WebSphere® Application Server for z/OS, Version 6.0.1

IBM

Setting up the application serving environment

GA22-7958-03

Note
Before using this information, be sure to read the general information under "Notices" on page 439.

Compilation date: March 14, 2005

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 919-254-0206.

  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.

Chapter 1. Who should use this book

This PDF is for the administrator who is responsible for integrating application serving capabilities into an existing network environment. It looks at the product as part of a larger system, typically a production environment or realistic test environment. This section reiterates some installation and customization activities, including topology planning and creating product configurations. It carries the focus into the administrative realm, discussing port configuration and other network concerns. See also the *Installing your application serving environment* PDF. This information expands the topology planning discussion by describing how to set up and maintain logical administrative domains of cells and nodes, and how to balance workload through clustering and high availability configurations.

Chapter 2. Overview and new features for setting up the application serving environment

Introduction: System administration

This topic describes the administration of WebSphere Application Server, Version 6.0.1 products and the applications that run on them.

Presentations from IBM Education Assistant

The following presentations provide a quick overview:

- System management architecture
- Administrative security
- Administrative clients overview
  - Start, stop, and monitor processes
  - Other commands
  - Browser-based administrative console
  - Scripting - wsadmin
- Topologies and logical administrative domains
  - Resource scoping
  - Cells, deployment managers, and node agents
  - Build cells - Add and remove nodes
  - Manage node groups
- Applications and application resources
  - Application management overview
  - JDBC
  - Installing and uninstalling applications
  - Managed application resources - Enhanced EAR files
  - Fine grained application updates
- Servers
  - Manage Web server nodes
- Configuration management
  - Configuration repository
  - Configuration archives
  - File synchronization

Getting started with WebSphere Application Server

Installing

See the *Installing your application serving environment* PDF for a description of installing the WebSphere Application Server product and other installable components.

Configuring

See "Configuring the product after installation" on page 29 for a description of what to do after installing the product.

Migrating

See the *Migrating, coexisting, and interoperating* PDF for a description of how to migrate applications and configuration data from a previous version of WebSphere Application Server.

Using the Samples Gallery

See "Accessing the Samples (Samples Gallery)" in the product Information Center for a description of the set of Samples that ship with each product. The Samples demonstrate common Web application tasks.

Deploying applications

The information center describes how to deploy Web components, such as servlets and JSP files.

Security considerations for WebSphere Application Server for z/OS

Functions supported on WebSphere Application Server for z/OS

WebSphere Application Server for z/OS supports the following functions.

Table 1. Functions supported on WebSphere Application Server for z/OS

Function	Additional information
RunAs EJB	For more information, see the *Securing applications and their environment* PDF.
RunAs for Servlets	For more information, see the *Securing applications and their environment* PDF.
SAF-based IIOP Protocols	For more information, see the *Securing applications and their environment* PDF.
z/OS connector facilities	For more information, see the *Troubleshooting and support* PDF.
Global security enable or disable	For more information, see the *Securing applications and their environment* PDF.
SAF keyrings	For more information, see the *Securing applications and their environment* PDF.
Authentication functions	*Authentication function examples*: Basic, SSL digital certificates, form-based login, security constraints, trust association interceptor
J2EE security resources	For more information, see the *Securing applications and their environment* PDF.
Web authentication (LTPA)	For more information, see the *Securing applications and their environment* PDF.
IIOP using LTPA	For more information, see the *Securing applications and their environment* PDF.
WebSphere application bindings	WebSphere application bindings can be used to provide user to role mappings.
Synch to OS Thread	For more information, see the *Securing applications and their environment* PDF.
J2EE role-based naming security	For more information, see the *Securing applications and their environment* PDF.

Table 1. Functions supported on WebSphere Application Server for z/OS (continued)

Function	Additional information
J2EE role-based administrative security	For more information, see the *Securing applications and their environment* PDF.
SAF registries	For more information, see the *Securing applications and their environment* PDF.
Identity assertion	For more information, see the *Securing applications and their environment* PDF.
Authentication protocols	*Example*: z/SAS, CSIV2 For more information, see For more information, see the *Securing applications and their environment* PDF.
CSIv2 conformance level ″0″	For more information, see For more information, see the *Securing applications and their environment* PDF.
J2EE 1.4 compliance	For more information, see For more information, see the *Securing applications and their environment* PDF.
JAAS programming model WebSphere extensions	For more information, see For more information, see the *Securing applications and their environment* PDF.

All basic WebSphere Application Server provide the following functions:

- **Using RunAs:** Use RunAs to change the identity of a caller, server, or role. This designation is now part of the servlet specification.
- **Support of SAF-based IIOP authentication protocols:** Network Deployment uses Secure Authentication Services (SAS) for IIOP authentication. z/OS has its own version of SAS called z/OS Secure Authentication Services (z/SAS) (with similar functions but different mechanisms), and it handles functions such as local security, Secure Sockets Layer (SSL)-based authorization, digital certificates with System Authorization Facility (SAF) mapping, and SAF identity assertion.
- **SAF-based authorization and RunAs capability:** This allows you to use SAF (EJBROLE) profiles for permission and delegation security information.
- **Support for z/OS connector facilities:** Instead of using an alias where a user ID and password is stored, the ability to propagate local OS identities is supported.
- **SAF keyring support for HTTP and IIOP:** Use SystemSSL for HTTP, IIOP, and SAF key ring support. You can also use JSSE.
- **Authentication functions:** Web Authentication mechanisms such as basic, SSL digital certificates, form-based login, security constraints, and trust association interceptor offer the same functionality in Version 6.0.x as offered in Version 5.
- **Authorization for J2EE resources:** Authorization for J2EE resources employs roles similar to the ones used in Version 4, and these roles are used as descriptors.
- **Security enablement:** Security can be enabled or disabled globally. When the server comes up there is some level of security on, but security is disabled until the administrator sets it up.
- **Web authentication using LTPA and SWAM:** Single-signon using Lightweight Third Party Authentication (LTPA) or Simple WebSphere Authentication Mechanism (SWAM) is supported.
- **IIOP authentication using LTPA:** IIOP authentication using LTPA is supported.
- **WebSphere Application Bindings for Authorization:** WebSphere Application Bindings for Authorization are now supported.
- **Synch to OS Thread:** Application Synch to OS Thread is supported.

