
      SMF_request_activity_enabled="1"
    />

    <!--
        The goodbyeWorld method specifies a dispatch_timeout of 60 seconds.  The
        classification element also defines a transaction_class, "TC1gbye", which
        overrides the transaction_class defined by its ancestry.  This element inherits
        the message_tag of its ancestry, "EJBApp1".                                  -->
    <iiop_classification_info
      method_name="goodbyeWorld"
      transaction_class="TC1gbye"
      dispatch_timeout="60"
    />

  </iiop_classification_info>
</iiop_classification_info>

<!--
      IIOP classification info for app "EJBApp2".  The classification element defines a
      dispatch_timeout of 15 seconds and a message_tag of "EJBApp2".  The
      transaction_class is inherited from the default_transaction_class on the
      InboundClassification, "TC".   All requests that are classified under this
      classification element have a 15 second dispatch timeout, and all trace
      records and log messages generated by those requests are tagged with the
      message_tag attribute value,  "EJBApp2".                                        -->
<iiop_classification_info
  application_name="EJBApp2"
  dispatch_timeout="15"
  message_tag="EJBApp2">

    <!--
        EJBApp2 contains two jar modules, "MyEJB2a.jar" and "MyEJB2b.jar".  The following
        two classification elements define a transaction_class for each jar module.  No
        other attributes are defined.  Both elements inherit the attributes of their
        ancestor nodes (dispatch_timeout="15" and message_tag="EJBApp2").             -->
    <iiop_classification_info
      module_name="MyEJB2a.jar"
      transaction_class="TC2a"
    />

    <iiop_classification_info
      module_name="MyEJB2b.jar"
      transaction_class="TC2b"
    />

  </iiop_classification_info>


<!--
      The following classification element defines attributes for a specific module,
      component, and method of the application "EJBApp3".  The module is
      "MyEJB3.jar", component
      is "MyEJB3Class", and method is "method3".   The transaction_class, dispatch_timeout
      queue_timeout_percent, SMF_request_activity_enabled, and
      SMF_request_activity_timestamps are all defined for this specific method in the
      EJBApp3 application.  No other method on no other EJB within this application are
      assigned these attributes.                                     -->
  <iiop_classification_info
    application_name="EJBApp3"
    module_name="MyEJB3.jar"
    component_name="MyEJB3Class"
    method_name="method3"
    transaction_class="TC3"
    dispatch_timeout="40"
    queue_timeout_percent="90"
    SMF_request_activity_enabled="1"
    SMF_request_activity_timestamps="1"
  />

</InboundClassification>


<!-- Internal Classification Rules -->
<InboundClassification
  type="internal"
```

```
    schema_version="1.0"
    default_transaction_class="internal" />

<!-- HTTP Classification Rules -->
<InboundClassification
  type="http"
  schema_version="1.0"
  default_transaction_class="HTC">

  <!--
      The following classification element defines a transaction_class, "HTC8080", for
      all HTTP requests received on host "my.server.com" and port "8080".  The
      classification element also defines RAS attributes dispatch_timeout,
      queue_timeout_percent, timeout_recovery, and stalled_thread_dump_action.  All
      child elements underneath this element inherit these RAS attributes.          -->
  <http_classification_info
    host="my.server.com"
    port="8080"
    transaction_class="HTC8080"
    dispatch_timeout="100"
    queue_timeout_percent="98"
    timeout_recovery="session"
    stalled_thread_dump_action="javacore">

    <!--
        The following classification element applies to all HTTP requests with a URI that
        begins with "/PlantsByWebSphere/".  Every HTTP request received on host
        my.server.com and port 8080 with a URI that begins with /PlantsByWebSphere
        fall under this classification (note: host and port inherited from the parent
        element).  The classification element also defines the message_tag attribute,
        "plantsbw", which are added to every trace record and log message generated
        by any /PlantsByWebSphere/* request.                                         -->
    <http_classification_info
      uri="/PlantsByWebSphere/*"
      message_tag="plantsbw">

      <!--
          The following classification element applies to all HTTP requests with a URI
          that matches "/PlantsByWebSphere/*.jpg (for example, /PlantsByWebSphere/mypic.jpg,
          /PlantsByWebSphere/some/path/anotherpic.jpg).  Again, this filter applies only
          to requests received on host my.server.com and port 8080 (as designated by
          an ancestor node).   The classification element defines a transaction_class,
          "HTCPjpg" and a dispatch_timeout, "10".  It inherits the remaining attributes
          from its ancestor nodes.                                                    -->
      <http_classification_info
        uri="*.jpg"
        transaction_class="HTCPjpg"
        dispatch_timeout="10"
      />

      <!--
          The following classification element applies to all HTTP requests with a URI
          that matches "/PlantsByWebSphere/*.html (for example, /PlantsByWebSphere/index.html,
          /PlantsByWebSphere/some/path/afile.html).                                  -->
      <http_classification_info
        uri="*.html"
        transaction_class="HTChtml"
      />
    </http_classification_info>
  </http_classification_info>

  <!--
      The following classification element defines a transaction_class, "HTC80", for
      all HTTP requests received on port "80".  The host attribute is not
      defined;  thus, this element matches any host.                                  -->
  <http_classification_info
    port="80"
    transaction_class="HTC80"
    dispatch_timeout="60"
    timeout_recovery="servant"
    message_tag="vanilla"
  />

</InboundClassification>

<!-- MDB Classification Rules -->
<InboundClassification
  type="mdb"
  schema_version="1.0"
  default_transaction_class="mdbdflt">

  <!-- Endpoint for LP 1414, skLP1, for MDB Plan 'A' Test  -->
  <endpoint
    type="messagelistenerport"
    name="skLP1"
    defaultclassification="lp1dft"
    description="Endpoint for LP 1414, skLP1, for MDB Plan 'A' Test">

    <classificationentry
      selector="JMSCorrelationID='TestCase1'"
```

546　Setting up the application serving environment

```
              classification="lp1s1"
              description="New MDB Sample, TestCase1"
              cputimeused_limit="200101"
              request_timeout="20"
              dispatch_timeout="30"
              dpm_interval="0"
              queue_timeout_percent="20"
              stalled_thread_dump_action="traceback"
              />
    </endpoint>

</InboundClassification>


<!-- SIB Classification Rules -->
<SibClassification type="jmsra" schema_version="1.0" default_transaction_class="Dclass">
      <sib_classification_info transaction_class="Tclass"
                               bus="testbus"
                               destination="themoon"
                               description="test1"
                               dispatch_timeout="30"
                               queue_timeout_percent="20"
                               request_timeout="20"
                               stalled_thread_dump_action="traceback"
                               cputimeused_limit="200101"
                               cputimeused_dump_action="traceback"
                               dpm_interval="0"
                               dpm_dump_action="traceback"
                               classification_only_trace="1"
                               message_tag="sibreqst"
      />
    </SibClassification>

</Classification>
```

DTD

The following DTD defines the elements and attributes used in the preceding sample workload classification documents.

```
<?xml version='1.0' encoding="UTF-8"?>
<!ELEMENT Classification (InboundClassification|SibClassification|WMQRAClassification)+>
<!ATTLIST Classification schema_version CDATA #REQUIRED>
<!ELEMENT InboundClassification ((iiop_classification_info*|http_classification_info*|endpoint*|ola_classification_info*))>
<!ATTLIST InboundClassification type (iiop|mdb|http|internal|sip|ola) #REQUIRED>
<!ATTLIST InboundClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST InboundClassification schema_version CDATA #REQUIRED>
<!ELEMENT iiop_classification_info (iiop_classification_info*)>
<!-- inputs -->
<!ATTLIST iiop_classification_info activity_workload_classification CDATA #IMPLIED>
<!ATTLIST iiop_classification_info application_name CDATA #IMPLIED>
<!ATTLIST iiop_classification_info component_name CDATA #IMPLIED>
<!ATTLIST iiop_classification_info description CDATA #IMPLIED>
<!ATTLIST iiop_classification_info method_name CDATA #IMPLIED>
<!ATTLIST iiop_classification_info module_name CDATA #IMPLIED>
<!-- outputs -->
<!ATTLIST iiop_classification_info transaction_class CDATA #IMPLIED>
<!ATTLIST iiop_classification_info dispatch_timeout CDATA #IMPLIED>
<!-- control_region_wlm_dispatch_timeout -->
<!ATTLIST iiop_classification_info queue_timeout_percent CDATA #IMPLIED>
<!-- control_region_iiop_queue_timeout_percent -->
<!ATTLIST iiop_classification_info request_timeout CDATA #IMPLIED>
<!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST iiop_classification_info stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
<!-- server_region_iiop_stalled_thread_dump_action -->
<!ATTLIST iiop_classification_info cputimeused_limit CDATA #IMPLIED>
        <!-- server_region_request_cputimeused_limit -->
<!ATTLIST iiop_classification_info cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_cputimeused_dump_action -->
<!ATTLIST iiop_classification_info dpm_interval CDATA #IMPLIED>
        <!-- MODIFY [JOBNAME],DPM,IIOP= -->
<!ATTLIST iiop_classification_info dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_dpm_dump_action -->
<!ATTLIST iiop_classification_info SMF_request_activity_enabled (0|1) #IMPLIED>
     <!-- server_SMF_request_activity_enabled -->
<!ATTLIST iiop_classification_info SMF_request_activity_timestamps (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST iiop_classification_info SMF_request_activity_security (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_security -->
<!ATTLIST iiop_classification_info SMF_request_activity_CPU_detail (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST iiop_classification_info classification_only_trace (0|1) #IMPLIED>
<!ATTLIST iiop_classification_info message_tag CDATA #IMPLIED>

<!ELEMENT endpoint (classificationentry*)>
<!ATTLIST endpoint defaultclassification CDATA #REQUIRED>
<!ATTLIST endpoint name CDATA #REQUIRED>
```

```
<!ATTLIST endpoint type (messagelistenerport) #REQUIRED>
<!ATTLIST endpoint description CDATA #IMPLIED>
<!ELEMENT classificationentry EMPTY>
<!-- inputs -->
<!ATTLIST classificationentry selector CDATA #REQUIRED>
<!ATTLIST classificationentry description CDATA #IMPLIED>
<!-- outputs -->
<!ATTLIST classificationentry classification CDATA #REQUIRED>
<!ATTLIST classificationentry dispatch_timeout CDATA #IMPLIED>
        <!-- control_region_mdb_request_timeout -->
<!ATTLIST classificationentry queue_timeout_percent CDATA #IMPLIED>
   <!-- control_region_mdb_queue_timeout_percent -->
<!ATTLIST classificationentry request_timeout CDATA #IMPLIED>
         <!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST classificationentry stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
 <!-- server_region_mdb_stalled_thread_dump_action -->
<!ATTLIST classificationentry cputimeused_limit CDATA #IMPLIED>
        <!-- server_region_request_cputimeused_limit -->
<!ATTLIST classificationentry cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
   <!-- server_region_cputimeused_dump_action -->
<!ATTLIST classificationentry dpm_interval CDATA #IMPLIED>
           <!-- MODIFY [JOBNAME],DPM,IIOP= -->
<!ATTLIST classificationentry dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
   <!-- server_region_dpm_dump_action -->
<!ATTLIST classificationentry SMF_request_activity_enabled (0|1) #IMPLIED>
      <!-- server_SMF_request_activity_enabled -->
<!ATTLIST classificationentry SMF_request_activity_timestamps (0|1) #IMPLIED>
   <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST classificationentry SMF_request_activity_security (0|1) #IMPLIED>
     <!-- server_SMF_request_activity_security -->
<!ATTLIST classificationentry SMF_request_activity_CPU_detail (0|1) #IMPLIED>
   <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST classificationentry classification_only_trace (0|1) #IMPLIED>
<!ATTLIST classificationentry message_tag CDATA #IMPLIED>
<!ELEMENT http_classification_info (http_classification_info*)>
<!-- inputs -->
<!ATTLIST http_classification_info host CDATA #IMPLIED>
<!ATTLIST http_classification_info port CDATA #IMPLIED>
<!ATTLIST http_classification_info uri CDATA #IMPLIED>
<!ATTLIST http_classification_info description CDATA #IMPLIED>
<!ATTLIST http_classification_info transaction_class CDATA #IMPLIED>
<!-- outputs -->
<!ATTLIST http_classification_info dispatch_timeout CDATA #IMPLIED>
        <!-- protocol_http_timeout_output -->
<!ATTLIST http_classification_info queue_timeout_percent CDATA #IMPLIED>
   <!-- control_region_http_queue_timeout_percent -->
<!ATTLIST http_classification_info request_timeout CDATA #IMPLIED>
         <!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST http_classification_info stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
   <!-- server_region_http_stalled_thread_dump_action -->
<!ATTLIST http_classification_info cputimeused_limit CDATA #IMPLIED>
        <!-- server_region_request_cputimeused_limit -->
<!ATTLIST http_classification_info cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
   <!-- server_region_cputimeused_dump_action -->
<!ATTLIST http_classification_info dpm_interval CDATA #IMPLIED>
           <!-- MODIFY [JOBNAME],DPM,HTTP= -->
<!ATTLIST http_classification_info dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
   <!-- server_region_dpm_dump_action -->
<!ATTLIST http_classification_info SMF_request_activity_enabled (0|1) #IMPLIED>
      <!-- server_SMF_request_activity_enabled -->
<!ATTLIST http_classification_info SMF_request_activity_timestamps (0|1) #IMPLIED>
   <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST http_classification_info SMF_request_activity_security (0|1) #IMPLIED>
      <!-- server_SMF_request_activity_security -->
<!ATTLIST http_classification_info SMF_request_activity_CPU_detail (0|1) #IMPLIED>
   <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST http_classification_info classification_only_trace (0|1) #IMPLIED>
<!ATTLIST http_classification_info message_tag CDATA #IMPLIED>
<!ATTLIST http_classification_info timeout_recovery (servant|session) #IMPLIED>
      <!-- protocol_http_timeout_output_recovery -->

<!ELEMENT ola_classification_info (ola_classification_info*)>
<!ATTLIST ola_classification_info transaction_class CDATA #IMPLIED>
<!ATTLIST ola_classification_info propagate_transaction_name (true|false) #IMPLIED>
<!ATTLIST ola_classification_info service_name CDATA #IMPLIED>
<!ATTLIST ola_classification_info description CDATA #IMPLIED>
<!ATTLIST ola_classification_info dispatch_timeout CDATA #IMPLIED>
        <!-- control_region_wlm_dispatch_timeout -->
<!ATTLIST ola_classification_info queue_timeout_percent CDATA #IMPLIED>
   <!-- control_region_iiop_queue_timeout_percent -->
<!ATTLIST ola_classification_info request_timeout CDATA #IMPLIED>
         <!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST ola_classification_info stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
 <!-- server_region_iiop_stalled_thread_dump_action -->
<!ATTLIST ola_classification_info cputimeused_limit CDATA #IMPLIED>
        <!-- server_region_request_cputimeused_limit -->
<!ATTLIST ola_classification_info cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
   <!-- server_region_cputimeused_dump_action -->
<!ATTLIST ola_classification_info dpm_interval CDATA #IMPLIED>
           <!-- MODIFY [JOBNAME],DPM,IIOP= -->
```

```
<!ATTLIST ola_classification_info dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_dpm_dump_action -->
<!ATTLIST ola_classification_info SMF_request_activity_enabled (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_enabled -->
<!ATTLIST ola_classification_info SMF_request_activity_timestamps (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST ola_classification_info SMF_request_activity_security (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_security -->
<!ATTLIST ola_classification_info SMF_request_activity_CPU_detail (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST ola_classification_info classification_only_trace (0|1) #IMPLIED>
<!ATTLIST ola_classification_info message_tag CDATA #IMPLIED>


<!ELEMENT SibClassification (sib_classification_info+)>
<!ATTLIST SibClassification type (jmsra|destinationmediation) #REQUIRED>
<!ATTLIST SibClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST SibClassification schema_version CDATA #REQUIRED>
<!ELEMENT sib_classification_info EMPTY>
<!-- inputs -->
<!ATTLIST sib_classification_info selector CDATA #IMPLIED>
<!ATTLIST sib_classification_info bus CDATA #IMPLIED>
<!ATTLIST sib_classification_info destination CDATA #IMPLIED>
<!ATTLIST sib_classification_info discriminator CDATA #IMPLIED>
<!ATTLIST sib_classification_info description CDATA #IMPLIED>
<!-- outputs -->
<!ATTLIST sib_classification_info transaction_class CDATA #REQUIRED>
<!ATTLIST sib_classification_info dispatch_timeout CDATA #IMPLIED>
        <!-- control_region_wlm_dispatch_timeout -->
<!ATTLIST sib_classification_info queue_timeout_percent CDATA #IMPLIED>
  <!-- control_region_iiop_queue_timeout_percent -->
<!ATTLIST sib_classification_info request_timeout CDATA #IMPLIED>
        <!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST sib_classification_info stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
 <!-- server_region_iiop_stalled_thread_dump_action -->
<!ATTLIST sib_classification_info cputimeused_limit CDATA #IMPLIED>
        <!-- server_region_request_cputimeused_limit -->
<!ATTLIST sib_classification_info cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_cputimeused_dump_action -->
<!ATTLIST sib_classification_info dpm_interval CDATA #IMPLIED>
          <!-- MODIFY [JOBNAME],DPM,IIOP= -->
<!ATTLIST sib_classification_info dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_dpm_dump_action -->
<!ATTLIST sib_classification_info SMF_request_activity_enabled (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_enabled -->
<!ATTLIST sib_classification_info SMF_request_activity_timestamps (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST sib_classification_info SMF_request_activity_security (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_security -->
<!ATTLIST sib_classification_info SMF_request_activity_CPU_detail (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST sib_classification_info classification_only_trace (0|1) #IMPLIED>
<!ATTLIST sib_classification_info message_tag CDATA #IMPLIED>


<!ELEMENT WMQRAClassification (wmqra_classification_info+)>
<!ATTLIST WMQRAClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST WMQRAClassification schema_version CDATA #REQUIRED>
<!ELEMENT wmqra_classification_info EMPTY>
<!-- inputs -->
<!ATTLIST wmqra_classification_info selector CDATA #IMPLIED>
<!ATTLIST wmqra_classification_info queue_manager CDATA #IMPLIED>
<!ATTLIST wmqra_classification_info destination CDATA #IMPLIED>
<!ATTLIST wmqra_classification_info description CDATA #IMPLIED>
<!-- outputs -->
<!ATTLIST wmqra_classification_info transaction_class CDATA #REQUIRED>
<!ATTLIST wmqra_classification_info dispatch_timeout CDATA #IMPLIED>
        <!-- control_region_wlm_dispatch_timeout -->
<!ATTLIST wmqra_classification_info queue_timeout_percent CDATA #IMPLIED>
  <!-- control_region_iiop_queue_timeout_percent -->
<!ATTLIST wmqra_classification_info request_timeout CDATA #IMPLIED>
        <!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST wmqra_classification_info stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
 <!-- server_region_iiop_stalled_thread_dump_action -->
<!ATTLIST wmqra_classification_info cputimeused_limit CDATA #IMPLIED>
        <!-- server_region_request_cputimeused_limit -->
<!ATTLIST wmqra_classification_info cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_cputimeused_dump_action -->
<!ATTLIST wmqra_classification_info dpm_interval CDATA #IMPLIED>
          <!-- MODIFY [JOBNAME],DPM,IIOP= -->
<!ATTLIST wmqra_classification_info dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_dpm_dump_action -->
<!ATTLIST wmqra_classification_info SMF_request_activity_enabled (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_enabled -->
<!ATTLIST wmqra_classification_info SMF_request_activity_timestamps (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST wmqra_classification_info SMF_request_activity_security (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_security -->
<!ATTLIST wmqra_classification_info SMF_request_activity_CPU_detail (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST wmqra_classification_info classification_only_trace (0|1) #IMPLIED>
<!ATTLIST wmqra_classification_info message_tag CDATA #IMPLIED>
```

Example classification of an HTTP inbound request

Use the following http request with the previously described sample z/OS workload classification document containing RAS attributes:

http://my.server.com:8080/PlantsByWebSphere/plants/newOrder.html

Because this request is an HTTP request, the application server scans only the http_classification_info elements. The application server scans the elements in the order that they occur in the workload classification file.

The application server finds the following http_classification_info element first in the sample z/OS workload classification document that contains RAS attributes:

```
<http_classification_info
  host="my.server.com"
  port="8080"
  transaction_class="HTC8080"
  dispatch_timeout="100"
  queue_timeout_percent="98"
  timeout_recovery="session"
  stalled_thread_dump_action="javacore">
```

This element has input attributes and values of `host="my.server.com"` and `port="8080"`. Since these attributes match the inbound HTTP request, the application server descends into this element and compares the child nodes. The application server finds the following child element:

```
<http_classification_info
  uri="/PlantsByWebSphere/*"
  message_tag="plantsbw">
```

The child element defines the input attribute and value of `uri="/PlantsByWebSphere/*"`, which matches the URI of the inbound request URI. The application server then scans the children of the element. The application server finds the first child element:

```
<http_classification_info
  uri="*.jpg"
  transaction_class="HTCPjpg"
  dispatch_timeout="10"
/>
```

This child element contains the `uri="*.jpg"` attribute and value. Since this attribute-value pair does not match the input URI, the application server moves to the next child element. The application server finds the second child element:

```
<http_classification_info
      uri="*.html"
      transaction_class="HTChtml"
    />
```

This child element contains the `uri="*.html"` attribute and value, which matches the input URI.

Since no further child elements exist, the classification element classifies the request. The application server assigns the request all the output attributes from this classification element and all its ancestor elements. The following attribute-value pairs are a complete list of output attribute-value pairs that the application server assigns to the request:

```
dispatch_timeout="100"
  queue_timeout_percent="98"
  timeout_recovery="session"
  stalled_thread_dump_action="javacore"
  message_tag="plantsbw"
  transaction_class="HTChtml"
```

The application server reads any RAS attributes not defined in the workload classification file from the server-wide configuration and assigns them to the request. The relevant server-wide configuration properties, including which ones the classification data overrode, are in the following list:

```
protocol_http_timeout_output                  -- overridden by dispatch_timeout
control_region_http_queue_timeout_percent     -- overridden by queue_timeout_percent
com.ibm.CORBA.RequestTimeout
```

```
server_region_http_stalled_thread_dump_action    -- overridden by stalled_thread_dump_action
server_region_request_cputimeused_limit
server_region_cputimeused_dump_action
server_region_dpm_dump_action
server_SMF_request_activity_enabled
server_SMF_request_activity_timestamps
server_SMF_request_activity_CPU_detail
protocol_http_timeout_output_recovery            -- overridden by timeout_recovery
```

For any server-wide configuration properties that the classification data does not override, the request inherits the server-wide property value.

Example classification of an IIOP inbound request

Use the following IIOP request with the previously described sample z/OS workload classification document containing RAS attributes:

IIOP inbound request for MyEJB2bBean.someMethod() in module MyEJB2b.jar from application EJBApp2.

Because the inbound request is an IIOP request, the application server scans only the iiop_classification_info elements. The application server finds the following iiop_classification_info element first in the sample z/OS workload classification document that contains RAS attributes:

```
<iiop_classification_info
    application_name="EJBApp1"
    transaction_class="TC1"
    message_tag="EJBApp1">
```

This element has an input attribute and value of `application_name="EJBApp1"`. The application name of EJBApp1 on the application_name attribute does not match the application name of EJBApp2 from the inbound request. Therefore, the application server skips this classification element and all of its child elements, and moves to the next element. The application server finds the next element:

```
<iiop_classification_info
    application_name="EJBApp2"
    dispatch_timeout="15"
    message_tag="EJBApp2">
```

This element has an input attribute and value of `application_name="EJBApp2"`. Because this attribute-value pair matches the application name of the inbound request, the scanner descends into this element. The application server finds the first child element:

```
<iiop_classification_info
    module_name="MyEJB2a.jar"
    transaction_class="TC2a"
/>
```

This element contains the input attribute and value of `module_name="MyEJB2a.jar"`. Because this attribute-value pair does not match the MyEJB2b.jar module name of the inbound request, the application server scans the next child element:

```
<iiop_classification_info
    module_name="MyEJB2b.jar"
    transaction_class="TC2b"
/>
```

This element contains the attribute and value of `module_name="MyEJB2b.jar"`. The module name of MyEJB2b.jar on the module_name attribute matches the module name of the inbound request. Since for no further child elements exist, the application server classifies the request to this element. The request inherits all the output attributes from this element and all the ancestor elements. The following attribute-value pairs are a complete list of output attribute-value pairs that the application server assigns to the request:

```
dispatch_timeout="15"
message_tag="EJBApp2"
transaction_class="TC2b"
```

The application server reads any RAS attributes not defined in the workload classification file from the server-wide configuration and assigns them to the request. The relevant server-wide configuration properties, including which ones the classification data overrode, are in the following list:

```
control_region_wlm_dispatch_timeout              -- overridden by dispatch_timeout
control_region_iiop_queue_timeout_percent
com.ibm.CORBA.RequestTimeout
server_region_iiop_stalled_thread_dump_action
server_region_request_cputimeused_limit
server_region_cputimeused_dump_action
server_region_dpm_dump_action
server_SMF_request_activity_enabled
server_SMF_request_activity_timestamps
server_SMF_request_activity_CPU_detail
```

For any server-wide configuration properties that the classification data does not override, the request inherits the server-wide property value.

Using transaction classes to classify workload for WLM

You can use transaction classes to classify client workload for workload management (WLM). The workload that WLM manages consists of different transactions that are targeted to separate servants, each with goals defined by specific service classes. The service classes chosen also determines the WLM goal when Java Garbage Collection (GC) is running, which can be CPU intensive. You do not want to set a servant higher in the service class hierarchy than more important work such as production WebSphere, CICS, or IMS transaction servers.

Before you begin

gotcha: Transaction class mapping file support is deprecated. You should use a workload classification document instead of a transaction class mapping file to classify work requests in a z/OS environment.

You must define the service objectives (goals) for your service classes. You must also define the service objectives of your servers. For more information about defining service objectives (goals) for each service class, see the *z/OS MVS Planning: Workload Management* book, SA22-7602, for example at http://publibz.boulder.ibm.com/epubs/pdf/iea2w131.pdf, or the z/OS WLM web page at http://www.ibm.com/servers/eserver/zseries/zos/wlm/.

gotcha: You do not have to define special classification rules and work qualifiers initially. However, they should be defined before this system becomes a production system.

About this task

Each transaction is dispatched in its own WLM enclave in a servant process, and is managed according to the goals of its service class. The service class chosen also determines the WLM goal when Java Garbage Collection (GC) is running, which can be CPU intensive.

You should classify the servants to a high STC importance service class so that they are initialized quickly when WLM determines they are needed. However, you do not want to set a servant higher in the service class hierarchy than more important work such as CICS, or IMS transaction servers.

Controllers perform some processing as they receive work into the system, manage the transport handlers, classify a work item, and handle housekeeping tasks. Therefore, controllers should also be classified a high STC importance service class.

You can use the WLM CB-type classification criteria to classify work items:
- Server name (CN)
- Server instance name (SI)
- User ID assigned to the transaction (UI)
- Transaction class (TC)

To classify work using server and userid criteria, use a combination of the WLM Workload Classification rules in the WLM ISPF dialog panels. For more information about defining WLM Classification rules, see Workload management (WLM) and its related article that includes an example of classification rules.

To classify work using transaction classes, you define and use transaction class mappings, as described in this task. The following steps are used to classify work using transaction classes:

Procedure

1. Define transaction class mappings based on the HTTP virtual host name, port number, and URI (Universal Resource Identifier - encoded address for any resource on the Web) provided with each work HTTP or HTTPS request.

   a. Create a Transaction Class mapping file (as a simple text file). For example: `/wasconfig/t5was/ MyTrMapFile.txt`

   Important: This file must be in EBCDIC format.

   b. Edit the Transaction Class mapping file to define each transaction class mapping that you want to use. Define each mapping on a separate line, using the following syntax:

   `TransClassMap host:port uritemplate tclass`

   Note: You can use wildcard characters for the host and port fields fi you use them for both fields. For example:

   ```
   TransClassMap wsc4.washington.ibm.com:9080   /MyIVT/index.*   TCLMYIVT
   TransClassMap wsc4.washington.ibm.com:9080   /MyIVT/ivtejb    TCLMYEJB
   TransClassMap wsc4.washington.ibm.com:*      /SuperSnoop*     TCLSNOOP
   TransClassMap wsc4.washington.ibm.com:*      /ssb/*           TCLSSB
   TransClassMap *:*                            /admin*          TCLADMIN
   ```

2. Specify the Transaction Class mapping file on the administrative properties for each server that is to handle work classified by transaction class. To specify the Transaction Class mapping file for a server:

   a. In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*. Then, in the Container Settings section, click **Container settings > Web container**.

   b. In the Additional Properties section , click **z/OS additional settings**.

   c. In the **Transaction Class Mapping** field, the fully qualified name of the Transaction Class mapping file that you edited in a previous step. For example: `/wasconfig/t5was/MyTrMapFile.txt`

   d. If you want to use a transaction class to classify outbound data that is delivered in response to HTTP and HTTPS requests, select the TCLASS option in the **Network QoS** field. If you specify TCLASS, the product uses the transaction class value that was used to classify the inbound request to the z/OS Workload Manager.

Example

The following table shows classification rules for STC-type work that covers the controller and servant regions started tasks:

```
        --------Qualifier--------            -------Class--------
Action   Type    Name    Start            Service    Report
                                DEFAULTS: OPS_DEF     _____
_____  1 TN     %%DMN    ___             OPS_HIGH   RWSDMN
_____  1 TN     T5SRV*   ___             OPS_MED    RT5SRV
_____  1 TN     WS%%%%   ___             SYSSTC     RWSCTLR
_____  1 TN     WS%%%%S  ___             OPS_HIGH   RWSSRVR
```

The following table shows classification rules for CB-type work in which the default service class is WSMED and has a reporting class of RWSDEFLT. Work run in the WSPROD server is classified as WSMED with a reporting class of RWSPROD, unless it has a transaction class of TCLASS1, TCLASS2, or TCLASS2 assigned through the transaction class mapping file below.

```
Qualifier  Qualifier Start      Service  Report
# type     name      position   Class    Class
- --------- -------- --------   -------- --------
                     Default: WSMED    RWSDEFLT
1 CN       WSPROD    1          WSMED    RWSPROD
2 . TC     . TCLASS1            WSFAST   RWSPRD1
2 . TC     . TCLASS2            WSMED    RWSPRD2
2 . TC     . TCLASS5            WSSLOW   RWSPRD5
1 CN       WSTEST    1          WSSLOW   RTSTEST
2 . UI     . USER1              WSMED    RTSTSTU2
2 . TC     . TCLASS5            WSSLOW   RTSTST5
```

The following table shows how work can be assigned a transaction class based on its host name, port number, or URI. For example, a web request of http://ibm.com:80/Webap1/myservlet handled by the WSPROD server would be assigned a transaction class of TCLASS1, a service class of WSFAST, and a reporting class of RWSPRD1 by the classification rules shown above.

```
TransClassMap www.ibm.com:80 /Webap1/myservlet TCLASS1
TransClassMap www.ibm.com:* /Webap1/myservlet TCLASS2
TransClassMap *:443 * TCLASS3
TransClassMap *:* /Webap1/myservlet TCLASS4
TransClassMap www.ibm.com:* /Webap5/* TCLASS5
TransClassMap * * TCLASS6
```

Example of the application of classification rules.

In this example, all work for BBOC001, except for work running under the user ID DBOOZ, gets classified as CBFAST. Work for DBOOZ gets classified as CBSLOW. All other work, such as work coming from clients outside the cell and including the work for the product runtime servers, gets classified as CBCLASS.

For the purpose of this example, let us assume you have three workload management service classes defined for the product(subsystem type CB):
1. CBFAST-designed for transactions requiring fast response times.
2. CBSLOW-designed for long-running applications that do not require fast response times.
3. CBCLASS-designed for remaining work requests.

You design a client workload called BBOC001 that requires fast response times. Also, you want to give work that runs under your manager's user ID (DBOOZ) slower response times. Finally, all remaining work requests should run under the default service class, CBCLASS.

Table 61. Example for classifying work.   The table includes the type, name, service, and goal for the work.

Type column	Name column	Service column	Goal
CN	BBOC001	CBFAST	90% complete in 2 seconds
UI	DBOOZ	CBSLOW	Velocity 50, importance = 3
(default)	(blank)	CBCLASS	Discretionary

You could set the following performance goals through IWMARIN0:
1. Issue IWMARIN0 and choose option 4:

```
   File  Utilities  Notes  Options  Help
   ------------------------------------------------------------------
   Functionality LEVEL003   Definition Menu  WLM Appl LEVEL004    Command ===>
   _____

   Definition data set  . . : 'CB.MYCB.WLM'
   Definition name  . . . . . CB390      (Required)
   Description  . . . . . . . WLM Setup for the product
   Select one of the following options. . . . . 4__
   1.  Policies
   2.  Workloads
```

```
           3.  Resource Groups
           4.  Service Classes
           5.  Classification Groups
           6.  Classification Rules
           7.  Report Classes
           8.  Service Coefficients/Options
           9.  Application Environments
          10.  Scheduling Environments
```
2. Create a service class called CBFAST and specify that it be 90% complete in 2 seconds.

   Note: The example assumes you have defined a workload called ONLINE.

```
        Service-Class  Notes  Options  Help
--------------------------------------------------------------------------
Create a Service Class
Row 1 to 2 of 2    Command ===> _____
Service Class Name . . . . . . CBFAST    (Required)
Description . . . . . . . . . Quick CB transactions
Workload Name  . . . . . . . . ONLINE    (name or ?)
Base Resource Group  . . . . . _____   (name or ?)
Specify BASE GOAL information.  Action Codes: I=Insert new period,
E=Edit period, D=Delete period.
---Period---  ---------------------Goal---------------------
Action  #  Duration   Imp.  Description

__                                                                      __
1              1    90% complete within 00:00:02.000
 ****************************** Bottom of data ******************************


  .----------------------------------------------------------------------.
  | Press EXIT to save your changes or CANCEL to discard them. (IWMAM970) |
  '----------------------------------------------------------------------'
```
3. Save the service class. You see the following:

```
        Service-Class  View  Notes  Options  Help
--------------------------------------------------------------------------
Service Class Selection List
Row 1 to 14 of 21   Command ===> _____
Action Codes: 1=Create, 2=Copy, 3=Modify, 4=Browse, 5=Print, 6=Delete,
/=Menu Bar
Action  Class
Description
Workload
__    CBFAST
 Quick CB Transactions
ONLINE
****************************** Bottom of data ******************************
```
4. Repeat these steps for the CBSLOW service class.
5. Create classification rules using the new service class. Choose option 6 on the main panel:

```
        File  Utilities  Notes  Options  Help
--------------------------------------------------------------------------
Functionality LEVEL003          Definition Menu        WLM Appl LEVEL004
Command ===> _____
Definition data set  . . : 'CB.MYCB.WLM'
Definition name  . . . . . CB390      (Required)
Description  . . . . . . . WLM Setup for the product
Select one of the following options. . . . . 6__
 1.  Policies
 2.  Workloads
 3.  Resource Groups
 4.  Service Classes
 5.  Classification Groups
 6.  Classification Rules
 7.  Report Classes
 8.  Service Coefficients/Options
 9.  Application Environments
10.  Scheduling Environments
```
6. Create a set of rules for your service classes:

```
        Subsystem-Type Xref  Notes  Options  Help
   ------------------------------------------------------------------------
   Create Rules for the Subsystem Type        Row 1 to 2 of 2
   Command ===> _____    SCROLL ===> PAGE
   Subsystem Type . . . . . . . . CB    (Required)
   Description  . . . . . . . . .  WebSphere  classification
   Fold qualifier names?  . . . . Y  (Y or N)
   Action codes:  A=After    C=Copy        M=Move      I=Insert rule
   B=Before    D=Delete row   R=Repeat    IS=Insert Sub-rule
    -------Qualifier-------------
   -------Class--------
   Action    Type        Name     Start
   Service      Report
   DEFAULTS: CBCLAS        _____
   ____   1  CN
   BBOC001     ___
   CBFAST      _____
   ____   1  UI
   DBOOZ       ___
   CBSLOW      _____
   *************************** BOTTOM OF DATA ***************************
```

Transaction class mapping file entries

Transaction class mapping file entries indicate the workload management (WLM) goal for each class of client work. Each client transaction is dispatched in its own WLM enclave in a servant region process, and is managed according to the goals specified for its service class.

depfeat: Transaction class mapping file support is deprecated. You should use a workload classification document instead of a transaction class mapping file to classify work requests.

Following is the syntax for entries in a transaction class mapping file:

```
TransClassMap host:port uritemplate tclass
```

where:

host   Is the value compared against the hostname of the HOST: header of the request.

>   **gotcha:** You cannot use wild-card characters in the host field unless you use it for the entire field; for example *:*.

port   Is the value compared against the port of the request.

>   **gotcha:** You cannot use wild-card characters in the port field unless you use it for the entire field; for example *:*.

uritemplate
>   Is the value compared against the URI of the request. Any query string will not be used in the comparison. This value can be a wildcard '*', or end in a wildcard.

tclass   Is the Workload Manager Transaction Class name that will be used in the creation of the enclave.

Examples:

```
TransClassMap www.ibm.com:80 /webap1/myservlet TCLASS1

TransClassMap www.ibm.com:* /webap1/myservlet TCLASS2

TransClassMap *:443 * TCLASS3

TransClassMap *:* /webap1/myservlet TCLASS4

TransClassMap www.ibm.com:* /webap2/* TCLASS5
```

```
TransClassMap * /myservlet TCLASS6
```

```
TransClassMap * * TCLASS6
```

Controller and Servant WLM classifications

Applications are deployed on a server or a cluster of servers. Each server consists of a controller and one or more servants. Each controller is started by the deployment manager, or by an MVS operator command as an MVS started tasks. Each servant is started by the Workload manager (WLM) as it is needed.

You can use transaction classes to classify client workload for WLM. Transaction classification can be based on the following WLM classification criteria:

- Server name (CN) – if the server is clustered, this value is the short name for the cluster; if the server is not clustered, this value is the value specified on the server custom property, ClusterTransitionName.
- The Server instance name (SI) – this is the short name for the server. This is usually not very useful because you cannot control which server instance runs a transaction within a cluster.
- User ID assigned to the transaction (UI) Transaction class (TC) - use the transaction class mapping file for the server to assign this ID to a transaction

.

Controller classification

You should classify controllers in SYSSTC or give them a high importance and velocity goal, because contollers do some of the processing that is required to receive work into the system, manage the HTTP transport handler, classify the work, and do other housekeeping tasks,

Important: A step in controller start-up procedures, such as BBO7ACR, invoke the BPXBATCH program to check the service levels delivered, applied, and pending, and log the results in the `/properties/service/logs/applyPTF.log` file. Because the BPXBATCH program is classified according the OMVS rules, instead of inheriting the service classification of the startup procedure, on a busy system several minutes might pass before BPXBATA2 gets control. You can minimize the impact of the BPXBATCH step by changing the WLM Workload Classification Rules for OMVS work to a higher service objective. In the following example, OMVS work is assigned a EBIZ_HI service class, which has an importance of 1, and Velocity of 50.

```
Subsystem Type . : OMVS
Description . . . E_Biz Classification Rule
--------Qualifier------- -------Class--------
Action Type Name Start Service Report
DEFAULTS: EBIZ_DEF _____
____ 1 TN FTPSERVE ____ EBIZ_HI _____
____ 1 UI OMVSKERN ____ SYSSTC _____
____ 1 TN WSSRV* ____ EBIZ_HI RPTACR <<==
```

Servant classification

When a servant region starts, WLM classifies the servant region as a started task, and places it in a service class and a report class based on the classification rules that are specified in the WLM policy. WLM uses this service class to determine the priority of the address space. The priority of an address space determines its access to system resources during the initialization of the servant region.

Each work request that the controller region receives is associated with a WLM enclave. Each of these enclaves is then associated with a WLM service class and report class. When WLM distributes the work requests from the controller region to the various servant regions, it tries to keep all of the work requests that are associated with the same service class together. This grouping of requests means that the work requests processed by a given servant region are associated with enclaves that are usually associated with the same service class. If all of the enclaves are associated with the same service class, it can be said that the servant region is associated with that service class.

After the servant region is initialized and starts to receive work requests, all of the dispatch threads and non-dispatch threads, such as the Java Garbage Collection (GC) threads, are managed according to the goals that are associated with the WLM service class of the enclaves associated with the work requests that are executing in the servant.

All of the resources that the dispatch threads consume while an enclave is associated with those threads are reported in the report class associated with the enclave. All of the resources that the threads that are not associated with an enclave consume are reported in the report class associated with servant region started task.

Note: You should make sure that the servant classification is not higher in the service class hierarchy than more important work, such as the controller and CICS, or IMS transaction servers.

WLM Classification Rules for STC-type work

Here is a simple example of the WLM Classification Rules for STC-type work that covers the controller and servant started tasks:

```
        --------Qualifier--------        -------Class--------
Action   Type    Name     Start          Service    Report
                                DEFAULTS: OPS_DEF    _____
_____  1 TN    %%DMN    ___             OPS_HIGH   RWSDMN
_____  1 TN    T5SRV*   ___             OPS_MED    RT5SRV
_____  1 TN    WS%%%%   ___             SYSSTC     RWSCTLR
_____  1 TN    WS%%%%S  ___             OPS_HIGH   RWSSRVR
```

Creating clusters

A cluster is a set of application servers that you manage together as a way to balance workload.

Before you begin

Before you create a cluster:
- Review the content of the topic "Clusters and workload management," especially the information about setting cluster weights.
- Decide if you want enterprise bean requests routed to the node on which the client resides.
- Decide if you want to use HTTP memory-to-memory replication.
- Determine the appropriate configuration settings for the first cluster member. A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.
- Decide on which node you want the first cluster member to reside.

About this task

You might want to create a cluster if you need to:
- Balance your client requests across multiple application servers.
- Provide a highly available environment for your applications.

A cluster enables you to manage a group of application servers as a single unit, and distribute client requests among the application servers that are members of the cluster.

If you plan to create a cluster of servers that spans multiple systems in a sysplex and has stateful session beans with an activation policy of Transaction deployed in them, the passivation directory should reside on an HFS (hierarchical file system) that is shared across the multiple systems in the sysplex on which the clustered servers are running.

To create a cluster:

Procedure

1. In the administrative console, click **Servers > Clusters > WebSphere application server clusters > New**. The Create a new cluster wizard starts.
2. Specify a name for the cluster.
3. Optional: Specify a short name for the cluster.

   For clustered servers, the WLM application environment is the default value for the cluster short name. If you specify a short name for a cluster, the name:

   - Must be one to eight characters in length.
   - Must contain only alpha-numeric or national language characters.
   - Cannot start with a number.
   - Must be unique in the cell.
   - Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

   gotcha: If you specify a short name, make sure that you set up a RACF SERVER class profile that includes this short name.

4. Select **Prefer local** if you want to enable host-scoped routing optimization. This option is enabled by default. When this option is enabled, if possible, EJB requests are routed to the client host. This option improves performance because client requests are sent to local enterprise beans.

   Note: If you enable the preferLocal optimization, the deployment manager must be running to affect the configuration. If the deployment manager is shut down, preferLocal optimization is not performed and requests might be dispersed across all the members of the cluster.

5. Select **Configure HTTP session memory-to-memory replication** if you want a memory-to-memory replication domain created for this cluster. The replication domain is given the same name as the cluster and is configured with the default settings for a replication domain. When the default settings are in effect, a single replica is created for each piece of data and encryption is disabled. Also, the web container for each cluster member is configured for memory-to-memory replication.

   To change these settings for the replication domain, click **Environment > Replication domains > *replication_domain_name***. To modify the web container settings, click **Servers > Clusters > WebSphere application server clusters** > *cluster_name* > **Clusters members >** *cluster_member_name*. Then, in the Container settings section, click **Web container settings > Web container >Session management > Distributed environment settings** in the administrative console. If you change these settings for one cluster member, you might also need to change them for the other members of this cluster.

6. Click **Next**.
7. Choose whether to create an empty cluster or to create the first member of the cluster.

   If you decide to create an empty cluster, to add members to this cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters** > *cluster_name* > **Clusters members > New**.

   To create an empty cluster:

   a. Select **None. Create an empty cluster.**

   b. Click **Next** to display a summary of the defined cluster.

   c. Click **Finish** to create the cluster, or click **Cancel** if you decide not to create this cluster.

   When you create the first cluster member, remember that a copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

   a. Specify the name of the first cluster member.

   b. Select the node on which you want this cluster member to reside.

c. Specify a short name for this cluster member. The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, and RACF.

d. Specify the weight value for the cluster member. The weight value controls the amount of work that is directed to the application server. If the weight value for this server is greater than the weight values that are assigned to other servers in the cluster, then this server receives a larger share of the workload. The weight value represents a relative proportion of the workload that is assigned to a particular application server. The value can range from 0 to 20.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

- For HTTP requests, weights are used to distribute HTTP traffic between the web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.

- For web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.

- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

e. Select **Generate unique HTTP ports** if you want to generate unique port numbers for every HTTP transport that is defined in the source server. When this option is selected, which is the default setting, this cluster member does not have HTTP transports or HTTP transport channels that conflict with any of the other servers that are defined on the same node. If you unselect this option, all of the cluster members will share the same HTTP ports.

f. Select the core group to which you want this cluster member to belong. You are prompted for the core group only if you have more than one core group defined for this cluster.

g. Select one of the following options to determine how the server resources are promoted in the cluster.

- **Cluster** to move the resources of the first cluster member to the cluster level. The resources of the first cluster member replace the resources of the cluster.

- **Server** to maintain the server resources at the new cluster member level. The cluster resources remain unchanged.

- **Both** to copy the resources of the cluster member (server) to the cluster level. The resources of the first cluster member replace the resources of the cluster. The same resources exist at both the cluster and cluster member scopes.

h. Select one of the following options as the basis for the first cluster member.

- Create the member using an application server template.

  If you select the **defaultZOS** template, which is the only one that is listed unless you used the **createServerTemplate** command for the AdminTask object to create additional templates, the first cluster member uses the default port assignments for z/OS. If some of these ports are already defined for use elsewhere in your system, your newly created cluster member might not start, might function incorrectly, or might generate unexpected error messages. Therefore, you must resolve any port conflicts before you start this server.

- Create the member using an existing application server as a template.

- Create the member by converting an existing application server.

  gotcha: You can only add an existing application server to the cluster if you select that server as the first cluster member. You cannot add other existing application servers to that cluster after you create the first cluster member. If you add an existing server to a cluster, the only way to remove that server from the cluster is to delete the server. Therefore, you might want to use the existing server as a template for the first cluster member instead of as the cluster member. If you keep the original application server out of the cluster, you can reuse that server as the template if you need to rebuild the configuration.

8. Click **Next**.

9. Create additional cluster members. Before you create additional cluster members, check the configuration settings of the first cluster member. These settings are displayed at the bottom of the Create additional cluster members panel of the Create a new cluster wizard. For each additional member that you want to create:

   a. Specify a unique name for the member. The name must be unique within the node.

   b. Select the node to which you want to assign the cluster member.

   c. Specify the weight you want given to this member. The weight value controls the amount of work that is directed to the application server. If the weight value for the server is greater than the weight values that are assigned to other servers in the cluster, then the server receives a larger share of the workload. The value can range from 0 to 20.

   d. Specify a short name for this cluster member. The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, and RACF.

   e. Select **Generate unique HTTP ports** if you want to generate unique port numbers for every HTTP transport that is defined in the source server.

   f. Click **Add member**. You can edit the configuration settings of any of the newly created cluster members other than the first cluster member, or you can create additional cluster members. Click **Previous** to edit the properties of the first cluster member. The settings for the first cluster member become the settings for the cluster member template that is automatically created when you create the first cluster member.

10. When you finish creating cluster members, click **Next**.

11. View the summary of the cluster and then click **Finish** to create the cluster, click **Previous** to return to the previous wizard panel and change the cluster, or click **Cancel** to exit the wizard without creating the cluster.

12. To further configure a cluster, click **Servers > Clusters > WebSphere application server clusters >** , and then click the name of the cluster. Only the **Configuration** and **Local Topology** tabs appear until you save your changes.

13. Click **Review** to review your cluster configuration settings. Repeat the previous step if you need to make additional configuration changes.

14. If you do not want to make any additional configuration changes, select Synchronize changes with Nodes and then click **Save**. Your changes are saved and synchronized across all of your nodes.

   gotcha: If you click **Save**, but do not select Synchronize changes with Nodes, when you restart the cluster, the product does not start the cluster servers because it cannot find them on the node. If you want to always synchronize your configuration changes across your nodes, you can select Synchronize changes with Nodes as one of your console preferences.

15. Restart the cluster.

Results

You have created a cluster to which you can assign work requests. The **Runtime** and **Local Topology** tabs appear the next time you access this page.

What to do next

- You can click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* > **Clusters members** in the administrative console, and then click the name of a cluster member to view all of the configuration settings for this cluster member. You can then use this page to change some of the configuration settings for the selected cluster member.

   For example, if you do not need to have all of the cluster member components start during the cluster startup process, you might want to reconfigure the cluster members, such that the **Start components**

as needed is selected. This option is not selected when a new cluster member is created. Selecting this option can improve cluster startup time, and reduce the memory footprint of the cluster members.

gotcha: Before selecting this option, verify that any other WebSphere products, that you are running in conjunction with this product, support this functionality.

depfeat: The default addressing mode for a new server is 64-bit. You can deselect the Run in 64-bit mode field if you need to use 31-bit addressing mode. However, support for running a server in 31-bit mode is deprecated.

- Use the administrative console to view or change the configuration settings for a cluster. For example, if you are running in a high availability environment, you can click **Servers > Clusters > WebSphere application server clusters >** *cluster_name*, and then select the **Enable failover of transaction log recovery** option for this cluster. This option allows the recovery of transactions to failover from one cluster member to another.

- Create additional cluster members.

  If you create a cluster member by converting an existing application server that is a member of a bus, you must migrate the messaging engine in the server to the scope of a cluster. To do this, use the wsadmin command migrateServerMEtoCluster. Do not delete the messaging engine at server scope and recreate it a cluster scope, because those actions prevent the messaging engine from working with previously configured destinations.

- Start the cluster.

- Use scripting to automate the task of creating clusters.

- Create a static routing table to temporarily handle IIOP routing for the cluster if your high availability infrastructure is disabled.

Creating a cluster: Basic cluster settings

Use this page to enter the basic settings for a cluster.

To view this administrative console page, click **Servers > Clusters > WebSphere application server clusters > New**.

Cluster name

Specifies the name of the cluster. The cluster name must be unique within the cell.

Short name

Specifies the short name for this cluster. This field only appears if you are running on z/OS.

The short name is used as the WLM APPLENV name for all servers that are part of this cluster.

If you specify a short name for a cluster member, the name:
- Must be one to eight characters in length
- Must contain only uppercase alphanumeric characters
- Cannot start with a number
- Must be unique in the cell

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Configure HTTP session memory-to-memory replication

Specifies that when the cluster is created, a memory-to-memory replication domain is created for each of the members of this cluster.

If a replication domain is created, it is given the same name as the cluster and is configured with the default settings for a replication domain. When the default settings are in effect, a single replica is created for each piece of data and encryption is disabled.

Also, if a replication domain is created, the web container for each cluster member is configured for memory-to-memory replication.

Starting with Version 7.0.0.5, if a replication domain is created, the SIP container and web container for each cluster member is configured for memory-to-memory replication.

To modify the replication domain settings, in the administrative console, click **Environment > Replication domains >** *replication_domain_name*.

The default mode setting for the replication domain is `Both client and server`. In this mode, all data sent to either the client or the server is replicated. This setting is good for an environment that has a middle to low traffic load. However, if your environment has a high traffic load, you should change the replication domain mode setting to either `Client only` , or `Server only`, because these settings provide better scaling. In `Client only` mode, only data sent to the client is replicated. In `Server only` mode, only data sent to the server is replicated.

To modify the mode setting for a replication domain:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*.
2. Under Container settings, click **Web container settings**, and then click **Session management > Distributed environment settings > Memory-to-memory replication**.

gotcha: If you change any of the replication domain settings for one cluster member, including the mode setting, you should change them for all of the other members of the cluster.

Creating a cluster: Create first cluster member

Use this page to specify settings for the first cluster member.

There are two ways to create the first member of a cluster:
- You can create the first member when you create a cluster.

  To create a cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > New**.
- You can create an empty cluster and then add a first member after you finish creating the cluster.

  To create a cluster member for an existing cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Clusters members > New**.

When you create the first cluster member, a copy of that member is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

When adding servers to a cluster, remember that the only way to remove an application server from a cluster is to delete the application server from the list of cluster members.

Member name

Specifies the name of the application server that is created for the cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the application server resides.

Short name
Specifies the short name for this cluster member. This field only applies to the z/OS platform.

The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster member, the name:
- Must be one to eight characters in length. By default, WebSphere Application Server for z/OS assumes that you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only uppercase alphanumeric characters
- Cannot start with a number.
- Must be unique in the cell
- Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Weight
Specifies the amount of work that is directed to the application server.

If the weight value for the server is greater than the weight values that are assigned to other servers in the cluster, the server receives a larger share of the cluster workload. The value can range from 0 to 20. Enter zero to indicate that you do not want requests to route to this application server unless this server is the only server that is available to receive requests.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system.
- For HTTP requests, weights are used to distribute HTTP traffic between the web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
- For web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Core group
Specifies the core group in which the application server resides. This field displays only if you have multiple core groups configured. You can change this value only for the first cluster member.

Generate unique HTTP ports
Specifies that a unique HTTP port is generated for the application server. By generating unique HTTP ports for the application server, you avoid potential port collisions and configurations that are not valid.

Select how the server resources are promoted in the cluster
Specifies how resources such as data sources and schedulers are initially created in the cluster. Create settings at the cluster scope if the settings can be used across the entire cluster. Otherwise, create settings at the cluster member (server) level. Creating settings at the cluster scope reduces the amount of configuration that you maintain.

Attention: When you create a cluster, this setting is disabled if you select **None. Create an empty cluster** for the **Select basis for first cluster member** setting. Otherwise, this setting is enabled.

The list has the following options.

Cluster

Moves the resources of the first cluster member to the cluster level. The resources of the first cluster member replace the resources of the cluster. Move resources of the first cluster member when you want to promote the resources of a server to the cluster. Promoting resources is the recommended option.

The Cluster option is the default behavior in the administrative console.

Server

Maintains the server resources at the new cluster member level and does not change the cluster level resources.

The Server option is most useful in the following situations:

- When you want different configuration settings for resources defined on each cluster member.
- When you have deleted all of your cluster members and want to create new cluster members. However, you do not want to replace the cluster scoped resources when you create the cluster members.

Both Copies the resources of the cluster member (server) to the cluster level. The resources of the first cluster member replace the resources of the cluster. The same resources exist at both the cluster and cluster member scopes.

Important: These options are available only for the first cluster member. All other members of a cluster are based on the cluster member template which is created from the first cluster member.

Select basis for first cluster member

Specifies the basis you want to use for the first cluster member.

- If you select **Create the member using an application server template**, the settings for the new application server are identical to the settings of the application server template you select from the list of available templates.

  If you select the defaultZOS template, which is the only one that is listed unless you used the **createApplicationServerTemplate** command for the AdminTask object to create additional templates, the first cluster member uses the default port assignments for the z/OS platform. If some of these ports are already defined for use elsewhere in your z/OS system, your newly created cluster member might not start, might function incorrectly, or might generate unexpected error messages. Therefore, you must resolve any port conflicts before you start this server.

- If you select **Create the member using an existing application server as a template**, the settings for the new application server are identical to the settings of the application server you select from the list of existing application servers.

- If you select **Create the member by converting an existing application server**, the application server you select from the list of available application servers becomes a member of this cluster.

- If you select **None. Create an empty cluster**, a new cluster is created but it does not contain any cluster members.

Important: The basis options are available only for the first cluster member. All other members of a cluster are based on the cluster member template which is created from the first cluster member.

Creating a cluster: Summary settings

Use this administrative console page to view and save settings when you create a cluster or cluster member.

You can view this administrative console page whenever you create a new cluster or a new cluster member. This summary page displays your configuration changes before you commit the changes and the new cluster or cluster member is created.

To create a cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > New**.

To create a cluster member for an existing cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Clusters members > New**

The bounding node group of the cluster is based on the first application server that is added as a member of the cluster. The settings for this first member become the settings for the cluster member template that is then used to create additional cluster members. To select a different bounding node group, in the administrative console, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Clusters members > New**, and then select the appropriate node group for the new cluster member. When you select a different node group, another cluster member template is automatically created for that node group, if one does not already exist.

Review the changes to your configuration, and then click **Finish** to complete and save your work.

Creating a cluster: Create additional cluster members

Use this page to create additional members for a cluster. You can add a member to a cluster when you create the cluster or after you create the cluster. A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

To add members to a cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Cluster members > New**. After you enter the required information about the new cluster member, click **Add Member** to add this member to the cluster member list.

After adding a cluster member, you might need to change one or more of the property settings for this cluster member, or another cluster member that you just added. To change one or more property settings for any cluster member that you just added, other than the first cluster member, select that cluster member, and then click **Edit**. When you finish changing the property settings, click **Update Member** to save your changes.

If you decide not to create a particular cluster member, select the member and then click **Delete**.

You cannot edit or delete the first cluster member or an already existing cluster member.

If you create additional cluster members immediately after you create the first cluster member, the list of cluster members includes a checklist in front of the names of these additional cluster members. However, a check box does not appear in front of the name of the first cluster member because you cannot delete this member or edit its settings. To modify the first cluster member, click **Previous**.

Similarly, if you are adding cluster members to a cluster that already has existing members, the existing members appear in the list of cluster members but a check box does not appear in front of the names of these cluster members. To delete one of these existing members or to change the settings of one of these cluster members, in the administrative console click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Cluster members**, and then select the member that you want to delete or whose configuration settings you want to change.

Member name

Specifies the name of the application server that is created for the cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the application server resides.

Short name

Specifies the short name for this cluster member. This field only displays if you are running on z/OS.

The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster member, the name:
- Must be one to eight characters in length. By default, the product assumes that you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only uppercase alphanumeric characters
- Cannot start with a number
- Must be unique in the cell
- Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Weight

Specifies the amount of work that is directed to the application server.

If the weight value for the server is greater than the weight values that are assigned to other servers in the cluster, the server receives a larger share of the cluster workload. The value can range from 0 to 20. Enter zero to indicate that you do not want requests to route to this application server unless this server is the only server that is available to receive requests.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system.
- For HTTP requests, weights are used to distribute HTTP traffic between the web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
- For web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Generate unique HTTP ports

Specifies that a unique HTTP port is generated for the application server. By generating unique HTTP ports for the application server, you avoid potential port collisions and configurations that are not valid.

Server cluster collection

Use this page to view information about and change configuration settings for a cluster. A cluster consists of a group of application servers. If one of the application servers fails, requests are routed to other members of the cluster.

To view this administrative console page, click **Servers > Clusters > WebSphere application server clusters**.

To define a new cluster, click **New** to start the Create a new cluster wizard.

Name

Specifies a logical name for the cluster. The name must be unique among clusters within the containing cell.

Status

This field indicates whether the cluster is partially started, started, partially stopped, stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

After you click **Start** or **Ripplestart** to start a cluster, each server that is a member of that cluster launches if it is not already running. When the first member launches, the status of the cluster changes to `partially started`. The status remains `partially started` until all cluster members are running. When all cluster members are running, the state changes to `running` and the status of the cluster changes to `started`. Similarly, when you click **Stop** or **ImmediateStop** to stop a cluster, the status changes to `partially stopped` when the first member stops, and then changes to `stopped` when all cluster members are not running.

Table 62. Status and meaning.   The following table describes the status icons.

	Started	The server is running.
	Partially started	The server is in the process of changing from a stopped state to a started state.
	Partially stopped	The server is in the process of changing from a started state to a stopped state.
	Stopped	The server is not running.
	Unknown	The server status cannot be determined.

Server cluster settings

Use this page to view or change the configuration of a server cluster instance, and to view the local topology of a server cluster instance.

To change the configuration and local topology of a server cluster, in the administrative console click **Servers > Clusters > Clusters >***cluster_name*.

To view runtime information, such as the state of the server cluster, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name*, and then click the **Runtime** tab.

To display the topology of a specific cluster, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name*, and then click the **Local Topology** tab.

If the high availability infrastructure is disabled and you require IIOP routing capabilities, follow the instructions contained in the "Enabling static routing for a cluster" topic to create a static route table. This table enables the cluster to handle IIOP requests.

gotcha:

- Because the information contained in the static route table does not account for server runtime state, you should only use this option if the high availability infrastructure is disabled.

  Use of a static route table preempts the use of the dynamic routing table that is contained in cluster members. After the static file is transferred to a node, whenever a cluster member residing in that node starts, that cluster member uses the static table instead of the dynamic

table to handle IIOP routing. If a cluster member is running when you create the static route table, then you must restart that cluster member to give that cluster member access to the static route table information, because the table content is loaded at run time.

- After the table is created, an informational message, similar to the following message, is issued that indicates the name of the file that contains the table and where that file is located:

```
The route table for cluster MyCluster was exported to file
/home/myInstall/was/server/profiles/dmgrProfile/config/cells/
    MyCell/clusters/Myfile.wsrttbl.
```

As this message indicates, the file containing the static route table is placed in the config directory of the deployment manager for this cluster. Keep a record of this location so that you can delete this file when you are ready to start using dynamic routing again.

- If you set up a static route table, then you must statically set the ORB_LISTENER_ADDRESS port on each of the cluster members because the route table is static, and the cluster members do not communicate during state changes. If this port is not assigned, then the cluster members restart on different ports, and the static routing information is not able to route requests to the cluster members.

Cluster name:

Specifies a logical name for the cluster. The name must be unique among clusters within the containing cell.

Short name:

Specifies the short name for this cluster. This field displays only if you are running on z/OS.

The short name is used as the WLM APPLENV name for all servers that are part of this cluster.

If you specify a short name for a cluster member, the name:
- Must be one to eight characters in length
- Must contain only uppercase alphanumeric characters
- Cannot start with a number
- Must be unique in the cell

If you do not specify a short name, then the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Unique Id:

Specifies the unique ID of this cluster.

The unique ID property is read only. The system automatically generates the value.

Bounding node group name:

Specifies the node group that forms the boundaries for this cluster. All application servers that are members of a cluster must be on nodes that are members of the same node group.

A node group is a collection of application server nodes. A node is a logical grouping of managed servers, usually on a system that has a distinct IP host address. All application servers that are members of a cluster must be on nodes that are members of the same node group. Nodes that are organized into a node group need enough capabilities in common to ensure that clusters formed across the nodes in the node group can host the same application in each cluster member. A node must be a member of at least one node group and can be a member of more than one node group.

Create and manage node groups by clicking **System administration > Node groups** in the administrative console.

Enable failover of transaction log recovery:

Specifies that for the transaction service component, failover of the transaction log for recovery purposes is enabled or disabled. The default is disabled.

When this setting is enabled, and the transaction service properties required for peer recovery of failed application servers in a cluster are properly configured, failover recovery of the transaction log occurs if the server processing the transaction log fails. If the transaction services properties required for peer recovery of failed application servers in a cluster are not properly configured, then this setting is ignored.

State:

Specifies whether the cluster is stopped, starting, or running.

If all cluster members are stopped, the cluster state is `stopped`. After you request to start a cluster, the cluster state briefly changes to `starting` and each server that is a member of that cluster launches, if it is not already running. When the first member launches, the state changes to `websphere.cluster.partial.start`. The state remains `partially started` until all cluster members are running, then the state changes to `running`. Similarly, when stopping a cluster, the state changes to `partially stopped` as the first member stops and changes to `stopped` when all members are not running.

Valid values	`starting`, `partially started`, `running`, `partially stopped`, or `stopped`.

Cluster topology

Use this page to display, in a tree format, a list of all of the application server clusters defined for your WebSphere Application Server environment. The list shows all of the nodes and cluster members that are included in each cluster contained in a cell.

To view this page, in the administrative console, click **Servers > Clusters > Cluster topology**.

Enabling static routing for a cluster

If your high availability infrastructure is disabled and you require IIOP routing capabilities, you can create a static routing table for the members of a cluster to use to handle enterprise bean requests. Because the information contained in this static routing table does not account for server runtime state, you should delete this table and return to using the dynamic routing table as soon as your high availability infrastructure is enabled.

Before you begin

Before you create a static route table, ensure that:
- The ORB_LISTENER_ADDRESS port is set to a non-zero value on each of the cluster members. Because the route table you create is static, and the cluster members do not communicate during state changes, if you do not set the ORB_LISTENER_ADDRESS port on each of the cluster members, the cluster members might restart on different ports, and IIOP requests will not be routed correctly.

  To change the value specified for the ORB_LISTENER_ADDRESS port:
  1. In the administrative console, click **Servers > Server Types > WebSphere application servers >** *server_name*, and then under Communications, click **Ports** .
  2. Click **ORB_LISTENER_ADDRESS** in the Port name field.
  3. Change the value specified for the Port field to a value that is greater than 0.

- Each cluster member is started and can use these new non-zero ORB_LISTENER_ADDRESS port values to correctly route IIOP requests.

About this task

You should only create a static route table if your high availability infrastructure is disabled and you require IIOP routing capabilities. To create a static route table:

Procedure

1. Start the wsadmin tool if it is not already running.
2. Identify the cluster managed bean (MBean) for the cluster for which you are creating the route table, and assign that MBean to a variable.
   - Using Jacl:
     ```
     cluster = AdminControl.completeObjectName('cell=
       cell_name,type=Cluster,name=cluster_name,*')
     print cluster
     ```
   - Using Jython:
     ```
     set cluster [$AdminControl completeObjectName cell=
         cell_name,type=Cluster,name=cluster_name,*]
     puts $cluster
     ```
   These commands return the name of the cluster MBean for the specified cluster. For example, for cluster cluster1, the output from these commands will be similar to the following message:
   ```
   WebSphere:cell=mycell,name=cluster1,mbeanIdentifier=Cluster,type=
       Cluster,process=cluster1
   ```
3. Export the route table.
   - Using Jacl:
     ```
     $AdminControl invoke $cluster exportRouteTable
     ```
   - Using Jython:
     ```
     AdminControl.invoke(cluster, 'exportRouteTable')
     ```
   After the table is created, the name of the route table file, is displayed in a message similar to the following message:
   ```
   /home/myInstall/was/server/profiles/dmgrProfile/config/cells/mycell/
       clusters/cluster1/cluster1.wsrttbl
   ```
   As this message illustrates, the file containing the table is placed in the config directory of the deployment manager for that cluster. You should keep a record of this location so that you can delete this file when you are ready to start using dynamic routing again.
4. Synchronize the configuration changes across nodes.
   a. Clear the configuration repository Epoch. If you do not clear the configuration repository Epoch, the synchronization only updates the files that the configure service component edited, which does not include the file that contains the static routing table.

      Using Jacl:
      ```
      set configRepository [$AdminControl completeObjectName
          node=node_name,type=ConfigRepository,*]
      $AdminControl invoke $configRepository refreshRepositoryEpoch
      ```
      Using Jython:
      ```
      configRepository = AdminControl.completeObjectName('node=node_name,
          type=ConfigRepository,*')
      AdminControl.invoke(configRepository, 'refreshRepositoryEpoch')
      ```
   b. Repeat this process for each node that you want to synchronize.
5. Stop the cluster. Follow the instructions specified either the *Stopping clusters* or *Stopping clusters using scripting* topic.
6. Exit the wsadmin tool.

7. Use the following Debug flag appended to the startServer command to manually start each member of this cluster.

```
-Dcom.ibm.websphere.management.registerServerIORWithLSD=false
```

For example, to start server1 on a Windows operating system with static routing enabled, issue the following command from the server profile's bin directory:

```
startServer.bat server1 -Dcom.ibm.websphere.management.registerServerIORWithLSD=false
```

Results

The cluster members use the static route table to perform IIOP routes.

What to do next

When your high availability infrastructure is enabled, follow the instructions in the topic *Disabling static routing for a cluster* to disable static routing. When static routing is disabled, the cluster members resume using dynamic routing.

Disabling static routing for a cluster

Because the information contained in a static routing table does not account for server runtime state, you should delete this table and return to using the dynamic routing table as soon as your high availability infrastructure is enabled. When you delete the static routing table, cluster members automatically resume using dynamic routing to handle enterprise bean requests.

About this task

Perform the following steps to delete the static routing table.

Procedure

1. For each member of the cluster, set the ORB_LISTENER_ADDRESS port to 0 (zero).
    a. In the administrative console, click **Servers > Server Types > WebSphere application servers > server_name**, and then in the Communications section, click **Ports**.
    b. Click **ORB_LISTENER_ADDRESS** in the Port name field.
    c. Change the value specified for the Port field to 0.
2. Manually delete the static route table file from the config directory of the deployment manager for the cluster.

    The path to this config directory was included in the message that you received when you originally exported this file. If you did not retain this information, you can do a search in the deployment manager config directory for the file cluster_name.wsrttbl.
3. Synchronize the configuration changes across nodes.
    a. Clear the configuration repository Epoch. If you do not clear the configuration repository Epoch, the synchronization only updates the files that the configure service component edited, which does not include the file that contains the static routing table.

    Using Jacl:

    ```
    set configRepository [$AdminControl completeObjectName
        node=node_name,type=ConfigRepository,*]
    $AdminControl invoke $configRepository refreshRepositoryEpoch
    ```

    Using Jython:

    ```
    configRepository = AdminControl.completeObjectName('node=node_name,
        type=ConfigRepository,*')
    AdminControl.invoke(configRepository, 'refreshRepositoryEpoch')
    ```

    b. Repeat this process for each node that you want to synchronize.

4. Stop the cluster. Follow the instructions specified either the *Stopping clusters* or *Stopping clusters using scripting* topic.
5. Start the cluster again. Follow the instructions specified either the *Starting clusters* or *Starting clusters using scripting* topic.
6. Exit the wsadmin tool.

Results

The cluster members resume using the dynamic routing table to handle IIOP requests.

Clusters on which stateful session beans will be deployed

For a cluster of servers that span multiple systems in a sysplex, and that will have stateful session beans with an activation policy of *Transaction* deployed in them, the passivation directory should reside on an HFS (hierarchical file system) that is shared across the multiple systems in the sysplex on which the clustered servers will be running.

Before creating the cluster, it is important to consider the following:
- Does the cluster have cluster members that are on different systems in the sysplex?
- If so, do any of the applications that are to be deployed or are already deployed have stateful session beans?
- If so, do the Stateful Session beans have an activation policy of Transaction?

If you answered yes to all the questions above, then you should define a shared HFS (hierarchical file system) to be used as the passivation directory for the stateful session beans.

The name of the passivation directory should contain the install root and cluster name. In the following example, `/WebSphere/V6R0M0/AppServer` is also known as USER_INSTALL_ROOT, and the name of the cluster is **cluster1**.

`/WebSphere/V6R0M0/AppServer/passivation/cluster1`

For optimal performance, you should create a passivation directory for each cluster. Also, the default passivation directory should not be used for clusters; it should be used for non-clustered servers and servers that do not have stateful session beans with an activation policy of *Transaction*.

For information about defining a shared HFS, see *z/OS UNIX System Services Planning*.

For information on specifying the passivation directory with the administrative console, see the *Administering applications and their environment* PDF.

Starting clusters

You can start all members of a cluster at the same time by requesting that the state of a cluster change to *running*. That is, you can start all application servers in a server cluster at the same time.

Before you begin

Make sure that the members of your cluster have the debug port properly set. If multiple servers on the same node have the same debug port set, the cluster could fail to start. Read about how to change the debug port in the topic about Java virtual machine settings.

If you want cluster member components to dynamically start as they are needed by the installed applications, verify that the **Start components as needed** option is selected in the configuration settings for each of the cluster members before you start the cluster. Selecting this option can improve cluster startup time, and reduce the memory footprint of the cluster members. Starting components as they are

needed is most effective if all of the applications that are deployed on the cluster are of the same type. For example, using this option works better if all of your applications are Web applications that use servlets, and JavaServer Pages (JSP). This option works less effectively if your applications use servlets, JSPs and Enterprise JavaBeans (EJB).

gotcha: To ensure compatibility with other WebSphere products, the default setting for this option is deselected. Before selecting this option, verify that any other WebSphere products, that you are running in conjunction with this product, support this functionality.

About this task

When you request that all members of a cluster start, the cluster state changes to partially started and each server that is a member of that cluster launches, if it is not already running. After all members of the cluster are running, the cluster state becomes running.

gotcha: From the z/OS MVS console, you must individually start each server that you want to run. With the administrative console, you can start each server individually, or you can start a cluster. Starting a cluster automatically starts all of the servers that are defined as part of that cluster.

Procedure

1. In the administrative console, click **Servers > Clusters > WebSphere application server clusters >** .
2. Select the clusters whose members you want started.
3. Click **Start** or **Ripplestart**.
   - **Start** launches the server process of each member of the cluster by calling the node agent for each server to start the servers. After all servers are running, the state of the cluster changes to *running*. If the call to a node agent for a server fails, the server does not start.
   - **Ripplestart** combines stopping and starting operations. It first stops and then restarts each member of the cluster. For example, your cluster contains 3 cluster members named server_1, server_2 and server_3. When you click Ripplestart, server_1 stops and restarts, then server_2 stops and restarts, and finally server_3 stops and restarts. Use the Ripplestart option instead of manually stopping and then starting all of the application servers in the cluster.

   gotcha: If recently added cluster members do not start, you might not have selected **Synchronize changes with nodes** when you added the members to the cluster. To determine if this is the problem:
   a. In the administrative console click **Servers > Clusters > WebSphere application server clusters >** , select the cluster whose members did not start, and click **Stop**.
   b. Click the name of the cluster, click **OK**, and then click **Review**.
   c. Select **Synchronize changes with Nodes**, and then click **Save**.
   d. Start the cluster and verify that all of the cluster members now start.

Results

When you start the members of a cluster, you automatically enable workload management.

Stopping clusters

Use this task to stop a cluster and any application servers that are members of that cluster.

Before you begin

About this task

You can stop all application servers that are members of the same cluster at the same time by stopping the cluster.

Procedure

1. Click **Servers > Clusters > WebSphere application server clusters >** in the console navigation tree to access the Server Cluster page.
2. Select those clusters whose members you want stopped.
3. Click **Stop** or **Immediate Stop**.
   - **Stop** halts each server in a manner that allows the server to finish existing requests and allows failover to another member of the cluster. When the stop operation begins the cluster state changes to *partially stopped*. After all servers stop, the cluster state becomes Stopped.
   - **Immediate Stop** brings down the server quickly without regard to existing requests. The server ignores any current or pending tasks. When the stop operation begins, the cluster state changes to *partially stopped*. After all servers stop, the cluster state becomes Stopped.

Results

All application servers in the sysplex associated with this cluster are issued a request to stop. In addition, a **stop** can be issued against each individual server from the MVS console. To shut down the product environment on a specific system, stop that system daemon. Stopping the system daemon brings down all other server instances on the system. To bring the product down on all of your systems, stop the daemons on all systems. When you stop the location service daemon on one system, it does not bring down the servers on the other systems.

What to do next

See Chapter 5, "Balancing workloads," on page 499 for more information about the tasks you can complete with clustering.

Adding members to a cluster

You can use clusters to balance workload in an environment containing multiple application servers.

Before you begin

Create a cluster if you do not already have a cluster defined for your environment.

About this task

If you are migrating from a previous version of the product, you can upgrade a portion of the nodes in a cell, while leaving others at the previous release level. For a time, you might be managing servers that are at a previous release level, and servers that are running at the current release level in the same cell.

When you create a cluster, you specify the node on which the first cluster member resides. In a mixed cell environment, you can use any server from within that node group to create a new cluster member. For example, if the node group to which the cluster belongs consists of a Version 7.0 node, and a Version 6.1 node, you can use a server from either the Version 6.1 or the Version 7.0 node to create a new cluster member.

gotcha: When you add a member to a cluster, the virtual host for the new cluster member is automatically configured if the deployment manager profile and the managed profile (the cluster member)

| reside on the same LPAR. Therefore, in this situation, the host alias ports are automatically
| added to the virtual host for the new cluster member. However, if the deployment manager profile
| and the managed profile reside on different LPARs, you must manually configure the virtual host
| for the new cluster member.

Use the following procedure to create a new cluster member, view information about existing cluster members, or manage existing cluster members.

Procedure

1. In the administrative console, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Cluster members**. The Cluster members page lists members of a cluster, and for each member indicates:

   - The node on which the member resides.
   - The version of the application server. This information specifies whether the cluster is a mixed cluster.
   - The configured weight for the member.
   - The runtime weight for the member. This weight indicates the proportionate workload that is currently directed to this cluster member.
   - Whether the member is started, stopped, or encountering problems.

2. Click **New** to create a new cluster member.

   Clicking **New** starts the Create a new cluster member wizard. Use this wizard to add new members to an already configured cluster.

   A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create. Usually, only one template is available for you to use to create additional cluster members for a cluster. However, if a cluster includes nodes that are at different versions of the product, there is a different template for each version. For example, if a cluster has cluster members that reside on both a Version 6.1 node and a Version 7.0 node, the cluster has two templates. The Version 6.1 template is used when you create an additional cluster member on the Version 6.1 node, and the Version 7.0 template is used when you create an additional cluster member on the Version 7.0 node.

   To view the cluster member templates that are available for creating a new member of a cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Cluster members > Templates**.

   a. Specify a name for the application server that you are defining as a cluster member. The name must be unique within the node.

   b. Select the node for the cluster member.

   c. Specify a short name for this cluster member. The short name is the default job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, RACF, and started task control.

   d. Specify the server weight.

   The weight value you specify controls the number of requests that are directed to the application server. Even though you specify a value of 0 to 20 as the weight of a server, the weight that is given to the server as a member of a cluster is a proportion that is based on the weight assigned to the server, and the sum of the weights of all members of the cluster. In this proportion, the weight that is assigned to the server is the numerator, and the sum of the weights of all members of the cluster is the denominator.

   When you add a new member to a cluster, the number of client, or application requests that are sent to each server in the cluster decreases, assuming the number of requests coming into the cluster stays the same. Similarly when you remove a new member from a cluster, the number of client or application requests that are sent to each server in the cluster increases, assuming the number of requests coming into the cluster stays the same.

For example, if you have a cluster that consists of members A, B, and C with weights 2, 3, and 4, respectively, then 2/9 of the requests are assigned to member A, 3/9 are assigned to member B, and 4/9 are assigned to member C. If a new member, member D, is added to the cluster and member D has a weight of 5, then member A now gets 2/14 of the requests, member B gets 3/14 of the requests, member C gets 4/14 of the requests, and member D gets 5/14 of the requests.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

- For HTTP requests, weights are used to distribute HTTP traffic between the web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.

- For web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.

- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

gotcha: For workload balancing with service integration, or with message driven beans in the application server, you require additional configuration. See the topic *Configuring high availability and workload sharing of service integration* for more information.

   e. Specify whether to generate unique HTTP ports.

   f. Click **Add member** to finish defining the cluster member. The first cluster member for this cluster is used as the template for this cluster member. You can repeat these steps to define other cluster members.

   g. When you finish defining additional cluster members, review the summary information for the new cluster members. If you have to change any of the property settings for any of the new members, select that cluster member, and then click **Edit**. When you finish changing the property settings, click **Update member** to save your changes.

   h. When you finish defining new cluster members, click **Next** to view the summary page for the cluster, and then click **Finish** to create these new cluster members.

3. Click **Review**, select **Synchronize changes with nodes**, and then click **Save** to save your changes.

Results

You created application servers that are members of an existing server cluster.

What to do next

If, when you created the new members, you chose to generate unique ports, update the alias list for the virtual host that you plan to use with the new servers.

You can also perform the following actions:

- On the Cluster members page in the administrative console, click the name of one of the cluster members, and examine the configuration settings for that cluster member. You can change any of the settings that are not appropriate.

  For example, if you do not need to have all of the cluster member components start during the cluster member startup process, you might want to select **Start components as needed**, which is not automatically selected when a new cluster member is created. When this property is selected, cluster member components are dynamically started as they are needed. When this property is not selected, all of the cluster member components are started during the startup process. Therefore, selecting this property usually results in improved startup performance because fewer components are started during the startup process.

- Click **Servers > Sever Types > WebSphere application servers >***sever_name* or click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Cluster members >** *cluster_member_name* to perform either of the following tasks:
  - Specify additional application server properties for this cluster member.
  - Click **Installed applications**.
- Start the cluster.
- Use scripting to automate the task of adding cluster members.

Cluster member collection

Use this page to view and manage application servers that belong to a cluster. You can also use this page to change the weight of any of the listed application servers.

Application servers that are part of a cluster are referred to as cluster members. A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

You can use this page to perform the following actions for the listed cluster members:
- Start a cluster member. To start a cluster member, select the server that you want to start. Then click **Start**.
- Restart a cluster member. To restart a cluster member, select the server that you want to restart. Then click **Restart**. When you restart a cluster member, the selected cluster member stops, following the normal server quiesce process, and then starts again.
- Stop a cluster member. To stop a cluster member, select the server that you want to stop, and then click one of the following buttons:

  Stop   When you click this button, the normal server quiesce process is followed. This process allows in-flight requests to complete before the entire server process shuts down.

  Immediate Stop
  When you click this button, the selected sever stops but the normal server quiesce process is not followed. This shutdown mode is faster than the normal server stop processing, but some application clients might receive exceptions if an in-flight request does not complete before the server process shuts down.

  Terminate
  You should only click **Terminate** if the cluster member does not respond when you click **Stop** or **Immediate Stop** or when you issue the Stop or ImmediateStop commands. Some application clients can receive exceptions. Therefore, you should always attempt an immediate stop before clicking **Terminate**.

  Make Idle
  When you click this button, the cluster member moves to the idle state.
- Delete a cluster member. To delete a cluster member, select the cluster member that you want to delete. Then click **Delete**.

Any individual configuration change that you make to a cluster member does not affect the configuration settings of the cluster member template. You must use wsadmin commands to modify this template. Similarly, any changes that you make to the template do not affect existing cluster members.

See the *Using the administrative clients* PDF for more information on how to modify this template.

To view this administrative console page, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Clusters members**.

Member name

Specifies the name of the server in the cluster. On most platforms, the name of the server is the process name. The name must match the (object) name of the application server.

Node

Specifies the name of the node for the cluster member.

Host Name

Specifies the IP address, the full domain name system (DNS) host name with a domain name suffix, or the short DNS host name for the cluster member.

Version

Specifies the version of the product on which the cluster member runs.

Configured weight

Specifies the weight that is currently configured for the cluster member. The weight determines the amount of work that is directed to the cluster member. If the weight value for the server is greater than the weight values assigned to other servers in the cluster, the server receives a larger share of the cluster workload.

To change the configured weight for a cluster member you can either specify a new weight in the Configured weight field and click the **Update** button for the Configured weight column, or click on the name of the cluster member. Clicking the name of the cluster member navigates you to the page where you can change any of the configuration settings for that cluster member.

Runtime weight

Specifies the proportionate workload that is currently directed to the cluster member in comparison to the runtime weights of the rest of the cluster members. The runtime weight only applies for Remote Method Invocation over Internet Inter-ORB Protocol (RMI-IIOP) requests, Enterprise JavaBeans (EJB) requests, and HTTP requests received through a WebSphere proxy server.

You can use the **Runtime weight** property to dynamically adjust the weight that is given to a particular cluster member without having to stop and then restart that cluster member. To change the proportion, specify a new weight for the **Runtime weight** property, and then click the **Update** button for the Runtime weight column. The runtime weight change takes effect immediately.

gotcha: Changing the runtime weight for a cluster member does not affect the configured weight setting for that cluster member. The configured weight setting still applies for HTTP requests that do not come through a WebSphere proxy server.

Status

This field indicates whether the cluster member is started, stopped, partially stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

Table 63. Status and meaning. The following table describes the status icons.

➔	**Started**	The server is running.
✸	**Partially stopped**	The server is in the process of changing from a started state to a stopped state.
✱	**Stopped**	The server is not running.
⊘	**Unknown**	The server status cannot be determined.

Cluster member settings

Use this page to manage the members of a cluster. A cluster of application servers are managed together and participate in workload management.

A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

Any individual configuration change that you make to a cluster member does not affect the configuration settings of the cluster member template. You can use wsadmin commands to modify the cluster member template, or you can click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Clusters members > Templates**. Any change that you make to the template does not affect existing cluster members. See the *Using the administrative clients* PDF for more information on how to use wsadmin commands to modify this template.

To view this administrative console page, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name*.

On the **Configuration** tab, you can edit fields. You can also click **Installed applications.** to view the status of applications that are running on this server. On the **Runtime** tab, which only appears when the cluster member is running, you can look at information about this cluster member. However, the information that displays on this page is read-only. You must return to the **Configuration** tab to change any of the settings that display.

Member name:

Specifies the name of the application server in the cluster. On most platforms, the name of the server is the process name. The member name must match the name of one of the servers that are listed on the application servers page.

Node name:

Specifies the name of the node on which the cluster member is running.

Weight:

Controls the number of requests that are directed to the application server. Even though you specify a value of 0 to 20 as the weight of a server, the weight that is assigned to the server is a proportion, in which, the weight assigned to the server is the numerator, and the sum of the weights of all members of the cluster is the denominator.

When you add a new member to a cluster, the number of client, or application requests that are sent to each server in the cluster decreases, assuming the number of requests coming into the cluster stays the same. Similarly when you remove a new member from a cluster, the number of client or application requests that are sent to each server in the cluster increases, assuming the number of requests coming into the cluster stays the same.

For example, if you have a cluster that consists of members A, B, and C with weights 2, 3, and 4, respectively, then 2/9 of the requests are assigned to member A, 3/9 are assigned to member B, and 4/9 are assigned to member C. If a new member, member D, is added to the cluster and member D has a weight of 5, then member A now gets 2/14 of the requests, member B gets 3/14 of the requests, member C gets 4/14 of the requests, and member D gets 5/14 of the requests.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system.
- For HTTP requests, weights are used to distribute HTTP traffic between the web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
- For web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.

• Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Data type	Integer
Range	0 to 20

Unique ID:

Specifies a numerical identifier for the application server that is unique within the cluster. The ID is used for affinity.

Data type	Hexadecimal

Short name:

Specifies the short name for this cluster member. This field only displays if you are running on z/OS.

The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster member, the name:
• Must be one to eight characters in length. By default, when you are running the product on z/OS, the product assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
• Must contain only uppercase alphanumeric characters
• Cannot start with a number.
• Must be unique in the cell.
• Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Data type	String

Run in development mode:

Enabling this option may reduce the startup time of an application server. This might include Java virtual machine (JVM) settings, such as disabling bytecode verification and reducing Just in Time (JIT) compiler compilation costs. Do not enable this setting on production servers. This setting is only available on application servers that are running in Version 6.0 and or higher cells.

Specifies that you want to use the JVM settings, **-Xverify** and **-Xquickstart**, on startup. After selecting this option, save the configuration and restart the server to activate development mode.

The default setting for this option is `false`, which indicates that the server is not started in development mode. Setting this option to `true` specifies that the server is started in development mode, using settings that decrease server startup time.

Data type	Boolean
Default	false

Parallel start:

Specifies whether to start the server on multiple threads. When you start the server on multiple threads, the server components, services, and applications start in parallel rather than sequentially, which might shorten the startup time.

The default setting for this option is `true`, which indicates that the server uses multiple threads when it starts. Setting this option to `false` specifies that the server uses a single thread when it starts, which might lengthen startup time.

The order in which applications start depends on the weight you assign to each application. The application with the lowest starting weight starts first. Applications with the same starting weight start in parallel. Use the `Starting weight` field on the **Applications > Application Types > WebSphere enterprise applications >** *application_name* **> Startup behavior** page of the administrative console to set the starting weight for an application.

Data type   Boolean
Default   true

Start components as needed:

Select this field if you want the cluster member components started as they are needed by an application that is running on this cluster member.

When this property is selected, cluster member components are dynamically started as they are needed. When this property is not selected, all of the cluster member components are started during the cluster startup process. Therefore, selecting this option can improve startup time, and reduce the memory footprint of the cluster members, because fewer components are started during the startup process.

Starting components as they are needed is most effective if all of the applications, that are deployed on the cluster, are of the same type. For example, using this option works better if all of your applications are web applications that use servlets, and JavaServer Pages (JSP). This option works less effectively if your applications use servlets, JSPs and Enterprise JavaBeans (EJB).

gotcha: To ensure compatibility with other WebSphere products, the default setting for this option is deselected. Before selecting this option, verify that any other WebSphere products, that you are running in conjunction with this product, support this functionality.

Run in 64 bit JVM mode:

Specifies that the application server runs in 64-bit mode, which is the default setting. Running in 64-bit mode provides additional virtual storage for user applications. This field only displays if you are running on z/OS.

You can deselect this setting if you need to run the application server in 31-bit mode.

There is no interdependence between the modes in which you are running different servers. Therefore, you can run some of your servers in 64-bit mode and some of your servers in 31-bit mode. However, you should eventually convert all of your servers to run in 64-bit mode because support for running servers in 31-bit mode is deprecated.

Access to internal server classes:

Specifies whether the applications that are running on this server can access many of the server implementation classes.

If you select `Allow`, then, applications can access most of the server implementation classes. If you select `Restrict`, then applications cannot access server implementation classes. The applications get a ClassNotFoundException error if they attempt to access these classes.

Usually, you should select `Restrict` for this property, because most applications use the supported APIs, and do not need to access any of the internal classes. However, if your application requires the use of one or more of the internal server classes, then select `Allow` as the value for this property.

The default value for this property is `Allow`.

Class loader policy:

Specifies whether there is a single class loader that loads all of the applications, or a different class loader loads each application.

Class loading mode:

Specifies whether the class loader searches in the parent class loader, or in the application class loader first to load a class. The standard for Developer Kit class loaders and product class loaders is `Classes loaded with parent class loader first`.

This field only applies if you set the Class loader policy field to `Single`.

If you select `Classes loaded with local class loader first (parent last)`, your application can override classes that are contained in the parent class loader, but this action can potentially result in ClassCastException, or linkage errors, if you have mixed use of overridden classes and non-overridden classes.

Process ID:

Specifies the native operating system process ID for this server.

The process ID property is read only. The system automatically generates the value.

Cell name:

Specifies the name of the cell in which this server is running.

The Cell name property is read only.

Node name:

Specifies the name of the node in which this server is running.

The Node name property is read only.

State:

Specifies the runtime state for this server.

The State property is read only.

Cluster member templates collection

Use this page to view the list of cluster member templates that exist for this cluster. To edit the server properties of a template, click the name of that template.

Usually, only one template exists that you can use to create additional members for a cluster. However, if a cluster includes nodes that are at different versions of the product, a different template exists for each of these versions. For example, if a cluster has cluster members residing on both a Version 6.1 node and a Version 7.0 node, the cluster has two templates. The Version 6.1. template is used when you create an additional cluster member on the Version 6.1 node, and the Version 7.0 template is used when you create an additional cluster member on the Version 7.0 node.

If you modify a template, all new cluster members are created with the server property settings of the modified template. However, the property settings of existing cluster members do not change. If you make any change to a cluster member template, you should make the same change to all of the existing cluster members.

To change the server attributes of an existing cluster member, in the administrative console, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Clusters members**, and then click the name of the existing cluster member.

To view the cluster member templates that are available for creating a new member of a cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters >** *cluster_name* **> Cluster members >** *cluster_member_name*, and then click **Templates**.

Name
Specifies the name of the cluster member template.

Platform
Specifies the operating system platform to which this template applies.

Version
Specifies the version of the product to which the template applies.

Description
Specifies a description of this cluster member template. This field is optional and might be blank.

Replicating data across application servers in a cluster

Use this task to configure a data replication domain to transfer data, objects, or events for session manager, dynamic cache, or stateful session beans. Data replication domains use the data replication service (DRS), which is an internal component that performs replication services, including replicating data, objects, and events among application servers.

Before you begin

Determine if you are using a multi-broker replication domain. If you configured a data replication domain with a previous version of the product, you might be using a multi-broker replication domain. Any replication domains that you create with the current version of the product are data replication domains. You should migrate any multi-broker replication domains to data replication domains.

About this task

Use this task to configure *replication*, a service that transfers data, objects, or events among the application servers in a cluster. Use replication to prevent loss of session data with session manager, to further improve the performance of the dynamic cache service, and to provide failover in stateful session beans.

gotcha: If you select the **Configure HTTP memory-to-memory replication** option when you create a cluster, the replication domain is automatically created for you.

Similarly if, instead of WAS01Network , the cell name is simply WAS1, you have to pad the high level qualifier with the first three characters of the string DRSSTREAM. The high level qualifier then becomes WAS1DRS.

Complete the following steps to enable data replication among the application servers in a cluster.

Procedure

1. Create a replication domain. Use one of the following methods to create a replication domain:
   - **Create a replication domain manually.**

     To create a replication domain manually without creating a new cluster, click **Environment > Replication domains > New** in the administrative console.

     On this page you can specify the properties for the replication domain, including timeout, encryption, and number of replicas.
   - **Create a replication domain when you create a cluster.**

     To create a replication domain when you create a cluster, click **Servers > Clusters > Clusters > New** in the administrative console. Then click **Configure HTTP memory-to-memory replication**. The replication domain that is created has the same name as the cluster and has the default settings for a replication domain. The default settings for a replication domain are to create a single replica of each piece of data and to have encryption disabled. To modify the replication domain properties, click **Environment > Replication domains > New** *replication_domain_name* in the administrative console.

2. Configure the consumers, or the components that use the replication domains. Dynamic cache, session manager, and stateful session beans are the three types of replication domain consumers. Each type of consumer must be configured with a different replication domain. For example, session manager uses one replication domain and dynamic cache uses a different replication domain. However, use one replication domain if you are configuring HTTP session memory-to-memory replication and stateful session bean replication. Using one replication domain in this case ensures that the backup state information of HTTP sessions and stateful session beans are on the same application servers.

Results

Data is replicating among the application servers in a configured replication domain.

What to do next

If you select DES or 3DES as the encryption type for a replication domain, an encryption key is used for the encryption of messages. At regular intervals, for example once a month, you should go to the **Environment > Replication domains > New** page in the administrative console, and click **Regenerate encryption key** to regenerate the key. After the key is regenerated, you must restart all of the application servers that are configured as part of the replication domain. Periodically regenerating the key improves data security.

Data replication

Replication is a service that transfers data, objects, or events among application servers. Data replication service (DRS) is the internal WebSphere Application Server component that replicates data.

Use data replication to make data for session manager, dynamic cache, and stateful session beans available across many application servers in a cluster. The benefits of using replication vary depending on the component that you configure to use replication.
- Session manager uses the data replication service when configured to do memory-to-memory replication. When memory-to-memory replication is configured, session manager maintains data about

sessions across multiple application servers, preventing the loss of session data if a single application server fails. For more information about memory-to-memory replication, see the *Administering applications and their environment* PDF.

- Dynamic cache uses the data replication service to further improve performance by copying cache information across application servers in the cluster, preventing the need to repeatedly perform the same tasks and queries in different application servers. For more information about replication in the dynamic cache, see the *Administering applications and their environment* PDF.

- Stateful session beans use the replication service so that applications using stateful session beans are not limited by unexpected server failures. For more information about stateful session bean failover, see the *Developing and deploying applications* PDF.

Important: When you use the replication services, ensure that the **Propagate security attributes** option is enabled. Security attribute propagation is enabled, by default.

You can define the number of *replicas* that DRS creates on remote application servers. A replica is a copy of the data that copies from one application server to another. The number of replicas that you configure affects the performance of your configuration. Smaller numbers of replicas result in better performance because the data does not have to copy many times. However, if you create more replicas, you have more redundancy in your system. By configuring more replicas, your system becomes more tolerant to possible failures of application servers in the system because the data is backed up in several locations.

Defining a single replica configuration helps you to avoid a single point of failure in the system. However, if your system must be tolerant to more failure, introduce extra redundancy in the system. Increase the number of replicas that you create for any HTTP session that is replicated with DRS. The **Number of replicas** property for any replication domain that is used by the dynamic cache service must be set to `Entire domain`.

Session manager, dynamic cache, and stateful session beans are the three *consumers* of replication. A consumer is a component that uses the replication service. When you configure replication, the same types of consumers belong to the same *replication domain*. For example, if you are configuring both session manager and dynamic cache to use DRS to replicate objects, create separate replication domains for each consumer. Create one replication domain for all the session managers on all the application servers and one replication domain for the dynamic cache on all the application servers. The only exception to this rule is to create one replication domain if you are configuring replication for HTTP sessions and stateful session beans. Configuring one replication domain in this case ensures that the backup state information is located on the same backup application servers.

See the *Administering applications and their environment* PDF for more information on how to configure replication.

Replication domain collection

Use this page to view the configured replication domains that are used for replication by the HTTP session manager, dynamic cache service, and stateful session bean failover components. All components that need to share information must be in the same replication domain. Data replication domains replace multi-broker replication domains that were available for replication in prior releases. Migrated application servers use multi-broker replication domains which are collections of replicators. You should migrate any multi-broker replication domains to be data replication domains.

To view this administrative console page, click **Environment > Replication domains**.

Name
Specifies a name for the replication domain. The name of the replication domain must be unique within the cell.

Domain type

Following are the two types of replication domains:

Multi-broker domain	Specifies a replication domain that was created with a previous version of WebSphere Application Server. This type of replication domain consists of replicator entries. Support of this type of domain remains for backward compatibility, but is deprecated. Multi-broker and data replication domains do not communicate with each other, so migrate any multi-broker replication domains to the new data replication domains. You cannot create a multi-broker domain or replicator entries in the administrative console after the deployment manager is upgraded to the current version of WebSphere Application Server.
Data replication domain	Specifies a replication domain created with the latest version of WebSphere Application Server. If the deployment manager has been upgraded to the latest version of WebSphere Application Server, you can create data replication domains only. With the data replication domain, you can specify a number of replicas instead of statically partitioning your replication settings. Specify a data replication domain for each consumer of the domain, for example, two separate domains for dynamic cache and session manager.

Data replication domain settings

Use this page to configure a data replication domain. Use data replication domains to transfer data, objects, or events for session manager, dynamic cache, or stateful session beans among the application servers in a cluster.

To view this administrative console page, click **Environment > Replication domains >** *replication_domain_name*.

Name:

Specifies a name for the replication domain. The name must be unique within the cell.

Request timeout:

Specifies how long a replication domain consumer waits when requesting information from another replication domain consumer before it gives up and assumes that the information does not exist.

Units	seconds
Default	5 seconds

Number of replicas:

Specifies the number of replicas that are created for every entry or piece of data that is replicated in the replication domain.

Single replica	When you select this option, every HTTP session is replicated to exactly one other application server. This is the default value.

| Entire domain | When you select this option, each object is replicated to every application server that is configured as a user of the replication domain. |
| Specify | When you select this option, you must specify, in the Number of replicas field, the number of replicas that you want created for each HTTP session. |

Migrating servers from multi-broker replication domains to data replication domains

You can migrate multi-broker replication domains to data replication domains. Any multi-broker domains that exist in your application server environment were created with a previous version of the product.

Before you begin

Determine if the application server configuration you are migrating:

1. Uses an instance of data replication service in peer-to-peer mode or in client/server mode.

   Before you begin migrating a client/server mode replication domain, consider if migrating your replication domains might cause a single point of failure. Because you migrate the servers to the new type of replication domain one at a time, you risk a single point of failure if there are 3 or fewer application servers. Before migrating, configure at least 4 servers that use multi-broker replication domains. Perform the following steps to migrate the multi-broker domains to data replication domains:

   Dynamic cache replication domains use the peer-to-peer topology.

2. Uses HTTP session memory-to-memory replication that is overloaded at the application or web module level.

   If the application server configuration you are migrating uses HTTP session memory-to-memory replication that is overloaded at the application or web module level, you must upgrade your deployment manager to the current version of the product before you start the migration process.

About this task

After you upgrade your deployment manager to the latest version of the product, you can only create data replication domains. Any multi-broker domains that you created with a previous version of the product are still functional, however, you cannot use the administrative console to create new multi-broker domains or replicators.

The different versions of application servers cannot communicate with each other. When migrating your servers to the current version of the product, keep at least two application servers running on the previous version so that replication remains functional.

Make sure that all of your application servers that are using this multi-broker domain have been migrated to the current version of the product before you start to migrate any multi-broker domains that exist in your configuration.

To migrate the multi-broker domains that exist in your configuration:

Procedure

1. Migrate two or more of your existing servers to the current version of the product. The remaining servers on the previous version of the product can still communicate with each other, but not with the migrated servers. The migrated servers can also communicate with each other.

2. In the administrative console, create an empty data replication domain. Click **Environment > > Replication domains > New** to create an empty data replication domain.

3. Add two of your migrated servers to the new data replication domain.

For example, if you are migrated four servers, only add two of them to the new replication domain.

4. Configure the two servers as consumers of the replication domain.

   Configuring the servers as consumers of the replication domain enables them to use the new domain to share data.

5. Add some of the clients to the new data replication domain.

   Perform this step only if the application server configuration you are migrating uses an instance of data replication service in client/server mode.

6. Configure these clients as consumers of the replication domain.

7. Verify that the new data replication domain are successfully sharing data.

   Only the servers and clients that are added to the data replication domain and are configured as consumers of this domain can use the data replication domain functions.

8. Add the rest of your migrated servers to the new data replication domain.

   When the servers can use the new data replication domain to successfully share data, migrate the rest of the servers that are using the multi-broker replication domain to the new data replication domain.

   For example, if you are migrated four servers, add the remaining two servers to the new replication domain.

9. Configure these servers as consumers of the replication domain.

10. Add the rest of the clients to the new data replication domain.

    Perform this step only if the application server configuration you are migrating uses an instance of data replication service in client/server mode.

11. Configure these clients as consumers of the replication domain.

12. Restart all of the application servers and clients.

13. Delete the empty multi-broker replication domain.

What to do next

During this process, you might lose existing sessions. However, the application remains active through the entire process, so users do not experience down time during the migration. Create a new replication domain for each type of consumer. For example, create one replication domain for the session manager and another replication domain for dynamic cache.

Data replication domains

Data replication domains and multi-broker domains both perform the same function, which is to provide data replication between application servers in a cluster. Even though you can still configure existing multi-broker domains with the current version of the product, after you upgrade your deployment manager, you can only create data replication domains in the administrative console.

transition: A replication domain that was created with a previous version of the product might be a multi-broker domain. You should migrate these multi-broker domains to data replication domains. Data replication domains enable you to:

- Configure all of the instances of replication that need to communicate in the same replication domain.
- Configure the session manager with both types of replication domains to use topologies such as peer-to-peer, and client-to-server to isolate the function of creating and storing replicas on separate application servers.
- Control the redundancy of replication for each type of replication domain, because with a data replication domain, you can specify a specific number of replicas.

If you used multi-broker domains with earlier releases of the product, use the following comparison chart to learn the differences between how Version 5.x and Version 6.x and Version 7.0 application servers use the two types of replication domains:

	Version 5.x application servers using replication domains	V6.x and Version 7.0 application servers using replication domains
Replication domain types	Uses only multi-broker replication domains for replication.	Servers that are using the current version of the product can be configured to use both multi-broker replication domains and data replication domains for replication. The two types of domains provide backward compatibility with multi-broker domains that were created with a Version 5.x server. You should migrate any multi-broker domains to data replication domains.
Data transport method	Uses multi-broker domain objects that contain configuration information for the internal Java Message Service (JMS) provider, which uses JMS brokers as replicators.	Uses data replication domain objects that contain configuration information to configure the high availability framework on the product. The transport is no longer based on the JMS API. Therefore, no replicators and no JMS brokers exist. You do not have to perform the complex task of configuring local, remote, and alternate replicators. The earlier version of the product did not support data replication domains. The current version of the product can be configured to perform replication using old multi-broker domains by ignoring any JMS-specific configuration and by using the other parameters to configure replication through the high availability framework.
Replication domain configuration	The earlier version of the product encourages the sharing of replication domains between different consumers, such as session manager and dynamic cache.	The current version of the product encourages creating a separate replication domain for each consumer. For example, create one replication domain for session manager and another replication domain for dynamic cache. The only situation where you should configure one replication domain is when configuring session manager replication and stateful session bean failover. Using one replication domain in this case ensures that the backup state information of HTTP sessions and stateful session beans are on the same application servers.

	Version 5.x application servers using replication domains	V6.x and Version 7.0 application servers using replication domains
Partial partitioning	You can configure partial partitioning. Partition the replication domain to filter the number of processes to send data.	Partial partitioning is deprecated. When using data replication domains, you can specify a specific number of replicas for each entry. However, if you specify a number of replicas larger than the number of backup application servers that are running, the number of replicas is the number of application servers that are running. After the number of application servers increases above your configured number of replicas, the number of replicas that are created is equal to the number that you specified.
Domain sharing	Multiple data replication service (DRS) instances share multi-broker domains. A limitation exists on the number of multi-broker domains that you can create because every multi-broker domain contains at least one replicator. A maximum of one replicator can be on each application server.	All DRS instances in a replication domain use the same mode. Each replication domain must contain either client only and server only instances, or client and server instances only. For example, if one instance is configured to client and server, all other instances must be client and server. If one instance in a replication domain is configured to be a client only, you can add client only and server only instances, but not a client and server instance.

Deleting replication domains

You can manually create a replication domain, or have a replication domain automatically generated when you create a cluster. When you no longer need a previously defined replication domain, you can use either the wsadmin AdminConfig delete command or the administrative console to delete the replication domain from your application server environment. Deleting a cluster does not automatically delete a replication domain that is associated with that cluster.

Before you begin

- Verify that none of the applications that you are running require the replication domain you are deleting.
- Verify that neither dynamic caching nor the session manager are configured to use the replication domain that you are deleting. Deleting a replication domain that either dynamic caching or the session manager has been configured to use might cause processing errors.

About this task

Perform the following steps if you want to use the administrative console to delete a replication domain.

Procedure

1. In the administrative console. click **Environment > > Replication domains**.
2. Select the replication domain that you want to delete.
3. Click **Delete**.
4. Select Synchronize changes with Nodes, and then click **Save** to save your workspace changes to the master configuration.

Results

Data is not replicated amongst the application servers that were part of the configured replication domain that you deleted, and all session data that is associated with the deleted replication domain is deleted.

Replicating data with a multi-broker replication domain

Use this task to manage replication domains that you migrated from a Version 6.x or earlier product environment.

Before you begin

transition: Multi-broker replication domains are not created in Version 7.0 product environments. However they can be migrated from existing Version 6.x product environments. If you migrate Version 6.x multi-broker replication domains, you can use the Multi-broker domain panel in the Version 7.0 administrative console to manage these domains.

Although you can manage migrated multi-broker domains with the current version of the product, after you upgrade your deployment manager, you can create only data replication domains in the administrative console. Consider migrating any existing multi-broker domains to the new data replication domains.

About this task

If you are performing this task, it is assumed that you configured replication with a previous version of the product, and defined replication domains that list connected replicator entries, residing in managed servers in the cell that can exchange data. You can manage these existing replication domains and replicator entries, but you cannot create new multi-broker replication domains or new replicator entries in the administrative console.

A replicator does not need to run in the same process as the application server that uses it. However, it might be easier to manage replicators and replication domains if a one-to-one relationship exists between replicators and application servers. During configuration, you can select the local replicator as the default replicator.

Procedure

1. Manage multi-broker replication domain configuration settings. In the administrative console, click **Environment > Replication domains**.
2. Click **Multi-broker domain >***multi-broker domain_name*, and update the values for that particular multi-broker replication domain. The default values are generally sufficient, especially for the pooling and timeout properties.
   a. Name the replication domain.
   b. Specify the timeout interval.
   c. Specify the encryption type. The DES and TRIPLE_DES options encrypt data sent between application server processes and better secure the network joining the processes.
   d. Partition the replication domain to filter the number of processes to which data is sent. Partitioning the replication domain is most often done if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails. Partitioning is not supported for sharing of cached data that is maintained by web container dynamic caching.
   e. Specify whether you want a single replication of data to be made. Enable the option if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails.
   f. Specify whether processes should receive data in objects or bytes. Processes receiving data in objects receive the data and class definitions. Processes receiving data in bytes receive the data only.

g. Configure a pool of replication resources. Pooling replication resources can enhance the performance of the replication service.

3. Maintain the replicators that you have already defined. You cannot create any replicators. The default convention is to define a replicator in each application server that uses replication. However, you can define a pool of replicators, separate from the servers hosting applications.

   a. In the administrative console, click **Environment > Replication domains** *replication_domain_name* **> Replicator entries >** *replicator_entry_name*.

   b. Specify a replicator name and select a server available within the cell to which you can assign a replicator. Also specify a host name and ports. Note that a replicator has two ports (replicator and client ports) that use the same host name but have different ports.

4. Click **OK** to save your changes.

Multi-broker replication domains

A multi-broker replication domain is a collection of replication entries, or *replicator* instances, used by clusters or individual servers within a cell. Multi-broker replication domains were created with a previous release of the product.

gotcha: After you upgrade your deployment manager to the latest version of the product, you can create data replication domains only. Any multi-broker domains that you created with a previous release of the product are still functional, however, you cannot create new multi-broker domains or replicator instances with the administrative console.

A replication entry, or replicator, is a run-time component that handles the transfer of internal product data. All replicators within a replication domain connect with each other, forming a network of replicators.

Components such as session manager and dynamic cache can connect to any replicator within a domain to receive data from their peer components on other application servers that are connected other replicators in the same domain. If the replicator that a component is connected to fails, the component automatically attempts to reconnect to another replicator in the domain and recover any data that was missed while the component was not connected to a replicator.

The default is to define a replication domain for a cluster when creating the cluster. However, replication domains can span across clusters.

Global default settings apply to a given replication domain across a cell. Most default settings tune and control the behavior of replicator entries that are in managed servers across the cell. Such default settings control the use of encryption or the serialization and transferring of objects. Some default settings tune and control how specific product functions, such as session manager and dynamic caching, leverage replication, such as session use of partitions.

For situations that require settings values other than the default, change the values for a given replication domain. Settings include various resource allocation, replication strategies, such as grouping or partitioning, and methods, as well as some security related items.

If you are using replication for HTTP session failover, you might also need to filter where the session replicates. For example, only replicate to two places out of many. The global default settings define the partition size or number of groups and the session manager settings define the groups to which a particular instance belongs.

Filtering is less important if you are using replication to distribute information on data that is no longer valid and actual cached data maintained by dynamic caching. Replication does not occur for failover as much as for data synchronization across a cluster or cell when you likely want to avoid expensive costs for generating data potentially needed across those various servers.

Note that you can filter or segment by using multiple replication domains.

Multi-broker replication domain settings

Use this page to configure a multi-broker replication domain. This administrative console page applies only to replication domains that were created with a previous version of the product. Replication domains use the data replication service (DRS).

To view this administrative console page, click **Environment > Replication domains >** *multibroker_replication_domain_name*.

An application server that is connected to a replicator within a domain can access the same set of data sent out by any application server connected to any other replicator, including the same replicator. Data is not shared across replication domains.

Name:

Specifies a name for the replication domain. The name must be unique within the cell.

Request timeout:

Specifies the number of seconds that a replication domain consumer waits when requesting information from another replication domain consumer before giving up and assuming that the information does not exist. The default is 5 seconds.

Data type	Integer
Units	Seconds
Default	5

DRS partition size:

Specifies the number of groups into which a replication domain is partitioned. By default, data sent by an application server process to a replication domain is transferred to all other application server processes connected to that replication domain. To filter or reduce the number of destinations for the data being sent, partition the replication domain. There should be at least one server listening to every partition. If there are no servers listening on a partition, all the replicas created in that partition are lost because there is no server to cache the objects. The default partition size is 10, and the partition size should be 10 or more to enhance performance.

Partitioning the replication domain is only applicable if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails. Partitioning is not supported for sharing of cached data maintained by web container dynamic caching. As to dynamic caching, all partitions or groups are always active and used for data replication.

When you partition a replication domain, you define the total number of groups or partitions. Use this setting to define the number of groups. Then, when you configure a specific session manager under a web container or as part of an enterprise application or web module, select the partition to which that session manager instance listens and from which it accepts data. To specify the groups to which an application server listens, change the settings for affected servers on a session manager page. In addition, you can set a role or runtime mode for a server. This role or mode affects whether an application server process sends data to the replication domain, receives data, or does both. The default is both to receive and send data.

Data type	Integer
Default	10

Single replica:

Specifies that a single replication of data is made. Use this option only if you are using session manager with memory to memory replication. Enable this option if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails. This option restricts the recipient of the data to a single instance.

gotcha: Do not enable this option on a domain that is using dynamic cache replication.
This setting provides filtering beyond grouping or partitioning. Using this setting, you can choose to have data only sent to one other listening instance in the replication domain.

Default   false

Serialization method:

Specifies the object serialization method to use when replicating data. An administrative concern with replicating Java objects is locating the class definition, especially in a Java Platform, Enterprise Edition (Java EE) environment where class definitions might exist only in certain web modules or enterprise applications. Object serialization methods define whether the processes receiving data also need the class definition.

The options for this setting are OBJECT and BYTES. The default is BYTES.

OBJECT instructs a replicator to write the object directly to the stream. With OBJECT, a replicator must instantiate the object on the receiving side so it must have the class definition.

BYTES instructs a replicator to break down the object into bytes and then send only the bytes across the stream. With BYTES, a replicator does not need to instantiate the object on the receiving side. The BYTES option is useful for failover, where the data is not used at the receiving side and the class definitions do not need to be stored on the receiving side. Or, the option requires that you move class definitions from the web application class path to the system class path.

DRS pool size:

Specifies the size of the pool of resources allocated for communication with its Java Message Service (JMS) transport. You must configure this number to be the same as the DRS partition size. The default is 10.

Pooling replication resources can enhance the performance of the internal data replication service.

DRS pool connections:

Specifies that the domain replication service should create a pool of connections with its Java Message Service (JMS) transport rather than reusing a single connection. You can pool connections when using a single replica or client server environment. You should not pool connections in a peer to peer environment.

The default is to not create a pool of connections for replication.

Replicator entry collection:

Use this page to view and manage replicator entries. Replicator entries are for use only with multi-broker replication domains. Each multi-broker replication domain consists of one or more replicator entries.

To view this administrative console page, click **Environment > Replication domains >** *replication_domain_name* **> Replicator entries**.

Replicator entries are only valid for multi-broker domains, which are replication domains created with a previous version of the product. When you migrate your deployment manager to the current version of the

product, you are no longer be able to create new replicator entries in the administrative console. You can only view and modify settings for replicator entries that were created with the previous version of the product.

Replicator name:

Specifies a name for the replicator entry.

Replicator entry settings:

Use this page to view and configure a replicator entry, or *replicator*. Replicators are used with multi-broker replication domains.

To view this administrative console page, click **Environment > Replication domains >** *replication_domain_name* **> Replicator entries >** *replicator_entry_name*.

Replicators communicate using Transmission Control Protocol/Internet Protocol (TCP/IP). Therefore, you must allocate an IP address and ports for replicators. Use this page to name a replicator and then to allocate an IP address and two ports (replicator and client ports) for the replicator.

Replicator name:

Specifies a name for the replicator entry.

Server:

Specifies the server for which you are defining a replicator. You can view the names of servers that do not already have replicators. You can create a maximum of one replicator on any application server.

Replicator and client host name:

Specifies the IP address, domain name service (DNS) host name with domain name suffix, or just the DNS host name, used by a client to request a web application resource (such as a servlet, JavaServer Pages (JSP) file, or HTML page).

A replicator port and client port share the same host name.

Replicator Port:

Specifies the port for which the replicator is configured to accept messages from other replicators. The port value is used in conjunction with the host name.

The replicator port enables communication among replicators. It provides replicator port to replicator communication. The usual value specified is 7874.

Client Port:

Specifies the port for which the web server is configured to accept client requests. The port value is used in conjunction with the host name.

The client port enables communication between an application server process and a replicator. It provides client port to application server communication. The usual value specified is 7873.

Deleting clusters

Use this task to remove a cluster and all of its cluster members.

Before you begin

Removing a cluster deletes the cluster and all associated cluster members. When you delete a cluster, there is no option to keep certain cluster members or applications that you have installed on any part of the cluster.

In a production environment, avoid deleting clusters that are carrying workload. You can, however, add and remove cluster members during production. When you want to remove a cluster, create a new cluster, adding new members while the old cluster is still operational. After the new cluster is working, remove the cluster members from the old cluster and then delete the cluster.

Tip: If the cluster you are removing has applications or modules mapped to it, remap the modules to another cluster, or create a new cluster and remap the modules to the new cluster, before removing the old cluster. After a cluster to which modules are mapped is deleted, the modules cannot be remapped to another cluster. Therefore, if you do not remap the modules to another cluster before deleting the old one, you must uninstall all of the modules that were mapped to the old cluster, and then reinstall them on a different cluster.

Procedure

1. In the administrative console, click **Servers > Clusters > WebSphere application server clusters**.
2. Make sure the cluster you want to remove is stopped.

   If the cluster is started, stop the cluster.
3. Delete the cluster. Select the cluster you want to delete, and click **Delete**.
4. Click **OK** and then click **Review** to preview your changes.
5. Select **Synchronize changes with Nodes**, and then click **Save** to save your changes.

Results

The cluster and all of the cluster members are deleted.

Deleting specific cluster members

Use this task to remove a cluster member from an existing cluster. Removing a cluster member deletes the associated application server.

About this task

You must delete an application server to remove it from a cluster.

If, in the administrative console, you select **Include cluster members in the collection** as one of your console page preferences for the Application servers page, you can use either the Application servers page or the Cluster members page to delete an application server.

To use the Cluster members page to remove an application server from a cluster:

Procedure

1. In the administrative console, click **Servers > Clusters > WebSphere application server clusters >** .
2. Click the name of the cluster that contains the cluster member that you are removing from the cluster, and then click **Cluster members**.
3. Check the status of the cluster member that you are removing. If the cluster member is started, select the cluster member, and click **Stop**, and then view the Status of this cluster member again, along with any messages or logs to make sure the cluster member stops. You cannot remove a cluster member while it is running.
4. Delete the cluster member. Select the cluster member you want to delete, and click **Delete**.

5. Click **OK** and then click **Review** to preview your changes.
6. Select **Synchronize changes with Nodes**, and then click **Save** to save your changes.

Results

The cluster member is deleted.

Chapter 6. Enabling request-level Reliability Availability and Serviceability (RAS) granularity

You can enable request-level Reliability Availability and Serviceability (RAS) granularity for HTTP, IIOP, optimized local adapter, and certain MDB requests by defining RAS attributes in the workload classification document. With request-level RAS granularity, you can specify RAS attribute values for specific requests, such as a unique dispatch timeout value for all HTTP requests with a URI that ends in .jpg.

About this task

Note: Reliability Availability and Serviceability (RAS) granularity is the ability to assign different RAS attribute values to different sets of requests within the same application server. You can improve the reliability, availability, and serviceability of the application server and the requests it processes by using the request-level RAS granularity capabilities introduced in Version 8.0 of the product.

To implement request-level RAS granularity, develop the workload classification document and convert it to ASCII if you use code page IBM-1047. Use the administrative console to specify the location of the workload classification file. Ensure that the application server recognizes the changed workload classification document by restarting the server or reloading the workload classification file. Use the DISPLAY WORK operator command to display classification information so that you can determine if your classification scheme is classifying the work as you intended.

Procedure

1. Develop the workload classification document. Use the information in the workload classification file topic to create the document. The topic contains examples of the workload classification document, with and without RAS attributes. Use one workload classification document whether you are using it to classify z/OS workload or to implement request-level RAS granularity.

2. If you create the document on a z/OS system in code page IBM-1047, the normal code page for files that exist in the HFS, convert the file to ASCII before you use the file. Use one of the following options to convert a working document into a document that can be used by the server:

   - native2ascii

     This is a utility in the Java SDK that can convert a file from the native code page to the ASCII code page. For example, if you are working on an XML document called x5sr02.classification.ebcdic.xml and you want to create a document called x5sr02.classification.xml, use the following command:

     ```
     /u/userid $ native2ascii \
     x5sr02.classification.ebcdic.xml > x5sr02.classification.xml
     ```

     The command line is split with the backslash (\) character to the next line for publication purposes.

   - iconv

     This is a z/OS utility that can convert files from one designated code page to a different designated code page. For example, if you are working on an XML document called x5sr02.classification.ebcdic.xml and you want to create a document called x5sr02.classification.xml, use the following command. The $ character is the prompt.

     ```
     /u/userid $ iconv -f IBM-1047 -t UTF-8 \
     x5sr02.classification.ebcdic.xml >x5sr02.classification.xml
     ```

     The command line is split with the backslash (\) character to the next line for publication purposes.

   - Create the document on your workstation and then FTP the file to the correct location on the z/OS system in binary format. By using this option, you can also create the `Classification.dtd` file in the same directory as the workload classification document. Then, you can perform an XML validity check on the document before installing it on a server. Use any type of validating parser. For example, you can use WebSphere Application Developer workbench to construct and validate the workload classification document.

3. Specify the location of the workload classification document in the administrative console. Use the `wlm_classification_file` variable to specify the XML file that contains the classification information. In the administrative console, click **Environment > WebSphere variables > New**. You can set the variable at the cell, node, or server instance level. If you specify the variable at the cell or node level, the information must be accessible and applicable to all the servers that inherit the specification from the node or cell.

4. Implement the changes to the file. You can restart the application server or reload the workload classification document without having to restart the application server:

   - Restart the application server.
   - Reload the workload classification document by issuing the following command:

     ```
     MODIFY|F <servername>,RECLASSIFY,FILE='/path/to/newfile.xml'
     ```

   If the workload classification document is not a well formed, valid XML document, it is ignored by the application server and the following message is displayed:

   ```
   BBOJ0085E PROBLEMS ENCOUNTERED PARSING WLM CLASSIFICATION XML FILE (0)
   ```

5. Use the DISPLAY WORK operator command to display classification information. Use this command to determine if your classification scheme is classifying the work as you intended. Issue the following command to display the IIOP, HTTP, INTERNAL, SIP, MDB, and optimized local adapter classification information:

   ```
   MODIFY|F <servername>,DISPLAY,WORK,CLINFO
   ```

   Issue this command against each application server.

   The following example shows a possible result of issuing the new operator command:

   ```
   00- SY1  f bbos001,display,work,clinfo
       SY1  BBOJ0129I: The /tmp/wlm4.class.xml workload classification file was loaded at
       2009/07/14 19:33:35.297 (GMT).
       SY1  BBO000281I CLASSIFICATION COUNTERS FOR IIOP WORK
       SY1  BBO000282I CHECKED 0, MATCHED 0, USED 0, COST 2, DESC: IIOP root
       SY1  BBO000282I CHECKED 0, MATCHED 0, USED 0, COST 4, DESC: leotag
       SY1  BBO000282I CHECKED 0, MATCHED 0, USED 0, COST 3, DESC: byetag
       SY1  BBO000282I CHECKED 0, MATCHED 0, USED 0, COST 4, DESC: hellotag
       SY1  BBO000283I FOR IIOP WORK: TOTAL CLASSIFIED 0, WEIGHTED TOTAL COST 0
       SY1  BBO000281I CLASSIFICATION COUNTERS FOR HTTP WORK
       SY1  BBO000282I CHECKED 2, MATCHED 2, USED 0, COST 2, DESC: HTTP root
       SY1  BBO000282I CHECKED 2, MATCHED 2, USED 0, COST 4, DESC: plantta4
       SY1  BBO000282I CHECKED 2, MATCHED 1, USED 1, COST 3, DESC: giftag4
       SY1  BBO000282I CHECKED 1, MATCHED 1, USED 1, COST 4, DESC: jpgtag4
       SY1  BBO000283I FOR HTTP WORK: TOTAL CLASSIFIED 2, WEIGHTED TOTAL COST 7
       SY1  BBO000188I END OF OUTPUT FOR COMMAND DISPLAY,WORK,CLINFO
   ```

   An explanation of the command output:

   BBOJ0129I: The *file* **workload classification file was loaded at** *time*.
   The message indicates the workload classification file currently active and the time that it was loaded.

   BBO000281I CLASSIFICATION COUNTERS FOR *type* **WORK**
   The header message for messages that display the usage of the workload classification rules. The value of *type* can be `HTTP`, `IIOP`, `INTERNAL`, `SIP`, `OLA`, or `MDB`.

   BBO000282I CHECKED *n1*, **MATCHED** *n2*, **USED** *n3*, **COST** *n4*, **DESC:** *text*
   This message displays information about a particular rule in the workload classification. This message displays the following information:

   - *n1* - The number of times the rule has been examined.
   - *n2* - The number of times that this rule has been matched by the request.
   - *n3* - The number of times that this rule has been used.
   - *n4* - The cost of using the rule, or the number of compares that are required to determine if this rule is the correct rule to use.

- *text* - The descriptive text from the classification rule so that you can tell which classification rule is being displayed.

BB000283I FOR *type* **WORK: TOTAL CLASSIFIED** *n1***, WEIGHTED TOTAL COST** *n2*

This message shows the summary information for the IIOP, HTTP, INTERNAL, SIP, MDB, or optimized local adapter work classification. This message displays the following information:

- *type* - The type of work that is being displayed. The value must be IIOP, HTTP, INTERNAL, SIP, MDB, or OLA.
- *n1* - The number of requests that were classified using the classification rules.
- *n2* - The weighted total cost, calculated by taking the number of times that each rule was used multiplied by the cost, or number of rule compares that were done, of using the rule and adding those up across all the rules.

The total cost *n2* divided by the total number of requests classified *n1* equals the cost of using the table. The closer that the value is to one, the lower the cost of using the defined rules. A value of 1 indicates that there is just the default classification, so no requests match it.

6. Repeat these steps until you achieve the RAS granularity that you want.

Results

You have used the workload classification document to implement request-level RAS granularity.

RAS granularity for HTTP, IIOP, MDB, and optimized local adapter requests

Reliability Availability and Serviceability (RAS) granularity is the ability to assign different RAS attribute values to different sets of requests within the same application server. You can improve the reliability, availability, and serviceability of the application server and the requests it processes by using the finely grained RAS granularity capabilities introduced in version 8.0 of the product.

The application server applies a set of RAS attributes to all the requests it processes. RAS attributes affect the reliability, availability, and serviceability of the server and the requests. Examples of RAS attributes include timeout values, timeout actions, trace settings, and so on.

RAS granularity is the ability to assign different sets of RAS attribute values to different sets of requests. The fineness of RAS granularity depends on how uniquely the application server can distinguish one set of requests from another. Before Version 8.0 of the product, RAS granularity was limited to a per-server basis or, for a few RAS attributes, a per-protocol basis.

Per-server RAS granularity means that a single set of RAS attribute values is defined in the server configuration. This single set of RAS attribute values applies to all requests that the application server processes. An example of a per-server RAS attribute is the trace setting. You can define only one trace setting to an application server. This trace setting applies to all requests that the application server processes.

Per-protocol RAS granularity means that multiple sets of RAS attribute values can be defined in the server configuration, one set for each protocol. The application server divides requests into sets based on the request protocol, such as the HTTP protocol or the IIOP protocol. The application server then applies the set of RAS attribute values defined for that protocol to the requests for that protocol. An example of a per-protocol RAS attribute is the dispatch timeout. You can define the dispatch timeout for IIOP requests by using the control_region_wlm_dispatch_timeout property and for HTTP requests by using the protocol_http_timeout_output property.

Starting with Version 8.0 of the product you can achieve finer RAS granularity by defining RAS attribute values on a per-workload-classification basis. Per-workload-classification RAS granularity means multiple

sets of RAS attribute values can be defined in the server configuration, one for each workload classification element in the workload classification file. The application server classifies requests based on the workload classification elements defined in the workload classification file. The application server then applies the set of RAS attribute values defined for a workload classification element to the requests that are classified under that workload classification element.

The fineness of RAS granularity that a per-workload-classification scheme achieves depends on how granular the application server can classify requests. The application server classifies a request using various attributes of the request. The following list gives examples of how granular the application server classifies requests.

- For HTTP requests, the application server can classify requests as granular as the specific uniform resource identifier (URI).

   The application server can assign a unique set of RAS attribute values to HTTP requests on a per-URI basis.

- For IIOP requests, the application server can classify requests as granular as the specific Enterprise JavaBeans (EJB) method being invoked.

   The application server can assign a unique set of RAS attribute values to IIOP requests on a per-EJB-method basis.

- For optimized local adapter requests, the application server can classify requests as granular as the service name or JNDI home name of the EJB application to be driven.

   The application server can assign a unique set of RAS attribute values to optimized local adapter requests on a per-service name or a per-JNDI home name basis.

- For MDB requests, the application server can classify requests as granular as the selector clause in the MDB deployment descriptor.

   The application server can assign a unique set of RAS attribute values to MDB requests on a per-selector basis.

Precedence for modify command parameters, request-level RAS attributes, and server-wide properties

Three ways exist to define Reliability Availability and Serviceability (RAS) attribute values: server-wide properties, request-level RAS attributes, or modify command parameters. Modify command parameters have precedence over associated RAS attributes and server-wide properties, while the RAS attributes have precedence over the associated server-wide properties.

The following list describes the order of precedence:

- The modify command takes precedence over the associated RAS attributes and server-wide properties.

   After you invoke a modify command, the command applies to all requests, regardless of the server-wide properties or the classification attributes that you have defined.

- The request-level RAS attributes defined in the workload classification file take precedence over the associated server-wide property settings.

   – If you specify request-level RAS attribute values on the HTTP classification element, the request-level RAS attribute values override the server-wide property settings for HTTP requests.

   – If you specify request-level RAS attribute values on the IIOP classification element, the request-level RAS attribute values override the server-wide property settings for IIOP requests.

   – If you specify request-level RAS attribute values on the optimized local adapter classification element, the request-level RAS attribute values override the server-wide property settings for optimized local adapter requests.

   – If you specify request-level RAS attribute values on the classification element for the message-driven bean (MDB) whose listener exists in the control region, the request-level RAS attribute values override the server-wide property settings for those MDB requests.

– If you specify request-level RAS attribute values with the activation specification in the control adjunct region, the request-level RAS attribute values override the server-wide property settings for those MDB requests.

The server-wide property settings continue to apply to any protocols that the request-level RAS attribute does not apply.

A `RESET` value is available on all modify commands that have corresponding server-wide properties that also have associated RAS attributes. The `RESET` value deactivates the modify command override. After a reset is invoked, the server returns to the server-wide properties and RAS attributes that were in effect before the modify override became activated.

Remember: How a RAS attribute takes effect depends on how you define the RAS attribute.
- If you define the RAS attribute as a server-wide property, you must restart the application server.
- If you define the RAS attribute as a request-level RAS attribute, you have two options. You can restart the application server or reload the workload classification document without having to restart the application server.
- If you define the RAS attribute with the modify command, the modify command takes effect dynamically without requiring a restart of the server.

Several of the RAS attributes added to the workload classification file are associated with server-wide environment properties which are associated with modify commands. For example, the SMF_request_activity_enabled request-level RAS attribute is associated with the server_SMF_request_activity_enabled server-wide property, which in turn is associated with the SMF,REQUEST modify command.

The SMF_request_activity_enabled request-level RAS attribute specifies whether the application server collects System Management Facilities (SMF) 120 subtype 9 records for requests. The requests can be HTTP requests, IIOP requests, optimized local adapter requests, or MDB requests that have the listener in the control region. The request-level RAS attribute is coded on the HTTP classification element for HTTP requests, on the IIOP classification element for IIOP requests, on the optimized local adapter classification element for optimized local adapter requests, and on the MDB classification element for MDB requests.

The server_SMF_request_activity_enabled server-wide property specifies whether the application server collects SMF 120 subtype 9 records for all requests that the application server processes.

The SMF_request_activity_enabled request-level RAS attribute overrides the server_SMF_request_activity_enabled server-wide property for any HTTP requests, IIOP requests, optimized local adapter requests, or MDB requests that have the listener in the control region. Like the server-wide property, the SMF,REQUEST modify command specifies whether the application server collects SMF 120 subtype 9 records for all requests that the application server processes. However, the modify command dynamically overrides the settings for the request-level RAS attribute and the server-wide property without requiring a restart of the server.

The example shows the following capabilities:
- The server-wide property of server_SMF_request_activity_enabled is in effect.
- The SMF_request_activity_enabled request-level RAS attribute can override the server_SMF_request_activity_enabled server-wide property.
- The SMF,REQUEST modify command can override both the server-wide property and the request-level RAS attribute.
- The reset option on the SMF request modify command can deactivate the modify command override.

Here are the particulars for each of the capabilities:

- You set the server-wide property of server_SMF_request_activity_enabled to 1 on the administrative console. The RAS attributes and the modify command do not override the property. All requests that the server processes collect SMF 120 subtype 9 records.

- You next create a workload classification document, which defines a single http_classification_info element:

```
<http_classification_info
    uri="/PlantsByWebSphere/*"
    SMF_request_activity_enabled="0"
  />
```

  The HTTP classification element specifies that the SMF_request_activity_enabled request-level RAS attribute is set to 0. This request-level RAS attribute overrides the server_SMF_request_activity_enabled server-wide property that is set to 1. None of the HTTP requests with a URI that begins with /PlantsByWebSphere/ collect SMF 120 subtype 9 records. All HTTP requests with URI that do not begin with /PlantsByWebSphere/, and all requests from non-HTTP protocols, observe the server-wide property setting. Those requests continue to collect SMF 120 subtype 9 records.

- You now issue the following modify command:

  `MODIFY BBOS001,SMF,REQUEST,ON`

  The modify command specifies that SMF 120 subtype 9 record collection is on. The modify command overrides the SMF_request_activity_enabled request-level RAS attribute which is off since it is set to 0. Therefore, all HTTP requests with URI that begin with /PlantsByWebSphere/ collect SMF 120 subtype 9 records. The modify command also overrides the server_SMF_request_activity_enabled server-wide property which is on since it is set to 1. However, the server_SMF_request_activity_enabled server-wide property and the modify command override are both on. Therefore, all requests that observed the server-wide property before you issued the modify command experience no change in behavior. Those requests continue to collect SMF 120 subtype 9 records.

  You next issue the following modify command:

  `MODIFY BBOS001,SMF,REQUEST,OFF`

  The OFF value means that SMF 120 subtype 9 record collection is turned off. The modify command overrides the SMF_request_activity_enabled request-level RAS attribute. However, the request-level RAS attribute is also off since it is set to 0. So, no change in behavior occurs. Both the modify command and the request-level RAS attribute turned off the collection of SMF 120 subtype 9 record collection for HTTP requests with URI that begin with /PlantsByWebSphere/. The modify command also overrides the server_SMF_request_activity_enabled server-wide property which is on since it is set to 1. Therefore, all other HTTP requests and all non-HTTP requests, that had observed the server_SMF_request_activity_enabled server-wide property that is set to 1 no longer collect SMF 120 subtype 9 records.

- You next issue the following modify command:

  `MODIFY BBOS001,SMF,REQUEST,RESET`

  The RESET value causes the modify command override to be deactivated. This means that all HTTP requests with a URI that begins with /PlantsByWebSphere/ go back to observing the SMF_request_activity_enabled request-level RAS attribute set to 0. None of the HTTP requests with a URI that begins with /PlantsByWebSphere/ collect SMF 120 subtype 9 records. All other HTTP requests, and all non-HTTP requests, go back to observing the server_SMF_request_activity_enabled server-wide property that is set to 1 by collecting SMF 120 subtype 9 records.

Workload classification file

The workload classification document is a common XML file that classifies inbound HTTP, IIOP, message-driven bean (MDB), Session Initiation Protocol (SIP), optimized local adapter, and mediation work for the z/OS workload manager.

Usage notes

This topic contains examples of the workload classification file with and without Reliability Availability and Serviceability (RAS) attributes. RAS attributes allow you to achieve request-level RAS granularity for HTTP requests, IIOP requests, MDB requests, and optimized local adapter requests. You specify these attributes on the http_classification_info element, the iiop_classification_info element, the classificationentry element, the sib_classification_info element, the wmqra_classification_info element, and the ola_classification_info element in the workload classification file.

You use the workload classification file when you complete the tasks for classifying z/OS workload or enabling request-level RAS granularity.

Required elements

<?xml version="1.0" encoding="UTF-8"?>
> Indicates that the workload classification document must be saved in ASCII to be processed by the application server. This statement is required.

<!DOCTYPE Classification SYSTEM "Classifications">
> Provides the XML parser with the name of the DTD document that is provided with the product, and that is used to validate the workload classification document. The workload classification document that you write must follow the rules that are described in this DTD. You must add this statement to the workload classification document.

Classification
> `<Classification schema_version="1.0">`
>
> Indicates the root of the workload classification document. Every workload classification document must begin and end with this element. The schema_version attribute is required. The only supported schema_version is 1.0. The Classification element contains one or more InboundClassification elements. For inbound service integration work, the Classification element can also contain up to two SibClassification elements. If classifying inbound messages for delivery to message-driven beans using WebSphere MQ messaging provider activation specifications, the Classification element can contain one or more WMQRAClassification elements.

> **InboundClassification**
>> `<InboundClassification type="iiop | http | mdb |internal | sip | ola"`
>> `schema_version="1.0" default_transaction_class="value">`
>>
>> Use the following rules when using the InboundClassification element:
>> - The **type** attribute is required. The value must be `internal`, `iiop`, `http`, `mdb`, `sip`, or `ola`. Only one occurrence of an InboundClassification element can occur in the document for each type. There can be up to five InboundClassification elements in a document. The types do not have to be specified in a certain order in your classification document.
>> - The **schema_version** attribute is required. The value must be set to 1.0.
>> - The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " ").
>> - The InboundClassification elements cannot be nested. Each InboundClassification element must end before the next InboundClassification element or SibClassification element can begin.

> **SibClassification**
>> `<SibClassification type="jmsra | destinationmediation" schema_version="1.0"`
>> `default_transaction_class="value">`
>>
>> Use the following rules when using the SibClassification element:

- The **type** attribute is required. The value must be jmsra or destinationmediation. There can be at most one SibClassification element in the document for each type. The types do not have to be specified in a certain order in your classification document.
- The **schema_version** attribute is required. The value must be set to 1.0.
- The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " ").
- The SibClassification elements cannot be nested. Each SibClassification element must end before the next InboundClassification element or SibClassification element can begin.

WMQRAClassification