- **J2EE role-based naming security:** J2EE roles are used to protect access to the namespace. The new roles and tasks are cosNamingRead, cosNamingWrite, cosNamingCreate, and cosNamingDelete.
- **Role-based administrative security:** The roles delimiting security are:
  - Monitor (least authorization and is read-only)
  - Operator (can do runtime changes)
  - Configurator (can monitor and configuration privileges)
  - Administrator (most authorization)

Comparing WebSphere Application Server for z/OS with other WebSphere Application Server platforms

A key similarity:
- **Pluggable security model:** The pluggable security model can be authenticated in IIOP (CSIv2), Web Trust Authentication, Java Management Extensions (JMX) Connectors, or the Java Authentication and Authorization Service (JAAS) programming model. You must:
  1. Determine which registry is appropriate and what authentication (token) mechanisms are needed
  2. Determine whether or not the registry is local or remote, and what Web authorizations should be used - Web authorizations include Simple WebSphere authentication mechanism (SWAM) and Lightweight Third-Party Authentication (LTPA)

Key differences include:
- **SAF registries:** Local operating system registries provide premium functionality on z/OS because z/OS spans a sysplex rather than a single server. z/OS provides certificate to user mapping, authorization, and delegation functions.
- **Identity assertion:** Use trusted servers or CBIND to get the authorization required for the server doing the assertion. Distributed platform requires a server to be placed in the trusted server list. z/OS requires a server ID to have a specific CBIND authorization. The Assertion types are SAF user ID, Distinguished Name (DN), and SSL client certificate.
- **zSAS and SAS authentication protocols for IIOP clients:** z/SAS differs from SAS because it supports RACF PassTickets. The SAS layer in WebSphere Distributed uses CORBA portable interceptors to implement their Secure Association Service, and z/OS does not.
- **CORBA features:** z/OS does not support CORBA security interfaces including the CORBA current, LoginHelper, Credentials, and ServerSideAuthenticator models. CORBA functions have been migrated to JAAS.
- **Authentication protocols:** CSIv2 is an Object Management Group (OMG) specification for the z/OS Security Server and is automatically enabled when WebSphere security is enabled. This is a three-layered approach involving a transport layer (SSL/TLS) for message protection, supplemental client authentication layer for user ID and password (GSSUP), and security attribute layer used by middle servers (who must be specially authorized to the target server) for identity assertion.

J2EE 1.3 compliance

Being J2EE-compliant involves:
- **CSIv2 conformance level ″0″:** This is an OMG (related to the z/OS Security Server) specification, which is part of what used to be the CORBA support. CSIv2 is automatically enabled when security is enabled.
- **Use of Java 2 security:** There is ″security-enabled″ and ″Java 2 security-enabled″, and the default for Java2 is ″on″. This provides a fine-grained access control that

is code-based as opposed to subject-based authorization. Each class belongs to one particular domain. Permissions protected by Java 2 security include file access, network access, sockets, exiting Java virtual machine (JVM), administration of properties, and threads. The ″security manager″ is what Java 2 uses as a mechanism for managing security and enforcing the required protections. Extensions to Java 2 security include use of dynamic policy (permissions resource type-based rather than code-based), use of specific default permissions defined for resources in template profiles, and use of filter files to disable policy.

- **Use of JAAS programming:** JAAS programming includes a standard set of APIs for authentication. JAAS is the strategic authorization and authentication mechanism. IBM Developer Kit for Java Technology Edition Version 1.4.2 WebSphere Application Server shipped with WebSphere Version 6.0.x (but some extensions are supplied).
- **Use of the servlet RunAs function:** WebSphere Application Server on the distributed platforms (not the z/OS platform) refers to this function as ″Delegation Policy″. You can change identity to run as a system, caller, or role (user). This function is now part of the servlet specification. Authentication involves using a user ID and password and then mapping the alias to the appropriate XML file to find the user ID of the RunAs role.

Compliance with WebSphere Network Deployment at the API/SPI level

Compliance with WebSphere Network Deployment at the API/SPI level makes deploying applications from Network Deployment on z/OS easier. Features enhanced or deprecated by Network Deployment are enhanced or deprecated by z/OS. However, this does not mean there is no migration for z/OS customers. Compliance with WebSphere Network Deployment at the API/SPI level includes:

- **WebSphere Application Server extensions to the JAAS programming model:** The authorization model is an extension of the Java 2 security model for JAAS programming (so it works with the J2EE model). Subject-based authorization is performed on authenticated user IDs. Instead of merely logging in with a user ID and password, there is now a login process that includes creating a login context, passing callback handlers that prompt for user ID and password, and logging in. WebSphere Application Server for z/OS supplies the login module, the callback handler to retrieve the necessary data, the callbacks, the WSSubject choice, getCallerSubject, and getRunAsSubject .
- **Use of the WebSphere Application Server security APIs:** z/OS supports WebSphere Application Server security APIs.
- **Use of secure JMX connectors:** JMX connectors can be used with user ID and password credentials. The two connector types are RMI and SOAP/HTTPS (and are for administration). The SOAP connector uses the JSSE SSL repertoires. The RMI connector is subject to the same advantages and restrictions as IIOP mechanisms (such as CSIv2).

Introduction: System administration

A variety of tools are provided for administering the WebSphere Application Server product:

- **Console**

  The administrative console is a graphical interface that provides many features to guide you through deployment and systems administration tasks. Use it to explore available management options.

  For more information, refer to “Introduction: Administrative console” on page 8.
-

Administrative agents

Servers, nodes and node agents, cells and the deployment manager are fundamental concepts in the administrative universe of the product. It is also important to understand the various processes in the administrative topology and the operating environment in which they apply.

For more information, refer to "Welcome to basic administrative architecture" on page 14.

- **Scripting**

  The WebSphere administrative (wsadmin) scripting program is a powerful, non-graphical command interpreter environment enabling you to run administrative operations in a scripting language. You can also submit scripting language programs to run. The wsadmin tool is intended for production environments and unattended operations.

  For more information, refer to the *Administering applications and their environment* PDF
- **Commands**

  Command-line tools are simple programs that you run from an operating system command-line prompt to perform specific tasks, as opposed to general purpose administration. Using the tools, you can start and stop application servers, check server status, add or remove nodes, and complete similar tasks.

  For more information, refer to "Introduction: Administrative commands" on page 13.
- **Programming**

  The product supports a Java programming interface for developing administrative programs. All of the administrative tools supplied with the product are written according to the API, which is based on the industry standard Java Management Extensions (JMX) specification.

  For more information, refer to "Introduction: Administrative programs" on page 13.
- **Data**

  Product configuration data resides in XML files that are manipulated by the previously-mentioned administrative tools.

  For more information, refer to "Introduction: Administrative configuration data" on page 13.

Introduction: Administrative console

The administrative console is a graphical interface for performing deployment and system administration tasks. It runs in your Web browser. Your actions in the console modify a set of XML configuration files.

You can use the console to perform tasks such as:

- Add, delete, start, and stop application servers
- Deploy new applications to a server
- Start and stop existing applications, and modify certain configurations
- Add and delete Java 2 Platform, Enterprise Edition (J2EE) resource providers for applications that require data access, mail, URLs, and so on
- Manage variables, shared libraries, and other configurations that can span multiple application servers
- Configure product security, including access to the administrative console
- Collect data for performance and troubleshooting purposes

- Find the product version information. It is located on the front page of the console.

The *Administering applications and their environment* PDF describes how to start and log off the administrative console. These actions will help you to begin using the console so that you can explore the available options. See also the **Reference > Administrator > Settings** section of the product Information Center navigation. It lists the settings or properties you can configure.

Use both the MVS console and the Application Server administrative console to administer the Application Server. For example:

- Use MVS commands that are issued from the MVS console to start the base application server controller region, and the node agent and deployment manager.
- In an application server configuration, you must start the first server with an MVS operator command. After the first server is started, you can use the administrative console, if it has this application, to start other application servers in the node. After the deployment manager and node agent are active (in an ND configuration), you can use the administrative console to start and stop application servers.
- Workload management starts all servant regions using Address Space Create (ASCRE) with the administrative console, you can display and modify Application Server applications and the environments in which they run.

Identifying where to perform WebSphere Application Server operations

Administering WebSphere Application Server involves the use of both the MVS console and the WebSphere Application Server administrative console. For example:

- Use MVS commands issued from the MVS console to start the base Application Server control region, the network deployment node agent, and the deployment manager.
- In a base Application Server configuration, you must start the first server with an MVS operator command. Once the first server starts, you can then use the administrative console, if it has this application, to start other Application Servers in the node. Once the deployment manager and node agent are active (in a network deployment configuration), you can use the administrative console to start and stop application servers.
- Workload management starts all servant regions using Address Space Create (ASCRE).

The following table lists the main Application Server operations tasks and directs you to information that helps you to perform these tasks. The Application Server activities and operations can be performed from:

- A z/OS or OS/390 MVS console (most operations)
- The Application Server administrative console (some operations)
- TSO or resource recovery services (RRS) panels (some operations).

Table 2. Application Server operations tasks

Task	MVS console	Application Server administrative console	TSO panel	Reference to associated procedure
Start operations				
Starting the Application Server environment and location service daemon	Yes	No	No	See Starting servers.
Starting a cluster or application server	Yes	Application server only	No	See Starting clusters, Starting servers, and the *Administering applications and their environment* PDF.
Stop operations				
Stopping the location service daemon	Yes	No	No	See Steps for stopping or canceling the location service daemon from the MVS console .
Stopping a cluster	Yes	Yes	No	See Stopping clusters.
Stopping an application server	Yes	Yes	No	See Stopping servers.
Cancel operations				
Canceling the location service daemon	Yes	No	No	See Steps for stopping or canceling the location service daemon from the MVS console.
Canceling a cluster	Yes	Yes	No	See Stopping clusters.
Canceling an application server	Yes	Yes	No	See Stopping servers.
Display operations				
Displaying the status of ARM-registered address spaces including clusters and servants	Yes	No	No	See Displaying the status of ARM-registered address spaces including WebSphere Application Server for z/OS and server instances.
Displaying units of work (threads) for DB2	Yes	No	No	See *DB2 Universal Database for OS/390 and z/OS Command Reference* athttp://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi.

Table 2. Application Server operations tasks (continued)

Task	MVS console	Application Server administrative console	TSO panel	Reference to associated procedure
Displaying indoubt units of work (threads) for DB2	Yes	No	No	See *DB2 Universal Database for OS/390 and z/OS Command Reference* athttp://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi.
Displaying units of work for RRS	No	No	Yes	See *z/OS MVS Programming: Resource Recovery* at http://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi.
Displaying units of work for CICS	Yes	No	Yes	See *CICS Operations and Utilities Guide* at http://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi
Displaying units of work (transactions) for IMS	Yes	No	No	See *IMS/ESA Summary of Operator Commands* at hhttp://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi.
Displaying the status of a cluster	Yes	Yes	No	See the *Administering applications and their environment* PDF.
Displaying the status of a server	Yes	Yes	No	See the *Administering applications and their environment* PDF.
Displaying active address spaces	Yes	No	No	See the *Administering applications and their environment* PDF.
Displaying active replies	Yes	No	No	See the *Administering applications and their environment* PDF.
Modify operations				
Getting help for the modify command	Yes	No	No	See the *Administering applications and their environment* PDF.
Canceling application clusters and servers	Yes	No	No	See the *Administering applications and their environment* PDF.
Modifying the Java trace string	Yes	No	No	See the *Administering applications and their environment* PDF.
Displaying status	Yes	No	No	See the *Administering applications and their environment* PDF.
Other Application Server operations				
ARM and restart	Yes	No	No	See the *Installing your application serving environment* PDF.

Table 2. Application Server operations tasks (continued)

Task	MVS console	Application Server administrative console	TSO panel	Reference to associated procedure
Setting up error log streams for different clusters and servants	No	You can associate a log stream with a cluster from the administrative console	No	See the *Troubleshooting and support* PDF.
Setting up System Management Facilities recording	Yes	Enable it from here, but initiate it from the MVS console.	No	See the *Troubleshooting and support* PDF..
Shutting down the WebSphere Application Server for z/OS environment	Yes	Application server only	No	See Stopping clusters, Stopping servers, and Steps for stopping or canceling the location service daemon from the MVS console.
Taking a WebSphere Application Server for z/OS system cluster out of service	Yes	Application server only; You cannot take a WebSphere Application Server for z/OS system cluster out of service from the administrative console	No	
Workload Management				
Displaying the status of a WLM application environment	Yes	No	No	See See the *Installing your application serving environment* PDF.
Handling workload management and server failures	Yes	No	No	See See the *Installing your application serving environment* PDF.

Table 2. Application Server operations tasks (continued)

Task	MVS console	Application Server administrative console	TSO panel	Reference to associated procedure
Getting out of the stopped state and back to the available state for an application environment	Yes	No	No	See See the *Installing your application serving environment* PDF.
Checking and managing the workload management application environment (display, stop/quiesce, restart/resume)	Yes	No	No	See See the *Installing your application serving environment* PDF and WLM dynamic application environment operator commands.