```
<WMQRAClassification schema_version="1.0" default_transaction_class="value">
```

The following rules apply to the WMQRAClassification element:
- The **schema_version** attribute is required. The value must be set to 1.0.
- The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class.
- The WMQRAClassification elements cannot be nested. Each WMQRAClassification element must end before any other classification elements can begin.

The rules and XML statements for classifying different types of work are similar, but there is slightly different syntax for each type. For more information about the syntax for each type of work, see the following sections:

InboundClassification

SibClassification

WMQRAClassification

Internal Classification

The InboundClassification element with the attribute type="internal" defines the section of the document that is applicable to internal work, such as requests that are dispatched in a servant, that originate in the owning controller. An example of this element follows:

```
<InboundClassification  type="internal" schema_version="1.0"
                  default_transaction_class="value1">
```

If an InboundClassification element with the type="internal" attribute is not specified, internal work is classified using the rules that are specified for IIOP work.

IIOP Classification

The InboundClassification element with the attribute type="iiop" defines the section of the document that is applicable to IIOP classification. An example of this element follows:

```
<InboundClassification  type="iiop" schema_version="1.0"
                        default_transaction_class="value1">
```

You can classify IIOP work based on the following Java Platform, Enterprise Edition (Java EE) application artifacts:

- Application name

    The name of the application that contains the enterprise beans. It is the display name of the application, which might not be the name of the `.ear` file that contains all the artifacts.

- Module name

    The name of the Enterprise JavaBeans(EJB) `.jar` file that contains one or more enterprise beans. There can be multiple EJB `.jar` files in an `.ear` file.

- Component name

    The name of the EJB that is contained in a module (or EJB `.jar` file). There can be one or more enterprise beans contained in a `.jar` file.

- Method name

    The name of a remote method on an EJB.

Classify IIOP work in various applications at any of these levels by using the iiop_classification_info element.

iiop_classification_info

```
<iiop_classification_info transaction_class="value1"
                          application_name="value2"
                          module_name="value3"
                          component_name="value4"
                          method_name="value5"
                          description="value6"
                          dispatch_timeout="value7"
                          queue_timeout_percent ="value8"
                          request_timeout="value9"
                          stalled_thread_dump_action="traceback"
                          cputimeused_limit="value11"
                          cputimeused_dump_action="traceback"
                          dpm_interval="value13"
                          dpm_dump_action="traceback"
                          SMF_request_activity_enabled="1"
                          SMF_request_activity_timestamps="1"
                          SMF_request_activity_security="1"
                          SMF_request_activity_CPU_detail="1"
                          classification_only_trace="1"
                          message_tag="value20">
```

With the iiop_classification_info element, you can build filters based on the application, module, component, and method names to assign TCLASS values, RAS attributes, or both to inbound requests. Use the following rules when using the iiop_classification_info element:

transaction_class

The transaction_class attribute is optional. If the attribute is not defined, it inherits the transaction class of its parent. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " "). By specifying a null or blank string, you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.

application_name, module_name, component_name, and method_name

The attributes application_name, module_name, component_name, and method_name can be used as you need them. These attributes act as selectors or filters that either assign a transaction class or allow a nested iiop_classification_info element to assign the transaction class. You can specify the values of these attributes in the following ways:

- The exact name of the application, module, component, or method.
- A wild-card value. You can place an asterisk (∗) anywhere in a string to indicate that any string that starts with the string preceding the asterisk and ends with the string that follows the asterisk is considered a match. If the asterisk is at the end of the string, any string that starts with the string preceding the asterisk is considered a match.

  Examples:
  - The string `Mar*61`, matches `Mar61`, `March61`, and `Mar20early61`, but does not match `March81` or MAR61.
  - The string `MAR*` matches `MARCH`, `MAR61`, and `MARS`, but does not match `Mar61` or `MAY61`.

  gotcha: The value comparisons that are performed are case sensitive.

  You can use any combination of these attributes to make a classification filter. However, use only the granularity that is required. For example, if there is only one application on the application server, the classification rules do not need to specify the application_name attribute.

RAS attributes

You can specify the following RAS attributes on the iiop_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the control_region_wlm_dispatch_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_iiop_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_iiop_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump,` `javacore,` `heapdump,` `traceback,` `javatdump,` and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump,` `javacore,` `heapdump,` `traceback,` `javatdump,` and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are 0 (no) and 1 (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to 1, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Classification filters

- The iiop_classification_info elements can be nested in a hierarchical manner. By nesting the elements, you can create classification filters that are based on the attribute values. The following filter classifies requests on the EJB1 and EJB2 enterprise beans in the MyAPP1 application:

```
<iiop_classification_info transaction_class="FAST"
                          application_name="MyAPP1"
                          component_name="EJB1" />
<iiop_classification_info transaction_class="SLOW"
                          application_name="MyAPP1"
                          component_name="EJB2" />
```

The following filter also classifies requests on EJB1 and EJB2 in the MyAPP1 application, but also classifies requests on any other EJB in the application:

```
<iiop_classification_info transaction_class="MEDIUM"
                          application_name="MyAPP1">
    <iiop_classification_info transaction_class="FAST"
                              component_name="EJB1" />
    <iiop_classification_info transaction_class="SLOW"
                              component_name="EJB2" />
</iiop_classification_info>
```

- If you specify an attribute value that conflicts with the attribute value of the parent element, the lower-level filter is negated. An example of a child value that conflicts with the attribute value of the parent element follows:

```
<iiop_classification_info transaction_class="FAST"
                          application_name="MyAPP1">
    <iiop_classification_info transaction_class="SLOW"
                              application_name="MyAPP2" />
</iiop_classification_info>
```

In this example, EJB Requests in MyAPP2 would never be assigned to transaction class "SLOW" because the higher level filter only allows IIOP requests for application_name="MyAPP1" to be passed through to the lower-level filter.

- The first filter at a specific level that matches the attributes of the request is used, not the best or most restrictive filter. Therefore, the order that you specify filters is important.

```
<iiop_classification_info transaction_class="FAST"
                          application_name="MyAPP" />
    <iiop_classification_info transaction_class="SLOW"
                              component_name="*" />
    <iiop_classification_info transaction_class="MEDIUM"
                              component_name="MySSB" />
</iiop_classification_info>
```

In the preceding example, all the IIOP requests that are processed by enterprise beans in the MyAPP application are assigned a TCLASS value of SLOW. This assignment is done for any requests to the MySSB enterprise as well. Even though MySSB is assigned a transaction class, the filter is not applied because the first filter was applied and was assigned a TCLASS value of SLOW. The remaining list of filters at the same level is ignored.

- The description field is optional. However, you should use a description on all iiop_classification_info elements. The description string prints as part of the operator command support so you can identify the classification rules that are being used. Keep your descriptions reasonably short because they are displayed in the MVS console.

HTTP classification

The InboundClassification element with the attribute type="http" defines the section of the document that is applicable to HTTP classification. An example of this element follows:

```
<InboundClassification  type="http"
                        schema_version="1.0"
                        default_transaction_class="value1">
```

HTTP work can be classified based on the following J2EE artifacts:

- Virtual host name

  Specifies the host name in the HTTP header to which the inbound request is being sent.
- Port number

  Specifies the port on which the HTTP catcher is listening.
- URI (Uniform Resource Identifier)

  The string that identifies the web application.

You can classify HTTP work in various applications at any of these levels by using the http_classification_info element.

```
<http_classification_info transaction_class="value1"
                          host="value2"
                          port="value3"
                          uri="value4"
                          description="value5"
                          dispatch_timeout="value6"
                          queue_timeout_percent ="value7"
                          request_timeout="value8"
                          stalled_thread_dump_action="traceback"
                          cputimeused_limit="value10"
                          cputimeused_dump_action="traceback"
                          dpm_interval="value12"
                          dpm_dump_action="traceback"
                          SMF_request_activity_enabled="1"
                          SMF_request_activity_timestamps="1"
                          SMF_request_activity_security="1"
                          SMF_request_activity_CPU_detail="1"
                          classification_only_trace="1"
                          message_tag="value19"
                          timeout_recovery="value20">
```

With the http_classification_info element, you can build filters based on the host, port, and URI to assign TCLASS values, RAS attributes, or both to inbound requests. Use the following rules when you use the http_classification_info element:

transaction_class

The transaction_class attribute is optional. If the attribute is not defined, it inherits the transaction class of its parent. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " "). By specifying a null or blank string,

you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.

host, port, and uri

The attributes of host, port, and uri can be used as you need them. These attributes act as selectors or filters that either assign a transaction class or allow a nested http_classification_info element to assign the transaction class. You can specify the values of these attributes in the following ways:

- The exact name of the host, port, or URI.
- Any value. To specify a match to any value, use the asterisk (*) symbol.
- A wild card value. You can place an asterisk (*) anywhere in a string to indicate that any string that starts with the string preceding the asterisk and ends with the string that follows the asterisk is considered a match. If the asterisk is at the end of the string, any string that starts with the string preceding the asterisk is considered a match.

  Examples:
  - The string `Mar*61`, matches `Mar61`, `March61`, and `Mar20early61`, but does not match `March81`.
  - The string `MAR*` matches `MARCH`, `MAR61`, and `MARS`, but does not match `Mar61` or `MAY61`.

  gotcha: The value comparisons that are performed are case sensitive.

Use any or all these attributes to make a classification filter. Only use the granularity that is required. For example, if there is only one application on the application server, the classification rules do not need to specify the uri attribute.

RAS attributes

You can specify the following RAS attributes on the http_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the following server properties:

HTTP protocol_http_timeout_output

HTTPS
protocol_https_timeout_output

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the following server properties:

HTTP control_region_http_queue_timeout_percent

HTTPS
control_region_https_queue_timeout_percent

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the following server properties:

HTTP server_region_http_stalled_thread_dump_action

HTTPS
server_region_https_stalled_thread_dump_action

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are 0 (no) and 1 (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to 1, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define classification_only_trace="1". Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines classification_only_trace="1", then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

timeout_recovery

Specifies the action that the server takes when a request that the classification element classified exceeds its dispatch timeout. Specify the dispatch timeout on the dispatch_timeout attribute.

Valid values are servant and session.

When the attribute is set to servant, the servant address space processing the request terminates with an ABEND EC3 and a reason code 04130007. The controller address space sends a failure response to the client, then closes the socket associated with the request.

When the attribute is set to session, the controller address space sends a failure response to the client, then closes the socket associated with the request. The Servant address space is not terminated. The request is not disrupted, but instead is permitted to run to completion.

The attribute overrides the following server properties:

HTTP   protocol_http_timeout_output_recovery

HTTPS

protocol_https_timeout_output_recovery

Classification filters

- You can nest the http_classification_info elements in a hierarchical manner. You can construct filters based on attribute names. Consider the two following filters:

```
<http_classification_info transaction_class="FAST"
                          host="MyVHost1.com"
                          uri="/MyWebApp1/*" />
<http_classification_info transaction_class="SLOW"
                          host="MyVHost2.com"
                          uri="/MyWebApp2/*" />

<http_classification_info transaction_class="MEDIUM"
                          host="MyVHost1.com">
    <http_classification_info transaction_class="FAST"
                          uri="/MyWebApp1/*" />
    <http_classification_info transaction_class="SLOW"
                          uri="/MyWebApp2/*" />
</http_classification_info>
```

Both filters classify requests to web applications that are identified by context roots /MyWebApp1 and /MyWebApp2 in the application server that is hosting web applications for virtual host MyVHost1.com. However, the second filter also classifies requests on any other context root in the application server.

- Specifying an attribute name that is different from the attribute value of the parent element effectively negates the lower-level filter. For example,

```
<http_classification_info transaction_class="FAST"
                          uri="/MyWebApp1/*">
    <http_classification_info transaction_class="SLOW"
                          uri="/MyWebApp2">
    </http_classification_info>
</http_classification_info>
```

This example would never result in web applications with a context root of /MyWebApp2 being assigned to the transaction class SLOW. The high-level filter only allows HTTP requests with a context root of /MyWebApp1/* to be passed to a lower-level filter.

- The first filter that is at a specific level is used, not the best or most restrictive filter. Therefore, the order of the filters at each level is important. For example:

```
<http_classification_info transaction_class="FAST"
                          host="MyVHost.com" />
    <http_classification_info transaction_class="SLOW"
                          uri="*" />
    <http_classification_info transaction_class="MEDIUM"
                          uri="/MyWebAppX/*" />
</http_classification_info>
```

In this example, HTTP requests processed by the application server by the virtual host "MyVHost.com" are assigned a TCLASS value of SLOW. Even requests to the web application with context root /MyWebAppX are assigned a TCLASS value of SLOW because the filter was not applied. The first filter that matches is used for the TCLASS assignment, and the remainder of the filters at the same level is ignored.

- The description field is optional, however, you should use it on all the http_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

MDB classification

The InboundClassification element with the attribute type="mdb" defines the section of the document that applies to work for EJB 2.0 message-driven beans (MDBs) deployed with listener ports. An example of this element follows:

```
<InboundClassification  type="mdb"
                        schema_version="1.0"
                        default_transaction_class="qrs">
```

Each InboundClassification element can contain one or more endpoint elements with a messagelistenerport type defined. Define one endpoint element for each listener port that is defined in the server that you want to associate transaction classes with the message-driven bean. An example of the endpoint element follows:

```
<endpoint  type="messagelistenerport"
           name="IPVListenerPort"
           defaultclassification="MDBX"
           description="ABC">
```

Use the following rules when defining your endpoint elements:

- The type attribute must always equal messagelistenerport.
- The name attribute corresponds to the listener for your endpoint element. The value of the name attribute must be the name of the listener port that is specified in the administration console for the server.
- The defaultclassification element is the default transaction class that is associated with the message-driven beans. The value of this attribute overrides the default transaction classification value.
- The description field is optional, however, you should use it on all the endpoint elements. The description is displayed when you monitor the transaction classes in the MVS console.

Each endpoint element can have zero, one, or more classificationentry elements. An example of a classification entry element follows:

```
<classificationentry  selector="Location=&apos;East&apos;"
                       classification="MDB2"
                       description="XYZ"
                       dispatch_timeout="value1"
                       queue_timeout_percent ="value2"
                       request_timeout="value3"
                       stalled_thread_dump_action="traceback"
                       cputimeused_limit="value5"
                       cputimeused_dump_action="traceback"
                       dpm_interval="value7"
                       dpm_dump_action="traceback"
                       SMF_request_activity_enabled="1"
                       SMF_request_activity_timestamps="1"
                       SMF_request_activity_security="1"
                       SMF_request_activity_CPU_detail="1"
                       classification_only_trace="1"
                       message_tag="value14"/>
```

selector

> Use the selector attribute of the classificationentry element to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor. Use the following rules when defining your classificationentry elements:
>
> - The value of the selector attribute must match exactly to the selector clause in the MDB deployment descriptor.
> - The value of the selector attribute must have the correct syntax for an XML document. You must replace the < symbol with the < entity reference and the and > symbol with the > entity reference. Similarly, if you use an apostrophe or quotation mark, use the ' and " entity references.

classification

> The classification attribute is optional. If the attribute is not defined, it inherits the classification of its parent. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " "). By specifying a null or blank string, you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.

RAS attributes

You can specify the following RAS attributes on the classificationentry element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the control_region_mdb_request_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_mdb_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_mdb_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are 0 (no) and 1 (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to 1, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Optimized local adapter classification

The InboundClassification element with the attribute type="ola" defines the section of the document that is applicable to optimized local adapter classification. An example of this element follows:

```
<InboundClassification  type="ola" schema_version="1.0"
                        default_transaction_class="value1"
>
```

You can classify optimized local adapter work by adding a section for each EJB application that uses the service name or Java Naming and Directory Interface (JNDI) home name. You can use a wildcard for the JNDI home name.

Classify optimized local adapter work in various applications at any of these levels by using the ola_classification_info element.

ola_classification_info

```
<ola_classification_info  transaction_class="value1"
                          propagate_transaction_name="value2"
                          service_name="value3"
                          description="value4"
                          dispatch_timeout="value5"
                          queue_timeout_percent ="value6"
                          request_timeout="value7"
                          stalled_thread_dump_action="traceback"
                          cputimeused_limit="value9"
                          cputimeused_dump_action="traceback"
                          dpm_interval="value11"
                          dpm_dump_action="traceback"
                          SMF_request_activity_enabled="1"
                          SMF_request_activity_timestamps="1"
                          SMF_request_activity_security="1"
                          SMF_request_activity_CPU_detail="1"
                          classification_only_trace="1"
                          message_tag="value18">
```

With the ola_classification_info element, you can build filters based on the service or JNDI name. Use the name to assign TCLASS values, RAS attributes, or both to inbound requests. Use the following rules when using the ola_classification_info element:

transaction_class
The transaction_class attribute is optional. If the attribute is not defined, it inherits the transaction class of its parent. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " "). By specifying a null or blank string, you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.

propagate_transaction_name
The propagate_transaction_name attribute is optional. If the attribute is specified, the string value must be `true` or `false`. By specifying a value of `true`, the WLM service class from Customer Information Control System (CICS) is propagated to the application server on each request or on each matching request if the service_name filter is specified. The work dispatched in the application server through the optimized local adapter runs under the same service class as the client request.

service_name
The service_name attribute is optional. This attribute acts as a selector or filter that either assigns a transaction class or allows a nested ola_classification_info element to assign the transaction class. You can specify the value of this attribute in the following ways:

- The exact service name or JNDI home name of the EJB application to be driven.
- A wild-card value. You can place an asterisk (*) anywhere in a string to indicate that any string that starts with the string preceding the asterisk and ends with the string that follows the asterisk is considered a match. If the asterisk is at the end of the string, any string that starts with the string preceding the asterisk is considered a match.

Examples

– `service_name="ejb/mySecondBean"`

– `service_name="ejb/my*Bean"`

– `service_name="ejb/my*"`

– `service_name="ejb/security/*"`

> **CAUTION:** The value comparisons that the application server performs are case sensitive.

RAS attributes

You can specify the following RAS attributes on the ola_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

> Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.
>
> The attribute overrides the control_region_wlm_dispatch_timeout server property.

queue_timeout_percent

> Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.
>
> The attribute overrides the control_region_iiop_queue_timeout_percent server property.

request_timeout

> Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.
>
> The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

> Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.
>
> The attribute overrides the server_region_iiop_stalled_thread_dump_action server property.

cputimeused_limit

> Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.
>
> The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

> Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.
>
> Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to 1, then classification level tracing is in effect for the application server. The only threads that generate trace records

are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Classification filters

- The ola_classification_info elements can be nested in a hierarchical manner. By nesting the elements, you can create classification filters that are based on the attribute values.