Introduction: Administrative commands

Command-line tools are simple programs that you run from an operating system command-line prompt to perform specific tasks, as opposed to general purpose administration. Using the tools, you can start and stop application servers, check server status, add or remove nodes, and complete similar tasks.

See **Reference > Commands** in the information center navigation for the names and syntax of all the commands that are available with the product. A subset of these commands are particular to system administration purposes.

Introduction: Administrative programs

The product supports a Java programming interface for developing administrative programs. All of the administrative tools supplied with the product are written according to the API, which is based on the industry standard Java Management Extensions (JMX) specification. You can write a Java program that performs any of the administrative features of the WebSphere Application Server administrative tools. You can also extend the basic WebSphere Application Server administrative system to include your own managed resources.

Introduction: Administrative configuration data

Administrative tasks typically involve defining new configurations of the product or performing operations on managed resources within the environment. IBM WebSphere Application Server configuration data is kept in files. Because all product configuration involves changing the content of those files, it is useful to know the structure and content of the configuration files.

The WebSphere Application Server product includes an implementation of the Java Management Extension (JMX) specification. All operations on managed resources in the product go through JMX functions. This setup means a more standard framework underlying your administrative operations as well as the ability to tap into the systems management infrastructure programmatically.

Welcome to basic administrative architecture

This article discusses basic concepts in the administrative architecture to help you understand system administration in a WebSphere Application Server environment. The fundamental concepts for WebSphere Application Server administration include software processes called servers, topological units referenced as nodes and cells, and the configuration repository used for storing configuration information.

Servers perform the actual running of the code. Several types of servers exist depending on the configuration. Each server runs in its own Java virtual machine (JVM). The application server is the primary run-time component in all WebSphere Application Server configurations. All WebSphere Application Server configurations can have one or more application servers. In some configurations, each application server functions as a separate entity. No workload distribution or common administration among application servers exists. In other configurations, workload can be distributed between servers and administration can be done from a central point.

A node is a logical group of WebSphere Application Server-managed server processes that share a common configuration repository. A node is associated with a single WebSphere Application Server profile. A WebSphere Application Server node does not necessarily have a one-to-one association with a system. One computer can host arbitrarily many nodes, but a node cannot span multiple computer systems. A node can contain zero or more application servers.

The configuration repository holds copies of the individual component configuration documents that define the configuration of a WebSphere Application Server environment. All configuration information is stored in .xml files.

A cell is a grouping of nodes into a single administrative domain. A cell can consist of multiple nodes, all administered from a deployment manager server. When a node becomes part of a cell (a federated node), a node agent server is installed on the node to work with the deployment manager server to manage the WebSphere Application Server environment on that node.

When a node is a standalone node, not part of a cell, the configuration repository is fully contained on the node. When a node is part of a cell, the configuration and application files for all nodes in the cell are centralized into a cell master configuration repository. This centralized repository is managed by the deployment manager server and synchronized to local copies that are held on each node. The local copy of the repository that is given to each node contains just the configuration information needed by that node, not the full configuration that is maintained by the deployment manager.

WebSphere Application Server types

This section discusses the three server types that interact to perform system administration.

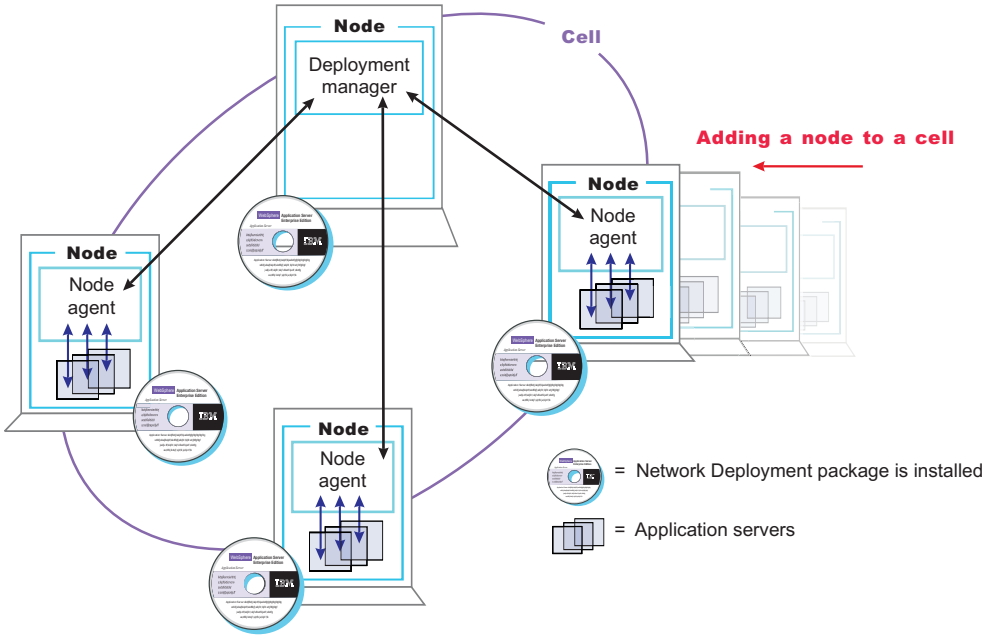
Application Server: A WebSphere Application Server provides the functions that are required to support and host user applications. An application server runs on only one node, but one node can support many application servers.

Node agent: When a node is federated, a node agent is created and installed on that node. The node agent works with the deployment manager to perform administrative activities on the node.

Deployment manager: With the deployment manager, you can administer multiple application servers from one centralized manager. The deployment manager works with the node agent on each node to manage all the servers in a distributed topology.

The following diagram depicts the concepts that are discussed in this article.

IBM WebSphere Application Server Network Deployment package



The concepts that are discussed in this article form the basis of WebSphere Application Server administration. More detailed descriptions can be found in other sections.

Introduction: Servers

Application servers

Application servers provide the core functionality of the WebSphere Application Server product family. They extend the ability of a Web server to handle Web application requests, and much more. An application server enables a server to generate a dynamic, customized response to a client request.

For additional overview, refer to “Introduction: Application servers” on page 16.

Clusters

Workload management optimizes the distribution of client processing tasks. Incoming work requests are distributed to the application servers that can most effectively process the requests. Workload management also provides failover when servers are not available, improving application availability.

Clusters are sets of application servers that are managed together and participate in workload management. The servers that are members of a cluster can be on different host machines, as opposed to the servers that are part of the same node and must be located on the same host machine.

For additional overview, refer to "Introduction: Clusters" on page 17.

Introduction: Application servers

Overview

An application server is a Java Virtual Machine (JVM) that is running user applications. The application server collaborates with the Web server to return a dynamic, customized response to a client request. Application code, including servlets, JavaServer Pages (JSP) files, enterprise beans and their supporting classes, runs in an application server. Conforming to the Java 2 platform, Enterprise Edition (J2EE) component architecture, servlets and JSP files run in a Web container, and enterprise beans run in an Enterprise JavaBeans (EJB) container.

To begin creating and managing an application server, see "Administering application servers" on page 198.

You can define multiple application servers, each running its own JVM. Enhance the operation of an application server by using the following options:

- Configure transport chains to provide networking services to such functions as the service integration bus component of IBM service integration technologies, WebSphere Secure Caching Proxy, and the high availability manager core group bridge service. See "Configuring transport chains" on page 257 for more information.
- Plug into an application server to define a hook point that runs when the server starts and shuts down. See "Custom services" on page 277 for more information.
- Define command-line information that passes to a server when it starts or initializes. See the *Administering applications and their environment* PDF for more information.
- "Tuning application servers" on page 304
- Enhance the performance of the application server JVM. See "Using the JVM" on page 288 for more information.
- Use an Object Request Broker (ORB) for RMI/IIOP communication. See the *Developing and deploying applications* PDF for more information.

Asynchronous messaging

The product supports asynchronous messaging based on the Java Message Service (JMS) of a JMS provider that conforms to the JMS specification version 1.1.

The JMS functions of the default message service in WebSphere Application Server are served by one or more messaging engines (in a service integration bus) that runs within application servers.

In a deployment manager cell, there can be WebSphere Application Server version 5 nodes. If a version 5 node is configured to use V5 default messaging (the version 5 embedded messaging), there can be at most one JMS server on that node.

Generic Servers

A generic server is a server that is managed in the WebSphere administrative domain, although it is not a server that is supplied by the WebSphere Application Server product. The generic server can be any server or process that is necessary to support the Application Server environment.

Introduction: Web servers

The application server and Web server communicate using Web server plug-ins. "Communicating with Web servers" on page 110 describes how to set up your Web server and Web server plug-in environment and how to create a Web server definition. The Web server definition associates a Web server with a previously defined managed or unmanaged node. After you define the Web server to a node, you can use the administrative console to perform the following functions for that Web server.

If the Web server is defined to a managed node, you can:

- Check the status of the Web server
- Generate a plug-in configuration file for that Web server.
- Propagate the plug-in configuration file after it is generated.

If the Web server it is defined to an unmanaged node, you can:

- Check the status of the Web server
- Generate a plug-in configuration file for that Web server.

After you set up your Web server and Web server plug-in, whenever you deploy a Web application, you must specify a Web server as the deployment target that serves as a router for requests to the Web application. The configuration settings in the plug-in configuration file (plugin-cfg.xml) for each Web server are based on the applications that are routed through that Web server. If the Web server plug-in configuration service is enabled, a Web server plug-in's configuration file is automatically regenerated whenever a new application is associated with that Web server.

Note: Before starting the Web server, make sure you are authorized to run any Application Response Measurement (ARM) agent associated with that Web server.

Refer to your Web server documentation for information on how to administer that Web server. For tips on tuning your Web server plug-in, see the *Tuning* PDF.

Introduction: Clusters

Clusters are groups of servers that are managed together and participate in workload management. A cluster can contain nodes or individual application servers. A node is usually a physical computer system with a distinct host IP address that is running one or more application servers. Clusters can be grouped under the configuration of a cell, which logically associates many servers and clusters with different configurations and applications with one another depending on the discretion of the administrator and what makes sense in their organizational environments.

Clusters are responsible for balancing workload among servers. Servers that are a part of a cluster are called cluster *members*. When you install an application on a cluster, the application is automatically installed on each cluster member.

Node groups bound clusters. All cluster members of a given cluster must be members of the same node group. For more information about clusters and node groups, see "Clusters and node groups" on page 308.

To learn more about clusters, see "Clusters and workload management" on page 308 and "Balancing workloads with clusters" on page 307 for more information.

Core groups

A group of clusters can be defined as a *core group*. All of the application servers defined as a member of one of the clusters included in a core group are automatically members of that core group. Individual application servers that are not members of a cluster can also be defined as a member of a core group. The use of core groups enables WebSphere Application Server to provide high availability for applications that must always be available to end users. You can also configure core groups to communicate with each other using the *core group bridge*. The core groups can communicate within the same cell or across cells.

To learn more about core groups, see "Setting up a high availability environment" on page 368.

Introduction: Environment

The environment of the product applies to the configuring of Web server plug-ins, variables, and objects that you want consistent throughout a cell.

Cell-wide settings

Cell-wide settings are sets of configuration data that are stored in files in the cell directory. These configuration files are replicated to every node in the cell. Several different configuration settings apply to the entire cell. These settings include the definition of virtual hosts, shared libraries, and any variables that must be consistent throughout the entire cell.

For more information, refer to "Introduction: Cell-wide settings."

Variables

A variable is a configuration property that can be used to provide a parameter for any value in the system. A variable has a name and a value to use in place of that name wherever the variable name is located within the system.

For more information, refer to "Configuring WebSphere variables" on page 165.

Introduction: Cell-wide settings

The configuration data for WebSphere Application Server is stored in XML files. The XML files exist in one of several directories in the configuration repository tree.

The directory in which a configuration file exists determines its scope, or how broadly or narrowly that data applies. Files in an individual server directory apply to that specific server only. Files in a node-level directory apply to every server on that node. Files in the cell directory apply to every server on every node within the entire cell.

Cell-wide settings are configuration files in the cell directory. The files are replicated to every node in the cell. Several different configuration settings apply to the entire cell. These settings include the definition of virtual hosts, shared libraries, and any variables that you want consistent throughout the entire cell.

Introduction: Variables

Variables in the WebSphere environment come in many varieties. Variables are used to control settings and properties relating to the server environment. Three main variable options that are important for a WebSphere Application Server user to know and understand are custom properties, environment variables, and WebSphere-specific variables.

Environment variables

Environment variables, also called *native environment variables*, are not specific to the 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 are specified in the administrative console. Click **Application Server >***server_name***> Process Definition > Servant Process > 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 used for three purposes:

- Configuring WebSphere Application Server path names, such as JAVA_HOME, and APP_INSTALL_ROOT.
- Configuring certain cell-wide customization values.
- Configuring the WebSphere Application Server for z/OS location service.