```
<ola_classification_info transaction_class="MEDIUM"
                         service_name="ejb/my*Bean">
    <ola_classification_info transaction_class="FAST"
                             service_name="ejb/myFastBean" />
    <ola_classification_info transaction_class="SLOW"
                             service_name="ejb/mySlowBean" />
</ola_classification_info>
```

- The description field is optional. However, use a description on all ola_classification_info elements. The description string prints as part of the operator command support so you can identify the classification rules that are being used. Keep your descriptions reasonably short because they are displayed in the MVS console.

SIP Classification

The InboundClassification element with the attribute type="sip" defines the section of the document that sets the default transaction class for Session Initiation Protocol (SIP) requests. An example of this element follows:

```
<InboundClassification  type="sip" schema_version="1.0"
                        default_transaction_class="value1">
```

JMS RA classification

The SibClassification element with the attribute type="jmsra" defines the section of the document that applies to work for message-driven beans (MDBs) deployed against JCA 1.5-compliant resources for use with the JCA resource adapter (RA) of the default messaging provider. An example of this element follows:

```
<SibClassification  type="jmsra"
                    schema_version="1.0"
                    default_transaction_class="a">
```

Each SibClassification element can contain one or more sib_classification_info elements. An example of a classification entry element follows:

```
<sib_classification_info
                transaction_class="sibb"
                selector="user.Location=&apos;East&apos;"
                bus="bigrred"
                destination="abusqueue"
                description="Some words"
                discriminator="XPath Expression"
                dispatch_timeout="value1"
                queue_timeout_percent ="value2"
                request_timeout="value3"
```

```
stalled_thread_dump_action="traceback"
cputimeused_limit="value5"
cputimeused_dump_action="traceback"
dpm_interval="value7"
dpm_dump_action="traceback"
SMF_request_activity_enabled="1"
SMF_request_activity_timestamps="1"
SMF_request_activity_security="1"
SMF_request_activity_CPU_detail="1"
classification_only_trace="1"
message_tag="value14"/>
```

selector

> Use the selector attribute of the sib_classification_info element to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor. Use the following rules when defining your sib_classification_info elements:
>
> - The value of the selector attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification, but it can operate on SIMessage messages (more than JMS messages). The syntax can select on system properties (including JMS headers, JMSX properties, and JMS_IBM_properties) and user properties (which must be prefixed by ".user" - for example, for the user property "Location", the selector would specify "user.Location" as shown in the preceding example). For more information, see the topic on working with message properties.
>
> - The value of the selector attribute must have the correct syntax for an XML document. You must replace the < symbol with the < entity reference and the > symbol with the > entity reference. Similarly, if you use an apostrophe or quotation mark, use the ' and " entity references.

bus The name of the service integration bus on which the target destination is assigned. The classification applies to the bus named by this property, or to any bus if you do not specify this property. The destinations to which the classification applies depend on your use of the destination property.

destination

> The name of the target bus destination to which the message has been delivered. This is the name of a queue or topic space. The classification applies to the destination named by this property, or any destination if you do not specify this property. The service integration buses to which the classification applies depend on your use of the bus property.

discriminator

> The property applies only when the destination property names a topic space. This discriminator value is then an XPath expression that selects one or more topics within the topic space.

description

> Although the description field is optional, you should use it on all the sib_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

RAS attributes

You can specify the following RAS attributes on the sib_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

> Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.
>
> The attribute overrides the control_region_wlm_dispatch_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_iiop_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_iiop_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to `1`, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Each sib_classification_info element can contain one or more of these properties as needed to classify the work for a message. A sib_classification_info element cannot contain more than one instance of each property.

If a message matches several sib_classification_info elements, the element that appears first is used. For example, consider the following specifications:

```
<sib_classification_info bus="MyBus" transaction_class="a" />
<sib_classification_info destination="MyDest" transaction_class="b" />
```

A message that arrives at destination MyDest from the service integration bus MyBus is assigned the classification "a". A message that arrives at MyDest from another bus is assigned the classification "b".

If a message does not match any sib_classification_info element in an enclosing SibClassification element, the message is assigned the default classification from the SibClassification element.

If a message does not match any sib_classification_info element in any SibClassification element, or if no SibClassification elements are defined, all work receives a built-in default classification with the value "SIBUS". You must perform z/OS Workload Manager actions that are required to use the TCLASS value "SIBUS", as described in "Classifying z/OS workload" on page 510.

Mediation classification

The SibClassification element with the attribute type="destinationmediation" defines the section of the document that applies to work for mediations assigned to destinations on a service integration bus. An example of this element follows:

```
<SibClassification type="destinationmediation"
                   schema_version="1.0"
                   default_transaction_class="b">
```

Each SibClassification element can contain one or more sib_classification_info elements. An example of a classification entry element follows:

```
<sib_classification_info
                   transaction_class="e"
                   selector="user.Location=&apos;East&apos;"
                   bus="bigrred"
                   destination="themoon"
                   discriminator="sides/dark"
                   description="n"
                   dispatch_timeout="value1"
                   queue_timeout_percent ="value2"
                   request_timeout="value3"
                   stalled_thread_dump_action="traceback"
                   cputimeused_limit="value5"
                   cputimeused_dump_action="traceback"
                   dpm_interval="value7"
                   dpm_dump_action="traceback"
                   SMF_request_activity_enabled="1"
                   SMF_request_activity_timestamps="1"
                   SMF_request_activity_security="1"
                   SMF_request_activity_CPU_detail="1"
                   classification_only_trace="1"
                   message_tag="value14"/>
```

selector

Use the selector attribute of the sib_classification_info element to assign a transaction class to a mediation that has a selector clause in its deployment descriptor. Use the following rules when defining your sib_classification_info elements:

- The value of the selector attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification, but it can operate on SIMessage messages (more than JMS messages). The syntax can select on system properties (including JMS headers, JMSX properties, and JMS_IBM_properties) and user properties (which must be prefixed by ".user" - for example, for the user property "Location", the selector would specify "user.Location" as shown in the preceding example).

- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < symbol with the < entity reference and the > symbol with the > entity reference. Similarly, if you use an apostrophe or quotation mark, use the ' and " entity references.

bus The name of the service integration bus on which the target destination is assigned. The classification applies to the bus named by this property, or to any bus if you do not specify this property. The destinations to which the classification applies depend on your use of the destination property.

destination

The name of the target bus destination to which the message has been delivered. This is the name of a queue or topic space. The classification applies to the destination named by this

property, or any destination if you do not specify this property. The service integration buses to which the classification applies depend on your use of the bus property.

discriminator

The property applies only when the destination property names a topic space. This discriminator value is then an XPath expression that selects one or more topics within the topic space.

description

Although the description field is optional, you should use it on all the sib_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

RAS attributes

You can specify the following RAS attributes on the sib_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the control_region_wlm_dispatch_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_iiop_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_iiop_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to `1`, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Each sib_classification_info element can contain one or more of these properties as needed to classify the work for a message. A sib_classification_info element cannot contain more than one instance of each property.

If a message matches several sib_classification_info elements, the element that appears first is used. For example, consider the following specifications:

```
<sib_classification_info transaction_class="e" destination="themoon" description="n" />
<sib_classification_info transaction_class="f" description="n" />
```

A message that arrives at the mediated destination themoon is assigned the classification "e". A message that arrives at another mediated destination is assigned the classification "f".

If a message does not match any sib_classification_info element in an enclosing SibClassification element, the message is assigned the default classification from the SibClassification element.

If a message does not match any sib_classification_info element in *any* SibClassification element, or if *no* SibClassification elements are defined, all work receives a built-in default classification with the value "SIBUS". You must perform z/OS Workload Manager actions that are required to use the TCLASS value "SIBUS", as described in "Classifying z/OS workload" on page 510.

WebSphere MQ messaging provider classification

The WMQRAClassification element defines the section of the document that applies to work for message-driven beans (MDBs) deployed against WebSphere MQ messaging provider activation specifications. An example of this element follows:

```
<WMQRAClassification default_transaction_class="TC99" schema_version="1.0">
```

A WMQRAClassification element can contain one or more wmqra_classification_info elements. Two examples of wmqra_classification_info elements follow:

```
<wmqra_classification_info transaction_class="TC_4"
                destination="topic://a/b/*"
                description="Any topics starting with a/b/ map to TC_4"/>

<wmqra_classification_info transaction_class="TC_3"
                selector="JMSPriority&gt;3 AND JMSPriority&lt;8"
                destination="queue://QMGR1/Q1"
                queue_manager="QMGR1"
                description="medium priorities with a queue manager name of QMGR1 and
                        a queue name of Q1 map to TC_3"
                dispatch_timeout="value1"
                queue_timeout_percent ="value2"
                request_timeout="value3"
                stalled_thread_dump_action="traceback"
                cputimeused_limit="value5"
                cputimeused_dump_action="traceback"
                dpm_interval="value7"
                dpm_dump_action="traceback"
                SMF_request_activity_enabled="1"
                SMF_request_activity_timestamps="1"
```

```
SMF_request_activity_security="1"
SMF_request_activity_CPU_detail="1"
classification_only_trace="1"
message_tag="value14"/>
```

selector

Use the selector attribute of the wmqra_classification_info element to assign a transaction class to a message based on its properties. This attribute can also be used to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor:

- The value of the selector attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification.
- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < symbol with the < entity reference and the > symbol with the > entity reference. Similarly, if you use an apostrophe or quotation mark, use the ' and " entity references.

destination

A URI representing the WebSphere MQ destination to which the message has been delivered. The classification applies to the destination named by this property, or to any destination if you do not specify this property. If the URI represents a queue type destination it can optionally include a queue manager name, but that name is ignored and is not be used for classification. If the URI represents a topic type destination, it can make use of wildcards. For more information on wildcard support with WebSphere MQ see the WebSphere MQ information center.

queue_manager

The name of the WebSphere MQ queue manager to which the message has been delivered. The classification applies to the queue manager named by this property, or to any queue manager if you do not specify this property. The queue manager name must conform to WebSphere MQ naming conventions.

Note that this field must not be set to the name of a WebSphere MQ queue sharing group. Instead, you must either create a wmqra_classification_info element for every queue manager in the queue sharing group, or base the classification on something else such as the destination attribute.

description

Although the description field is optional, you must use it on all the wmqra_classification_info elements.

RAS attributes

You can specify the following RAS attributes on the wmqra_classification_info element. Nested elements inherit the RAS attributes of the parent element. Nested elements can override the RAS attributes of a parent element. All the RAS attributes are optional.

dispatch_timeout

Specifies the amount of time that a control region waits after dispatching a request to the WLM queue. If the control region does not receive a response in the specified time, it issues a time out. The request is a request that the classification element has classified.

The attribute overrides the control_region_wlm_dispatch_timeout server property.

queue_timeout_percent

Specifies the WLM queue timeout as a percentage of the dispatch timeout. The range for the percentage is 0 - 99. The queue timeout is the amount of time a request can remain on the WLM queue undispatched before the request times out. The request is a request that the classification element has classified.

The attribute overrides the control_region_iiop_queue_timeout_percent server property.

request_timeout

Specifies the timeout value in seconds applied to outbound requests that originate under dispatched requests. The dispatched request is a request that the classification element has classified.

The attribute overrides the com.ibm.CORBA.RequestTimeout server property.

stalled_thread_dump_action

Specifies the dump action that the server takes when requests exceed their dispatch timeout specified on the dispatch_timeout attribute. The request is a request that the classification element has classified. Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_iiop_stalled_thread_dump_action server property.

cputimeused_limit

Specifies the CPU timeout in milliseconds. The CPU timeout is the amount of CPU time allowed for the request before the request times out. The request is a request that the classification element has classified.

The attribute overrides the server_region_request_cputimeused_limit server property.

cputimeused_dump_action

Specifies the dump action that the server takes when requests exceed their CPU timeout specified on the cputimeused_limit attribute. The request is a request that the classification element has classified.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_cputimeused_dump_action server property.

dpm_interval

Specifies in seconds the Dispatch Progress Monitor (DPM) interval. The DPM monitor triggers the DPM dump action on the interval for requests that the classification element has classified. Specify the DMP dump action on the dpm_dump_action attribute.

The attribute does not override any server properties. You must use the modify command to enable server-wide DPM intervals.

dpm_dump_action

Specifies the DPM dump action that is taken at every DPM interval for requests that the classification element classifies. Specify the DPM interval on the dpm_interval attribute.

Valid values are `svcdump`, `javacore`, `heapdump`, `traceback`, `javatdump`, and `none`.

The attribute overrides the server_region_dpm_dump_action server property.

SMF_request_activity_enabled

Specifies whether System Management Facilities (SMF) 120 subtype 9 records are collected for requests that the classification element classifies.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_enabled server property.

SMF_request_activity_timestamps

Specifies whether to format timestamps in human-readable form on the SMF 120 subtype 9 record for requests. The requests are requests that the classification element has classified.

Valid values are `0` (no) and `1` (yes).

The attribute overrides the server_SMF_request_activity_timestamps server property.

SMF_request_activity_security

Specifies whether the security data section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_security server property.

SMF_request_activity_CPU_detail

Specifies whether the CPU usage breakdown section of the SMF 120 subtype 9 record is collected for requests. The requests are requests that the classification element has classified.

Valid values are 0 (no) and 1 (yes).

The attribute overrides the server_SMF_request_activity_CPU_detail server property.

classification_only_trace

Specifies whether to generate trace records for requests that the classification element classifies.

Valid values are 0 (no) and 1 (yes).

The attribute does not override any server properties. The TRACERECORD modify command overrides the classification_only_trace.

If any classification element has classification_only_trace set to 1, then classification level tracing is in effect for the application server. The only threads that generate trace records are threads that process requests that a classification element classifies. That classification element must define `classification_only_trace="1"`. Any thread not processing such a request has trace collection disabled and does not generate trace records.

If no classification element defines `classification_only_trace="1"`, then classification level tracing is not in effect for the application server. The server does not disable trace collection for any threads. All trace records are written to output as prescribed by the trace specification.

The classification_only_trace attribute does not affect how a trace specification is defined and activated. The attribute only affects which threads write trace records.

message_tag

Specifies a string token that prints with all the trace records, log messages, and system messages generated for requests that the classification element classifies.

The attribute has a maximum of eight characters.

The attribute does not override any server properties.

Each wmqra_classification_info element can contain one or more of these properties as needed to classify the work for a message. A wmqra_classification_info element cannot contain more than one instance of each property.

If a message matches several wmqra_classification_info elements, the element that appears first is used. For example, consider the following specifications:

```
<wmqra_classification_info queue_manager="QMGR1" transaction_class="TC_1" />
<wmqra_classification_info destination="queue:///Q1" transaction_class="TC_2" />
```

A message that arrives at destination Q1 on queue manager QMGR1 is assigned the classification "TC_1". A message that arrives at Q1 from another queue manager is assigned the classification "TC_2".

If a message does not match any wmqra_classification_info element in an enclosing WMQRAClassification element, the message is assigned the default classification from the WMQRAClassification element. If there are multiple WMQRAClassification elements, the default transaction class from the first WMQRAClassification element is used.

If no WMQRAClassification elements are defined, all work receives the default classification "WMQRA". You must perform z/OS Workload Manager actions that are required to use the TCLASS value "WMQRA", as described in "Classifying z/OS workload" on page 510.

Sample z/OS workload classification document without RAS attributes

The sample z/OS workload classification document contains attributes for classifying inbound HTTP, IIOP, Session Initiation Protocol (SIP), and message-driven bean (MDB) work requests for the z/OS workload manager. This sample does not contain RAS attributes.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Classification SYSTEM "Classification.dtd" >
<Classification schema_version="1.0">
<!--
      Internal Classification Rules
-->
   <InboundClassification  type="internal"
                           schema_version="1.0"
                           default_transaction_class="value1"/>

<!--
      IIOP Classification Rules
-->
   <InboundClassification type="iiop"
                          schema_version="1.0"
                          default_transaction_class="A0">
    <iiop_classification_info transaction_class="A1"
                              application_name="IIOPStatelessSampleApp"
                              module_name="StatelessSample.jar"
                              component_name="Sample20"
                              description="Sample20 EJB Classification">
    <iiop_classification_info transaction_class=""
                              method_name="echo"
                              description="No TCLASS for echo()" />
    <iiop_classification_info  transaction_class="A1B"
                              method_name="ping"
                              description="Ping method" />
    </iiop_classification_info>
    <iiop_classification_info application_name="*"
                              module_name="*"
                              component_name="*"
                              transaction_class="A2"
                              description="TCLASS the rest to A2">
    <iiop_classification_info transaction_class="A2A"
                              method_name="resetFilter"
                              description="Sp1 case resetFilter()" />
    </iiop_classification_info>
  </InboundClassification>

<!--
      HTTP Classification Rules
-->
   <InboundClassification type="http"
                          schema_version="1.0"
                          default_transaction_class="M">
    <http_classification_info transaction_class="N"
                              host="yourhost.yourcompany.com"
                              description="Virtual Host yourhost">
    <http_classification_info transaction_class="O"
                              port="9080"
                              description="Def yourhost HTTP reqs">
    <http_classification_info transaction_class="Q"
                              uri="/gcs/admin"
                              description = "Gcs" />
    <http_classification_info  transaction_class="S"
                              uri="/gcs/admin/1*"
                              description="GCS login" />
    <http_classification_info transaction_class="P"
                              port="9081"
                              description=" Def yourhost HTTPS reqs "/>
    <http_classification_info transaction_class=""
                              uri="/gcss/mgr/*"
                              description="GCSS Mgr" />
    </http_classification_info>
    </http_classification_info>
  </InboundClassification>

<!--
      SIP Classification Rules
-->
   <InboundClassification type="sip"
                          schema_version="1.0"
                          default_transaction_class="value1"/>

<!--
```

```
         MDB Classification Rules
-->
   <InboundClassification type="mdb"
                          schema_version="1.0"
                          default_transaction_class="qrs">
     <endpoint type="messagelistenerport"
             name="IVPListenerPort"
             defaultclassification="MDBX"
             description="ABC">
       <classificationentry  selector="Location=&apos;East&apos;"
                             classification="MDB1"
                             description="DEF"/>
       <classificationentry  selector="Location&lt;&gt;&apos;East&apos;"
                             classification="MDB2"
                             description="XYZ" />
     </endpoint>
     <endpoint  type="messagelistenerport"
             name="SimpleMDBListenerPort"
             defaultclassification="MDBX"
             description="GHI" />

   </InboundClassification>

     <SibClassification type="jmsra" schema_version="1.0"
         default_transaction_class="a">
       <sib_classification_info transaction_class="b"
          selector="user.Location=&apos;East&apos;" bus="magic"
          destination="nowhere" description="n" />
       <sib_classification_info transaction_class="c"
          selector="user.Location=&apos;West&apos;" bus="omni"
description="n" />
     </SibClassification>
     <SibClassification type="destinationmediation" schema_version="1.0"
         default_transaction_class="b">
       <sib_classification_info transaction_class="e"
          selector="user.Location=&apos;East&apos;" destination="themoon"
          discriminator="sides/dark" description="n" />
       <sib_classification_info transaction_class="f"
          selector="user.Location=&apos;West&apos;" description="n"
/>
     </SibClassification>

     <WMQRAClassification default_transaction_class="TC99" schema_version="1.0">
       <wmqra_classification_info transaction_class="TC_1"
                             queue_manager="GOLD"
                             description="gold queue manager maps
to TC_1"/>
       <wmqra_classification_info transaction_class="TC_2"
                             selector="JMSPriority&gt;7"
                             description="high priorities maps to
TC_2"/>
       <wmqra_classification_info transaction_class="TC_3"
                             selector="JMSPriority&gt;3 AND
JMSPriority&lt;8"
                             description="medium priorities maps
to TC_3"/>
     </WMQRAClassification>

<!--
       OLA Classification Rules
-->
   <InboundClassification type="ola"
                          schema_version="1.0"
                          default_transaction_class="A0">
     <ola_classification_info transaction_class="FAST1"
                             service_name="ejb/InteractiveTransactionBean"
                             description="EJB classification for quick turnaround"/>
     <ola_classification_info transaction_class="SLOW1"
                             service_name="ejb/BackgroundBean"
                             description="EJB classification for low priority" />
     <ola_classification_info propagate_transaction_name="true"
                             service_name="ejb/CalledFromCICSBean"
                             description="use same service class as client" />
   </InboundClassification>

<!--
   Workload Classification Document for P5SR01x servers
   Change History

   _____
   Activity               Date            Author
   Created                01-28-2005      IPL

-->

</Classification>
```

Sample z/OS workload classification document containing RAS attributes

The sample z/OS workload classification document contains attributes for classifying inbound HTTP, IIOP, and MDB work requests for the z/OS workload manager. This sample contains RAS attributes.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Classification SYSTEM "Classification.dtd" >
<Classification schema_version="1.0">

<!-- IIOP Classification Rules -->
<InboundClassification
  type="iiop"
  schema_version="1.0"
  default_transaction_class="TC">

  <!--
      IIOP classification info for app "EJBApp1".  The classification element defines a
      transaction_class of "TC1".  This value overrides the default_transaction_class of
      the InboundClassification element ("TC").  The classification element also
      specifies a message_tag, which is applied to all requests that are classified
      underneath this classification element.                                          -->
  <iiop_classification_info
    application_name="EJBApp1"
    transaction_class="TC1"
    message_tag="EJBApp1">

    <!--
        EJBApp1 contains a jar module named "MyEJB.jar" with an EJB named "MyEJBClass".
        The transaction_class is not defined for this element; thus the element
        inherits the transaction_class from its parent node, "TC1".  This element also
        inherits the message_tag attribute from its parent node, "EJBApp1".  In general,
        a classification element inherits all the RAS attributes from all its ancestor
        nodes, with nearer ancestor nodes (for example, direct parents) taking precedence over
        ancestor nodes further up the chain (for example, grandparents).          -->
    <iiop_classification_info
      module_name="MyEJB.jar"
      component_name="MyEJBClass">

      <!--
          MyEJBClass contains methods named "helloWorld" and "goodbyeWorld".  helloWorld
          is assigned a dispatch_timeout of 30 seconds and a queue_timeout_percent of 90,
          meaning that the queue timeout value is 90% of the dispatch_timeout value.  The
          classification element also specifies SMF_request_activity_enabled=1, meaning
          SMF 120 subtype 9 records are collected for all requests targeted against
          the helloWorld method.  Also note that this classification element does not
          define a transaction_class; therefore it inherits the transaction_class from
          the nearest ancestor element that defines one.  In this case, the nearest
          ancestor element that defines a transaction_class is the grandparent element,
          "TC1".  Note: if no ancestor elements define a transaction_class, then the
          classification element inherits the default_transaction_class of the
          InboundClassification element.  The default_transaction_class on the
          InboundClassification is required.
          This classification element also inherits the message_tag attribute from its
          grandparent element, "EJBApp1".                                         -->
      <iiop_classification_info
        method_name="helloWorld"
        dispatch_timeout="30"
        queue_timeout_percent="90"
        SMF_request_activity_enabled="1"
      />

      <!--
          The goodbyeWorld method specifies a dispatch_timeout of 60 seconds.  The
          classification element also defines a transaction_class, "TC1gbye", which
          overrides the transaction_class defined by its ancestry.  This element inherits
          the message_tag of its ancestry, "EJBApp1".                             -->
      <iiop_classification_info
        method_name="goodbyeWorld"
        transaction_class="TC1gbye"
        dispatch_timeout="60"
      />

    </iiop_classification_info>
  </iiop_classification_info>

  <!--
      IIOP classification info for app "EJBApp2".  The classification element defines a
      dispatch_timeout of 15 seconds and a message_tag of "EJBApp2".  The
      transaction_class is inherited from the default_transaction_class on the
      InboundClassification, "TC".   All requests that are classified under this
      classification element have a 15 second dispatch timeout, and all trace
      records and log messages generated by those requests are tagged with the
      message_tag attribute value,  "EJBApp2".                                     -->
  <iiop_classification_info
    application_name="EJBApp2"
    dispatch_timeout="15"
    message_tag="EJBApp2">

    <!--
        EJBApp2 contains two jar modules, "MyEJB2a.jar" and "MyEJB2b.jar".  The following
        two classification elements define a transaction_class for each jar module.  No
```

```
            other attributes are defined.  Both elements inherit the attributes of their
            ancestor nodes (dispatch_timeout="15" and message_tag="EJBApp2").             -->
    <iiop_classification_info
      module_name="MyEJB2a.jar"
      transaction_class="TC2a"
    />