WebSphere variables are specified in the administrative console by clicking **Environment > Manage WebSphere variables**. How the WebSphere variable is set determines its scope. A variable can apply to a cell, a node, or a server. 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 HTTP transport custom properties, follow one of the following paths:

- **Servers > Application Servers >** *server_name* **> Web Container > HTTP Transport > Additional Properties > Custom Properties**
- **Servers > Application Servers >** *server_name* **> Web Container > Additional Properties > 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 the HTTP transport custom properties page apply only to that specific transport. If the same properties are set on both pages, the settings on the transport page override the settings that are defined on the Web container page for that specific transport.

Chapter 3. How do I administer applications and their environments?

- Establish the application serving environment
- Secure the application serving environment - see the *Security* PDF.
- Set up Web access for applications
- Set up resources for applications to use
- Configure class loaders - see the *Developing and deploying applications* PDF.
- Deploy and administer applications
- Use the administrative clients
- Troubleshoot deployment and administration

Legend for ″How do I?...″ links

Documentation	Show me	Tell me	Guide me	Teach me
Refer to the detailed steps and reference	Watch a brief multimedia demonstration	View the presentation for an overview	Be led through the console pages	Perform the tutorial with sample code
Approximate time: Varies	**Approximate time:** 3 to 5 minutes	**Approximate time:** 10 minutes+	**Approximate time:** 1/2 hour+	**Approximate time:** 1 hour+

Establish the application serving environment

The following tasks involve establishing application serving capability in your network environment, whether you use single or clustered application servers. Servers can be grouped into administrative domains known as nodes and cells.

See also the overview:

- Version 6 topology and terminology

Administer nodes

A node is a grouping of managed servers. Use this task to view information about and manage nodes.

Show me

Tell me:

- Add and remove nodes
- Manage node groups
- Cell, deployment managers, nodes, and node agents

Administer 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. A node agent is created automatically when a node is added to a cell.

Show me Tell me

Administer cells

When you installed the WebSphere Application Server Network Deployment product, a cell was created. A cell provides a way to group one or more nodes of your Network Deployment product. You probably will not need to reconfigure the cell. Use this task to view information about and manage a cell.

Show me Tell me

Administer configurations

Application server configuration files define the available application servers, their configurations, and their contents. You should periodically save changes to your administrative configuration. You can change the default locations of configuration files, as needed.

Tell me:

- Repository
- Archives

Configure 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. 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.

Tell me

Administer application servers

Create, configure, and operate application server processes. An application server configuration provides settings that control how an application server provides services for running enterprise applications and their components.

Show me Tell me

Administer other server types

One step in the process of creating an application server is to specify a template. A server template is used to define the configuration settings of the new server. You have the option of specifying the default server template or choosing a template that is based on a server that already exists. The default template will be used if you do not specify a different template when you create the server.

You can create other types of servers, to represent Web servers in your topology, or for other purposes. There are two types of *generic* servers: (1) Non-Java applications or processes, or (2) Java applications or processes. A *custom service* provides the ability to plug into a WebSphere application server to define a hook point that runs when the server starts and shuts down.

Tell me:

- Generic servers

Balance workloads by clustering application servers

To monitor application servers and manage the workloads of servers, use server clusters and cluster members provided by the Network Deployment product.

Show me

Tell me:

- WLM details
- Data replication service

Establishing high availability (HA) for failover

Planning ahead for high availability support is important in order to avoid the risk of a failure without failover coverage. The application server runtime of the infrastructure managed by a high availability manager includes such entities as cells and clusters. These components relate closely to core groups, high availability groups, and the policy that defines the high availability infrastructure. In a properly configured high availability environment, a high availability manager can reassess the environment it is managing and accept new components as they are added to the environment.

Tell me:

- Overview
- Details, core groups

Administer the UDDI registry

The UDDI Registry is supplied as a J2EE application file, uddi.ear. Change its configuration properties using the assembly tools. You can use either the WebSphere Application Server administrative console or the Java Management Extensions (JMX) management interface to manage UDDI Registries.

Tell me

Set up Web access for applications

These tasks involve enabling HTTP requests for applications on the application server.

Administer communication with Web servers (plug-ins)

The product provides plug-ins for supported Web servers, to enable the Web servers to pass requests to the application server, for applications running on the application server. See also the Web server related tasks in the *Installing your application serving environment* PDF.

Show me Tell me

Administer HTTP sessions

Configure the service that the product provides for managing HTTP sessions: Session Manager.

Show me

Administer IBM HTTP Server Version 6.*x*

The product provides a complementary Web server with its own documentation that can be installed into the information center.

Set up resources for applications to use

Make a variety of resources available to your applications that are deployed on the application server.

Provide access to naming and directory resources (JNDI)

Configure naming. Naming is used by clients of WebSphere Application Server applications to obtain references to objects related to those applications, such as Enterprise JavaBeans (EJB) homes. These objects are bound into a mostly hierarchical structure, referred to as a name space. The name space structure consists of a set of name bindings, each consisting of a name relative to a specific context and the object bound with that name.

Tell me:

- Introduction
- Basic concepts
- Advanced concepts
- Examples
- Troubleshooting

Provide access to relational databases (JDBC resources)

Configure data sources that applications use to access the data from databases.

Show me:

- Cloudscape
- DB2
- Oracle

Tell me

Provide access to messaging resources (default messaging provider)

Use one of various ways to implement a messaging provider for use with WebSphere Application Server. A messaging provider enables use of the Java Messaging Service (JMS) and other message resources in the product.

Show me

Tell me

Use IBM service integration technologies

Tell me:

- Overview
- Architecture
- Mediation

Establish workload balancing and high availability (HA) of messaging engines

Tell me

Access Service Integration (SI) bus resources

Show me

Tell me:

- Service integration bus resources
- JMS resources for service integration bus

Deploy and administer applications

These tasks involve deploying applications onto the application server, then administering the applications.

Install applications

Installable modules include enterprise archive (EAR), enterprise bean (EJB), Web archive (WAR), resource adapter (connector or RAR), and application client files.

Show me Tell me

Start and stop applications

You can start an application that is not running (has a status of Stopped) or stop an application that is running (has a status of Started).

Show me Tell me

Update applications

Update deployed applications or modules using the administrative console or **wsadmin** scripting. Learn which changes are candidates for hot deployment and dynamic reloading, in which you can make various changes to applications and their modules without having to stop the server and start it again.

Tell me Teach me

Deploy applications rapidly (WebSphere Rapid Deployment)

Take advantage of new rapid deployment capabilities. WebSphere rapid deployment offers the following advantages: You do not need to assemble your J2EE application files prior to deployment. You do not need to use other installation tools mentioned in this table to deploy the files. Refer to the **Rapid deployment tools** documentation in the information center.

Enhanced EAR files

Tell me Teach me

Deploy and administer Web services applications

To deploy Web services that are based on the Web Services for Java 2 platform, Enterprise Edition (J2EE) specification, you need an enterprise application, also known as an enterprise archive (EAR) file that has been configured and enabled for Web services. You can use either the administrative console or the wsadmin scripting interface to deploy an EAR file.

Show me Tell me

Use the administrative clients

A variety of tools are provided for administering the product.

Choose an administrative client

Learn about and decide among the available administrative clients, including a graphical console, scripting (wsadmin), command line tools, and Java Management Extensions (JMX) programs.

Tell me

Use the administrative console

The administrative console is a Web-based tool that you use to administer the product. The administrative console supports a full range of product administrative activities.

Show me Tell me

Use scripting (wsadmin)

Scripting is a non-graphical alternative that you can use to configure and manage WebSphere Application Server. The WebSphere Application Server **wsadmin** tool provides the ability to run scripts. The tool supports a full range of product administrative activities.

Tell me

Troubleshoot deployment and administration

Troubleshoot problems that occur when you are deploying applications onto the application server, or when you are administering an established application serving environment.

Chapter 4. Setting up the application serving environment

This topic summarizes the contents of the documentation that helps you set up your application serving environment. This information is for administrators, particularly those performing installation, customization, and maintenance of topologies.

"Configuring the product after installation"

This topic describes what to do after installing the product.

"Configuring ports" on page 110

This topic provides information about port number settings for Version 6 and previous versions, for use in coexistence and interoperability situations.

"Communicating with Web servers" on page 110

This topic describes how to install and configure WebSphere plug-ins for Web servers, enabling communication between Web servers and application servers.

"Setting up the administrative architecture" on page 113

This topic describes how to set up logical administrative domains, including cells and nodes.

"Configuring the environment" on page 156

This topic describes how to configure cell-wide settings for virtual hosts, variables and shared libraries to assist in handling requests among Web applications, Web containers, and application servers.

"Working with server configuration files" on page 190

This topic describes how to change the default locations of configuration files, as needed. Application server configuration files define the available application servers, their configurations, and their contents.

"Administering application servers" on page 198

This topic describes how to configure individual application servers to provide services for running enterprise applications and their components.

"Balancing workloads with clusters" on page 307

This topic describes how to configure clusters, which are sets of servers that are managed together and participate in workload management.

"Setting up a high availability environment" on page 368

This topic describes planning ahead for high availability support, which is important in order to avoid the risk of a failure without failover coverage. In a properly set up high availability environment, a high availability manager can reassess the environment it is managing and accept new components as they are added to the environment.

Configuring the product after installation

Use this task to configure WebSphere Application Server for z/OS application serving environments for your z/OS target systems.

- Choose a z/OS target system and complete the steps for installing the product and additional software and for preparing the base operating system that are described in the *Installing your application serving environment* PDF.
- Choose a WebSphere Application Server for z/OS configuration (practice, stand-alone or Network Deployment cell) and complete the steps for planning for product configuration that are described in the *Installing your application serving environment* PDF.

Configuring a WebSphere Application Server for z/OS application serving environment consists of setting up the WebSphere Application Server for z/OS configuration directory for the environment, making any required changes to the z/OS target system that pertain to the particular application serving environment, and starting the new environment to verify the configuration. Configuring these application serving environments after product installation requires a fair amount of planning and coordination. If you have not previously configured WebSphere Application Server for z/OS, we recommend that you configure a "practice" stand-alone application server using the default options, then proceed to configure your actual desired product configuration. See the *Installing your application serving environment* PDF for more information.

If you have already created a Network Deployment cell, follow the instructions in this section to expand the cell by creating a new managed node or federating an existing stand-alone application server node into the Network Deployment cell.

WebSphere Application Server for z/OS application serving environment nodes are created using the ISPF-based Customization Dialog. Once a node is configured and running, make further changes using the Web-based administrative console or scripting.

After you have installed the WebSphere Application Server for z/OS product, prepared your z/OS target system(s), and planned your new application serving environment, perform the tasks in this section to configure and start the application serving environment.

1. Review the use of the Customization Dialog. See “Using the Customization Dialog” on page 31 for more information.
2. If this application serving environment uses a new security domain, create the security domain before proceeding. See “Creating a security domain” on page 37 for detailed instructions.
3. Follow the directions for the type of application serving environment you wish to configure:
   - “Creating a stand-alone application server cell” on page 44
   - “Creating a Network Deployment cell” on page 67
   - “Creating a managed server node” on page 85
   - “Federating a stand-alone application server into a Network Deployment cell” on page 100

Once your application serving environment is up and running, you can install and test applications. You may also wish to configure your Web servers to interact with WebSphere Application Server for z/OS.

See the Related Tasks section for additional tasks you can perform once your application serving environment is configured.

Using the Customization Dialog

This article provides general information on starting and using the Customization Dialog. See the instructions for each customization task for detailed directions on using the Customization Dialog to perform that particular task.

The WebSphere Application Server for z/OS Customization Dialog is an ISPF Dialog, running under TSO, that you use for the initial setup of WebSphere Application Server for z/OS cells and nodes. The Customization Dialog itself does not create the cells and nodes. Instead, it creates batch jobs, scripts and data files that you then use to perform WebSphere Application Server for z/OS customization tasks.

Note: In WebSphere Application Server for z/OS, you must use the Customization Dialog and the jobs it generates to create new cells and nodes. Once you have created a stand-alone application server or Network Deployment cell, however, you use the WebSphere Application Server for z/OS administrative console or scripting to administer it.

The Customization Dialog consists of a set of ISPF panels, file-tailoring skeletons, message libraries, CLISTs and REXX execs which are installed as part of the WebSphere Application Server for z/OS product. It is intended for use under TSO by a systems programmer or WebSphere Application Server for z/OS administrator who is familiar with the z/OS target system on which the resulting WebSphere Application Server for z/OS cells and nodes will run.

The Customization Dialog uses ISPF variables to hold the various values used to create WebSphere Application Server for z/OS customization jobs, scripts and files. See "Working with Customization Dialog variables" on page 33 for more information.

Follow these steps to perform most Customization Dialog customization and migration tasks:

Note: See the *Migrating, coexisting, and interoperating* PDF. for information on the migration portion of the Customization Dialog.