    <iiop_classification_info
      module_name="MyEJB2b.jar"
      transaction_class="TC2b"
    />

  </iiop_classification_info>


  <!--
      The following classification element defines attributes for a specific module,
      component, and method of the application "EJBApp3".  The module is
      "MyEJB3.jar", component
      is "MyEJB3Class", and method is "method3".   The transaction_class, dispatch_timeout
      queue_timeout_percent, SMF_request_activity_enabled, and
      SMF_request_activity_timestamps are all defined for this specific method in the
      EJBApp3 application.  No other method on no other EJB within this application are
      assigned these attributes.                                           -->
  <iiop_classification_info
    application_name="EJBApp3"
    module_name="MyEJB3.jar"
    component_name="MyEJB3Class"
    method_name="method3"
    transaction_class="TC3"
    dispatch_timeout="40"
    queue_timeout_percent="90"
    SMF_request_activity_enabled="1"
    SMF_request_activity_timestamps="1"
  />

</InboundClassification>


<!-- Internal Classification Rules -->
<InboundClassification
  type="internal"
  schema_version="1.0"
  default_transaction_class="internal" />

<!-- HTTP Classification Rules -->
<InboundClassification
  type="http"
  schema_version="1.0"
  default_transaction_class="HTC">

  <!--
      The following classification element defines a transaction_class, "HTC8080", for
      all HTTP requests received on host "my.server.com" and port "8080".  The
      classification element also defines RAS attributes dispatch_timeout,
      queue_timeout_percent, timeout_recovery, and stalled_thread_dump_action.  All
      child elements underneath this element inherit these RAS attributes.        -->
  <http_classification_info
    host="my.server.com"
    port="8080"
    transaction_class="HTC8080"
    dispatch_timeout="100"
    queue_timeout_percent="98"
    timeout_recovery="session"
    stalled_thread_dump_action="javacore">

    <!--
        The following classification element applies to all HTTP requests with a URI that
        begins with "/PlantsByWebSphere/".  Every HTTP request received on host
        my.server.com and port 8080 with a URI that begins with /PlantsByWebSphere
        fall under this classification (note: host and port inherited from the parent
        element).  The classification element also defines the message_tag attribute,
        "plantsbw", which are added to every trace record and log message generated
        by any /PlantsByWebSphere/* request.                                 -->
    <http_classification_info
      uri="/PlantsByWebSphere/*"
      message_tag="plantsbw">

      <!--
          The following classification element applies to all HTTP requests with a URI
          that matches "/PlantsByWebSphere/*.jpg (for example, /PlantsByWebSphere/mypic.jpg,
          /PlantsByWebSphere/some/path/anotherpic.jpg).  Again, this filter applies only
          to requests received on host my.server.com and port 8080 (as designated by
          an ancestor node).   The classification element defines a transaction_class,
          "HTCPjpg" and a dispatch_timeout, "10".  It inherits the remaining attributes
          from its ancestor nodes.                                       -->
      <http_classification_info
        uri="*.jpg"
        transaction_class="HTCPjpg"
        dispatch_timeout="10"
```

```
      />

      <!--
          The following classification element applies to all HTTP requests with a URI
          that matches "/PlantsByWebSphere/*.html (for example, /PlantsByWebSphere/index.html,
          /PlantsByWebSphere/some/path/afile.html).                          -->
      <http_classification_info
        uri="*.html"
        transaction_class="HTChtml"
      />
    </http_classification_info>
  </http_classification_info>

  <!--
      The following classification element defines a transaction_class, "HTC80", for
      all HTTP requests received on port "80".  The host attribute is not
      defined;  thus, this element matches any host.                          -->
  <http_classification_info
    port="80"
    transaction_class="HTC80"
    dispatch_timeout="60"
    timeout_recovery="servant"
    message_tag="vanilla"
  />

</InboundClassification>

<!-- MDB Classification Rules -->
<InboundClassification
  type="mdb"
  schema_version="1.0"
  default_transaction_class="mdbdflt">

  <!-- Endpoint for LP 1414, skLP1, for MDB Plan 'A' Test  -->
  <endpoint
    type="messagelistenerport"
    name="skLP1"
    defaultclassification="lp1dft"
    description="Endpoint for LP 1414, skLP1, for MDB Plan 'A' Test">

    <classificationentry
      selector="JMSCorrelationID='TestCase1'"
      classification="lp1s1"
      description="New MDB Sample, TestCase1"
      cputimeused_limit="200101"
      request_timeout="20"
      dispatch_timeout="30"
      dpm_interval="0"
      queue_timeout_percent="20"
      stalled_thread_dump_action="traceback"
      />
  </endpoint>

</InboundClassification>


<!-- SIB Classification Rules -->
<SibClassification type="jmsra" schema_version="1.0" default_transaction_class="Dclass">
      <sib_classification_info transaction_class="Tclass"
                               bus="testbus"
                               destination="themoon"
                               description="test1"
                               dispatch_timeout="30"
                               queue_timeout_percent="20"
                               request_timeout="20"
                               stalled_thread_dump_action="traceback"
                               cputimeused_limit="200101"
                               cputimeused_dump_action="traceback"
                               dpm_interval="0"
                               dpm_dump_action="traceback"
                               classification_only_trace="1"
                               message_tag="sibreqst"
      />
  </SibClassification>

</Classification>
```

DTD

The following DTD defines the elements and attributes used in the preceding sample workload classification documents.

```
<?xml version='1.0' encoding="UTF-8"?>
<!ELEMENT Classification (InboundClassification|SibClassification|WMQRAClassification)+>
<!ATTLIST Classification schema_version CDATA #REQUIRED>
<!ELEMENT InboundClassification ((iiop_classification_info*|http_classification_info*|endpoint*|ola_classification_info*))>
<!ATTLIST InboundClassification type (iiop|mdb|http|internal|sip|ola) #REQUIRED>
```

```
<!ATTLIST InboundClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST InboundClassification schema_version CDATA #REQUIRED>
<!ELEMENT iiop_classification_info (iiop_classification_info*)>
<!-- inputs -->
<!ATTLIST iiop_classification_info activity_workload_classification CDATA #IMPLIED>
<!ATTLIST iiop_classification_info application_name CDATA #IMPLIED>
<!ATTLIST iiop_classification_info component_name CDATA #IMPLIED>
<!ATTLIST iiop_classification_info description CDATA #IMPLIED>
<!ATTLIST iiop_classification_info method_name CDATA #IMPLIED>
<!ATTLIST iiop_classification_info module_name CDATA #IMPLIED>
<!-- outputs -->
<!ATTLIST iiop_classification_info transaction_class CDATA #IMPLIED>
<!ATTLIST iiop_classification_info dispatch_timeout CDATA #IMPLIED>
<!-- control_region_wlm_dispatch_timeout -->
<!ATTLIST iiop_classification_info queue_timeout_percent CDATA #IMPLIED>
<!-- control_region_iiop_queue_timeout_percent -->
<!ATTLIST iiop_classification_info request_timeout CDATA #IMPLIED>
<!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST iiop_classification_info stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
<!-- server_region_iiop_stalled_thread_dump_action -->
<!ATTLIST iiop_classification_info cputimeused_limit CDATA #IMPLIED>
      <!-- server_region_request_cputimeused_limit -->
<!ATTLIST iiop_classification_info cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_cputimeused_dump_action -->
<!ATTLIST iiop_classification_info dpm_interval CDATA #IMPLIED>
          <!-- MODIFY [JOBNAME],DPM,IIOP= -->
<!ATTLIST iiop_classification_info dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_dpm_dump_action -->
<!ATTLIST iiop_classification_info SMF_request_activity_enabled (0|1) #IMPLIED>
     <!-- server_SMF_request_activity_enabled -->
<!ATTLIST iiop_classification_info SMF_request_activity_timestamps (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST iiop_classification_info SMF_request_activity_security (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_security -->
<!ATTLIST iiop_classification_info SMF_request_activity_CPU_detail (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST iiop_classification_info classification_only_trace (0|1) #IMPLIED>
<!ATTLIST iiop_classification_info message_tag CDATA #IMPLIED>

<!ELEMENT endpoint (classificationentry*)>
<!ATTLIST endpoint defaultclassification CDATA #REQUIRED>
<!ATTLIST endpoint name CDATA #REQUIRED>
<!ATTLIST endpoint type (messagelistenerport) #REQUIRED>
<!ATTLIST endpoint description CDATA #IMPLIED>
<!ELEMENT classificationentry EMPTY>
<!-- inputs -->
<!ATTLIST classificationentry selector CDATA #REQUIRED>
<!ATTLIST classificationentry description CDATA #IMPLIED>
<!-- outputs -->
<!ATTLIST classificationentry classification CDATA #REQUIRED>
<!ATTLIST classificationentry dispatch_timeout CDATA #IMPLIED>
        <!-- control_region_mdb_request_timeout -->
<!ATTLIST classificationentry queue_timeout_percent CDATA #IMPLIED>
  <!-- control_region_mdb_queue_timeout_percent -->
<!ATTLIST classificationentry request_timeout CDATA #IMPLIED>
         <!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST classificationentry stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
 <!-- server_region_mdb_stalled_thread_dump_action -->
<!ATTLIST classificationentry cputimeused_limit CDATA #IMPLIED>
        <!-- server_region_request_cputimeused_limit -->
<!ATTLIST classificationentry cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_cputimeused_dump_action -->
<!ATTLIST classificationentry dpm_interval CDATA #IMPLIED>
          <!-- MODIFY [JOBNAME],DPM,IIOP= -->
<!ATTLIST classificationentry dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_dpm_dump_action -->
<!ATTLIST classificationentry SMF_request_activity_enabled (0|1) #IMPLIED>
     <!-- server_SMF_request_activity_enabled -->
<!ATTLIST classificationentry SMF_request_activity_timestamps (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST classificationentry SMF_request_activity_security (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_security -->
<!ATTLIST classificationentry SMF_request_activity_CPU_detail (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST classificationentry classification_only_trace (0|1) #IMPLIED>
<!ATTLIST classificationentry message_tag CDATA #IMPLIED>
<!ELEMENT http_classification_info (http_classification_info*)>
<!-- inputs -->
<!ATTLIST http_classification_info host CDATA #IMPLIED>
<!ATTLIST http_classification_info port CDATA #IMPLIED>
<!ATTLIST http_classification_info uri CDATA #IMPLIED>
<!ATTLIST http_classification_info description CDATA #IMPLIED>
<!ATTLIST http_classification_info transaction_class CDATA #IMPLIED>
<!-- outputs -->
<!ATTLIST http_classification_info dispatch_timeout CDATA #IMPLIED>
        <!-- protocol_http_timeout_output -->
<!ATTLIST http_classification_info queue_timeout_percent CDATA #IMPLIED>
  <!-- control_region_http_queue_timeout_percent -->
<!ATTLIST http_classification_info request_timeout CDATA #IMPLIED>
         <!-- com.ibm.CORBA.RequestTimeout -->
```

```
<!ATTLIST http_classification_info stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_http_stalled_thread_dump_action -->
<!ATTLIST http_classification_info cputimeused_limit CDATA #IMPLIED>
      <!-- server_region_request_cputimeused_limit -->
<!ATTLIST http_classification_info cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_cputimeused_dump_action -->
<!ATTLIST http_classification_info dpm_interval CDATA #IMPLIED>
        <!-- MODIFY [JOBNAME],DPM,HTTP= -->
<!ATTLIST http_classification_info dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_dpm_dump_action -->
<!ATTLIST http_classification_info SMF_request_activity_enabled (0|1) #IMPLIED>
     <!-- server_SMF_request_activity_enabled -->
<!ATTLIST http_classification_info SMF_request_activity_timestamps (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST http_classification_info SMF_request_activity_security (0|1) #IMPLIED>
   <!-- server_SMF_request_activity_security -->
<!ATTLIST http_classification_info SMF_request_activity_CPU_detail (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST http_classification_info classification_only_trace (0|1) #IMPLIED>
<!ATTLIST http_classification_info message_tag CDATA #IMPLIED>
<!ATTLIST http_classification_info timeout_recovery (servant|session) #IMPLIED>
     <!-- protocol_http_timeout_output_recovery -->

<!ELEMENT ola_classification_info (ola_classification_info*)>
<!ATTLIST ola_classification_info transaction_class CDATA #IMPLIED>
<!ATTLIST ola_classification_info propagate_transaction_name (true|false) #IMPLIED>
<!ATTLIST ola_classification_info service_name CDATA #IMPLIED>
<!ATTLIST ola_classification_info description CDATA #IMPLIED>
<!ATTLIST ola_classification_info dispatch_timeout CDATA #IMPLIED>
      <!-- control_region_wlm_dispatch_timeout -->
<!ATTLIST ola_classification_info queue_timeout_percent CDATA #IMPLIED>
  <!-- control_region_iiop_queue_timeout_percent -->
<!ATTLIST ola_classification_info request_timeout CDATA #IMPLIED>
       <!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST ola_classification_info stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
 <!-- server_region_iiop_stalled_thread_dump_action -->
<!ATTLIST ola_classification_info cputimeused_limit CDATA #IMPLIED>
      <!-- server_region_request_cputimeused_limit -->
<!ATTLIST ola_classification_info cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_cputimeused_dump_action -->
<!ATTLIST ola_classification_info dpm_interval CDATA #IMPLIED>
        <!-- MODIFY [JOBNAME],DPM,IIOP= -->
<!ATTLIST ola_classification_info dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_dpm_dump_action -->
<!ATTLIST ola_classification_info SMF_request_activity_enabled (0|1) #IMPLIED>
     <!-- server_SMF_request_activity_enabled -->
<!ATTLIST ola_classification_info SMF_request_activity_timestamps (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST ola_classification_info SMF_request_activity_security (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_security -->
<!ATTLIST ola_classification_info SMF_request_activity_CPU_detail (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST ola_classification_info classification_only_trace (0|1) #IMPLIED>
<!ATTLIST ola_classification_info message_tag CDATA #IMPLIED>

<!ELEMENT SibClassification (sib_classification_info+)>
<!ATTLIST SibClassification type (jmsra|destinationmediation) #REQUIRED>
<!ATTLIST SibClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST SibClassification schema_version CDATA #REQUIRED>
<!ELEMENT sib_classification_info EMPTY>
<!-- inputs -->
<!ATTLIST sib_classification_info selector CDATA #IMPLIED>
<!ATTLIST sib_classification_info bus CDATA #IMPLIED>
<!ATTLIST sib_classification_info destination CDATA #IMPLIED>
<!ATTLIST sib_classification_info discriminator CDATA #IMPLIED>
<!ATTLIST sib_classification_info description CDATA #IMPLIED>
<!-- outputs -->
<!ATTLIST sib_classification_info transaction_class CDATA #REQUIRED>
<!ATTLIST sib_classification_info dispatch_timeout CDATA #IMPLIED>
        <!-- control_region_wlm_dispatch_timeout -->
<!ATTLIST sib_classification_info queue_timeout_percent CDATA #IMPLIED>
  <!-- control_region_iiop_queue_timeout_percent -->
<!ATTLIST sib_classification_info request_timeout CDATA #IMPLIED>
         <!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST sib_classification_info stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
 <!-- server_region_iiop_stalled_thread_dump_action -->
<!ATTLIST sib_classification_info cputimeused_limit CDATA #IMPLIED>
      <!-- server_region_request_cputimeused_limit -->
<!ATTLIST sib_classification_info cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_cputimeused_dump_action -->
<!ATTLIST sib_classification_info dpm_interval CDATA #IMPLIED>
        <!-- MODIFY [JOBNAME],DPM,IIOP= -->
<!ATTLIST sib_classification_info dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_dpm_dump_action -->
<!ATTLIST sib_classification_info SMF_request_activity_enabled (0|1) #IMPLIED>
     <!-- server_SMF_request_activity_enabled -->
<!ATTLIST sib_classification_info SMF_request_activity_timestamps (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST sib_classification_info SMF_request_activity_security (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_security -->
```

```
<!ATTLIST sib_classification_info SMF_request_activity_CPU_detail (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST sib_classification_info classification_only_trace (0|1) #IMPLIED>
<!ATTLIST sib_classification_info message_tag CDATA #IMPLIED>

<!ELEMENT WMQRAClassification (wmqra_classification_info+)>
<!ATTLIST WMQRAClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST WMQRAClassification schema_version CDATA #REQUIRED>
<!ELEMENT wmqra_classification_info EMPTY>
<!-- inputs -->
<!ATTLIST wmqra_classification_info selector CDATA #IMPLIED>
<!ATTLIST wmqra_classification_info queue_manager CDATA #IMPLIED>
<!ATTLIST wmqra_classification_info destination CDATA #IMPLIED>
<!ATTLIST wmqra_classification_info description CDATA #IMPLIED>
<!-- outputs -->
<!ATTLIST wmqra_classification_info transaction_class CDATA #REQUIRED>
<!ATTLIST wmqra_classification_info dispatch_timeout CDATA #IMPLIED>
        <!-- control_region_wlm_dispatch_timeout -->
<!ATTLIST wmqra_classification_info queue_timeout_percent CDATA #IMPLIED>
  <!-- control_region_iiop_queue_timeout_percent -->
<!ATTLIST wmqra_classification_info request_timeout CDATA #IMPLIED>
        <!-- com.ibm.CORBA.RequestTimeout -->
<!ATTLIST wmqra_classification_info stalled_thread_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
 <!-- server_region_iiop_stalled_thread_dump_action -->
<!ATTLIST wmqra_classification_info cputimeused_limit CDATA #IMPLIED>
      <!-- server_region_request_cputimeused_limit -->
<!ATTLIST wmqra_classification_info cputimeused_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_cputimeused_dump_action -->
<!ATTLIST wmqra_classification_info dpm_interval CDATA #IMPLIED>
          <!-- MODIFY [JOBNAME],DPM,IIOP= -->
<!ATTLIST wmqra_classification_info dpm_dump_action (none|svcdump|javacore|heapdump|traceback|javatdump) #IMPLIED>
  <!-- server_region_dpm_dump_action -->
<!ATTLIST wmqra_classification_info SMF_request_activity_enabled (0|1) #IMPLIED>
     <!-- server_SMF_request_activity_enabled -->
<!ATTLIST wmqra_classification_info SMF_request_activity_timestamps (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_timestamps -->
<!ATTLIST wmqra_classification_info SMF_request_activity_security (0|1) #IMPLIED>
    <!-- server_SMF_request_activity_security -->
<!ATTLIST wmqra_classification_info SMF_request_activity_CPU_detail (0|1) #IMPLIED>
  <!-- server_SMF_request_activity_CPU_detail -->
<!ATTLIST wmqra_classification_info classification_only_trace (0|1) #IMPLIED>
<!ATTLIST wmqra_classification_info message_tag CDATA #IMPLIED>
```

Example classification of an HTTP inbound request

Use the following http request with the previously described sample z/OS workload classification document containing RAS attributes:

http://my.server.com:8080/PlantsByWebSphere/plants/newOrder.html

Because this request is an HTTP request, the application server scans only the http_classification_info elements. The application server scans the elements in the order that they occur in the workload classification file.

The application server finds the following http_classification_info element first in the sample z/OS workload classification document that contains RAS attributes:

```
 <http_classification_info
   host="my.server.com"
   port="8080"
   transaction_class="HTC8080"
   dispatch_timeout="100"
   queue_timeout_percent="98"
   timeout_recovery="session"
   stalled_thread_dump_action="javacore">
```

This element has input attributes and values of `host="my.server.com"` and `port="8080"`. Since these attributes match the inbound HTTP request, the application server descends into this element and compares the child nodes. The application server finds the following child element:

```
 <http_classification_info
   uri="/PlantsByWebSphere/*"
   message_tag="plantsbw">
```

The child element defines the input attribute and value of `uri="/PlantsByWebSphere/*"`, which matches the URI of the inbound request URI. The application server then scans the children of the element. The application server finds the first child element:

```
<http_classification_info
    uri="*.jpg"
    transaction_class="HTCPjpg"
    dispatch_timeout="10"
/>
```

This child element contains the `uri="*.jpg"` attribute and value. Since this attribute-value pair does not match the input URI, the application server moves to the next child element. The application server finds the second child element:

```
<http_classification_info
        uri="*.html"
        transaction_class="HTChtml"
    />
```

This child element contains the `uri="*.html"` attribute and value, which matches the input URI.

Since no further child elements exist, the classification element classifies the request. The application server assigns the request all the output attributes from this classification element and all its ancestor elements. The following attribute-value pairs are a complete list of output attribute-value pairs that the application server assigns to the request:

```
dispatch_timeout="100"
  queue_timeout_percent="98"
  timeout_recovery="session"
  stalled_thread_dump_action="javacore"
  message_tag="plantsbw"
  transaction_class="HTChtml"
```

The application server reads any RAS attributes not defined in the workload classification file from the server-wide configuration and assigns them to the request. The relevant server-wide configuration properties, including which ones the classification data overrode, are in the following list:

```
protocol_http_timeout_output                  -- overridden by dispatch_timeout
control_region_http_queue_timeout_percent     -- overridden by queue_timeout_percent
com.ibm.CORBA.RequestTimeout
server_region_http_stalled_thread_dump_action -- overridden by stalled_thread_dump_action
server_region_request_cputimeused_limit
server_region_cputimeused_dump_action
server_region_dpm_dump_action
server_SMF_request_activity_enabled
server_SMF_request_activity_timestamps
server_SMF_request_activity_CPU_detail
protocol_http_timeout_output_recovery         -- overridden by timeout_recovery
```

For any server-wide configuration properties that the classification data does not override, the request inherits the server-wide property value.

Example classification of an IIOP inbound request

Use the following IIOP request with the previously described sample z/OS workload classification document containing RAS attributes:

IIOP inbound request for MyEJB2bBean.someMethod() in module MyEJB2b.jar from application EJBApp2.

Because the inbound request is an IIOP request, the application server scans only the iiop_classification_info elements. The application server finds the following iiop_classification_info element first in the sample z/OS workload classification document that contains RAS attributes:

```
<iiop_classification_info
    application_name="EJBApp1"
    transaction_class="TC1"
    message_tag="EJBApp1">
```

This element has an input attribute and value of `application_name="EJBApp1"`. The application name of EJBApp1 on the application_name attribute does not match the application name of EJBApp2 from the inbound request. Therefore, the application server skips this classification element and all of its child elements, and moves to the next element. The application server finds the next element:

```
<iiop_classification_info
    application_name="EJBApp2"
    dispatch_timeout="15"
    message_tag="EJBApp2">
```

This element has an input attribute and value of `application_name="EJBApp2"`. Because this attribute-value pair matches the application name of the inbound request, the scanner descends into this element. The application server finds the first child element:

```
<iiop_classification_info
    module_name="MyEJB2a.jar"
    transaction_class="TC2a"
/>
```

This element contains the input attribute and value of `module_name="MyEJB2a.jar"`. Because this attribute-value pair does not match the MyEJB2b.jar module name of the inbound request, the application server scans the next child element:

```
<iiop_classification_info
    module_name="MyEJB2b.jar"
    transaction_class="TC2b"
/>
```

This element contains the attribute and value of `module_name="MyEJB2b.jar"`. The module name of MyEJB2b.jar on the module_name attribute matches the module name of the inbound request. Since for no further child elements exist, the application server classifies the request to this element. The request inherits all the output attributes from this element and all the ancestor elements. The following attribute-value pairs are a complete list of output attribute-value pairs that the application server assigns to the request:

```
dispatch_timeout="15"
message_tag="EJBApp2"
transaction_class="TC2b"
```

The application server reads any RAS attributes not defined in the workload classification file from the server-wide configuration and assigns them to the request. The relevant server-wide configuration properties, including which ones the classification data overrode, are in the following list:

```
control_region_wlm_dispatch_timeout              -- overridden by dispatch_timeout
control_region_iiop_queue_timeout_percent
com.ibm.CORBA.RequestTimeout
server_region_iiop_stalled_thread_dump_action
server_region_request_cputimeused_limit
server_region_cputimeused_dump_action
server_region_dpm_dump_action
server_SMF_request_activity_enabled
server_SMF_request_activity_timestamps
server_SMF_request_activity_CPU_detail
```

For any server-wide configuration properties that the classification data does not override, the request inherits the server-wide property value.

Appendix. Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories imply specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations - z/OS

app_server_root

Refers to the top directory for a WebSphere Application Server node.

The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are *was.install.root* and *WAS_HOME*.

The default varies based on node type. Common defaults are `configuration_root`/AppServer and `configuration_root`/DeploymentManager.

configuration_root

Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.

The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.

The default is `/wasv8config/`*cell_name*`/`*node_name*.

plug-ins_root

Refers to the installation root directory for Web Server Plug-ins.

profile_root

Refers to the home directory for a particular instantiated WebSphere Application Server profile.

Corresponding product variables are *server.root* and *user.install.root.*

In general, this is the same as *app_server_root*`/profiles/`*profile_name*. On z/OS, this will always be *app_server_root*`/profiles/default` because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root

Refers to the root directory for product code installed with SMP/E or IBM Installation Manager.

The corresponding product variable is *smpe.install.root*.

The default is `/usr/lpp/zWebSphere/V8R0`.

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

APACHE INFORMATION. This information may include all or portions of information which IBM obtained under the terms and conditions of the Apache License Version 2.0, January 2004. The information may also consist of voluntary contributions made by many individuals to the Apache Software Foundation. For more information on the Apache Software Foundation, please see http://www.apache.org. You may obtain a copy of the Apache License at http://www.apache.org/licenses/LICENSE-2.0.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

> IBM Director of Intellectual Property & Licensing
> IBM Corporation
> North Castle Drive
> Armonk, NY 10504-1785
> USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

> IBM Corporation
> Mail Station P300
> 2455 South Road
> Poughkeepsie, NY 12601-5400
> USA
> Attention: Information Requests

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

645

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol ($^®$ or $^{™}$), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. For a current list of IBM trademarks, visit the IBM Copyright and trademark information Web site (www.ibm.com/legal/copytrade.shtml).

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.

Index