1. Start the Customization Dialog. See "Starting the Customization Dialog" for instructions.
2. Choose a set of configuration data sets to hold the generated jobs and files for this task.
3. Set the Dialog variables to appropriate values. See "Working with Customization Dialog variables" on page 33 for instructions.
4. Generate the customized jobs, scripts and files based on the Dialog variable values you provided, and place them in the configuration data sets.
5. Move the configuration data sets to the z/OS system on which you will perform WebSphere Application Server for z/OS customization or migration tasks.
6. Follow the generated instructions in the .CNTL configuration data set to complete the customization or migration task.

Starting the Customization Dialog

Before you can start the Customization Dialog, you must install the WebSphere Application Server for z/OS product and make the product libraries available to the z/OS system on which the Customization Dialog will run. All product libraries

should be cataloged. The Customization Dialog does not use the product directory (/usr/lpp/zWebSphere/V6R0 by default), but you must mount it and make it available before the Customization Dialog's generated jobs can run.

You will need the ability to log on to TSO from a real or emulated 3270-type terminal. Your logon display must support 3270 emulation and be set to a minimum of 32 rows by 80 columns (32 x 80) in order for the ISPF Customization Dialog to run.

- If your terminal has exactly 32 display rows, be sure to issue the `PFSHOW OFF` command in ISPF to hide PF key settings. These can overlay Customization Dialog input fields.
- If you have a 32-row display and use the ISPF split screen function, deselect "Always show split line" on the ISPF Settings panel and split the screen at the extreme top or bottom of the display. This prevents the split screen line from displaying and lines in the Customization Dialog from being obscured. Other uses of split screen will obscure lines in the Customization Dialog.

The following steps outline how to change your display size setting if you use the IBM Personal Communications Workstation program. Complete all the steps before you start your TSO session.

1. In the menu bar of the session window, select "Communication."
2. From the Communication window, select "Configure..."
3. In the window that appears, press the "Session Parameters..." button.
4. Use the pull-down menu to select a Screen Size setting between 32x80 and 62x160.
5. Press the "OK" button until you are back at the session window.

You will need a TSO user ID that has READ access to the WebSphere Application Server for z/OS product libraries.

Starting the Dialog: The BBOWSTRT command

To start the customization dialog, log on to TSO and invoke the BBOWSTRT exec from the SBBOCLIB product library using the following EXEC command.

Note: If you enter the command from TSO Option 6, just type in the command. If you enter the command on the ISPF command line, you must prefix it with "TSO".

```
EXEC 'was_hlq.SBBOCLIB(BBOWSTRT)' 'options'
```

where

was_hlq
is the data set name high level qualifier for your WebSphere Application Server for z/OS product libraries.

Note: The Dialog libraries (SBBOCLIB, SBBOPxxx, SBBOMxxx, SBBOSLIB and SBBOSLB2) must have the same data set name high level qualifier in order for the Customization Dialog to allocate the libraries correctly.

Be sure to use the was_hlq value that corresponds to the level of WebSphere Application Server for z/OS product code that you will use to run the resulting WebSphere Application Server for z/OS cell(s).

You can also specify the following options, separated by spaces and surrounded in a single set of quotes:

APPL(*value*)
Specifies the application name you intend to use. This provides for separate sets of Customization Dialog variables saved from one Customization Dialog session to the next. The default value is BBO6. See "Working with Customization Dialog variables" for details.

LANG(*value*)
Specifies the national language you wish the Customization Dialog use. Specify "ENUS" for English or "JAPN" for Japanese. The default value is ENUS.

Note: Some messages will still appear in English even if you specify another national language using the LANG option.

PROD(*list*)
Specifies a list of WebSphere Application Server for z/OS add-on products which are also customized using the Dialog. The list should consist of one or more three-character product identifiers from the following table, separated by spaces. If you wish to not customize any add-on products, omit the PROD option.

BBZ	WebSphere Business Infrastructure

PRODHLQ(*list*)
Specifies a list of high-level qualifiers for the WebSphere Application Server for z/OS add-on products you specified with the PROD option. The list should consist of one high-level qualifier for each entry in the PROD list, given in the same order. If you wish to not customize any add-on products, omit the PRODHLQ option.

If you are starting the Dialog for the first time or using a new APPL value, you will first see a copyright screen. Press ENTER to continue.

You should then see the Customization Dialog main menu, similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>                                                    Appl: BBO6

   Use this dialog to create WebSphere Application Server for z/OS
   cells and nodes.  Specify an option and press Enter.


   1  Configure a security domain.

   2  Create stand-alone application server nodes.  You must complete
      Option 1 before starting this option.

   3  Create Network Deployment cells and nodes.  You must complete
      Option 1 before starting this option.

   4  Migrate V5.x Nodes to V6 Nodes.
```

If you do not see the main menu, review the steps and requirements above.

Working with Customization Dialog variables

The WebSphere Application Server for z/OS Customization Dialog uses ISPF variables to store customization values. Although it is possible to re-enter these

values every time you use the Customization Dialog, this process can be time-consuming and error-prone. The Customization Dialog provides two methods you can use to save Dialog variable values from one session to the next: ISPF variable pools and Customization Dialog save files.

ISPF variable pools

Variable pools are a feature of ISPF that allow the Dialog variables for one application to reside separate from those for other applications. Whenever you start the Dialog using the `ISPSTART` command, you can use the `NEWAPPL` option to assign a one to four character name to the application and its variable pool. Variable pool names must be one to four characters long and alphanumeric with a beginning alphabetic letter. The default variable pool name for the WebSphere Application Server for z/OS Version 6.0.1 Customization Dialog is BBO6.

When you exit ISPF normally, the current editor settings and Dialog variable values for each variable pool are saved in a pair of members in the user ID's ISPPROF (ISPF profile) data set. The data set name of the ISPPROF data set is usually something like userid.ISPF.ISPPROF, but it can vary from installation to installation. If saving setting for application xxxx, the editor settings are saved in member xxxxEDIT and the variable settings are saved in member xxxxPROF.

You can use the `APPL` option when starting the WebSphere Application Server for z/OS Customization Dialog to specify the application name (variable pool) you wish to use:

```
EXEC 'was_hlq.SBBOCLIB(BBOWSTRT)' 'APPL(xxxx)'
```

Whenever you start the Customization Dialog for the first time with a particular application name, the product copyright page is displayed before the main menu appears. Each Customization Dialog task menu panel shows the application name in the upper right-hand corner, labelled "Appl:".

By using different application names, you can separate variable pools for different WebSphere Application Server for z/OS releases, target systems, or cells. Just be sure to specify the same APPL value whenever you start the Customization Dialog in order to work with a particular WebSphere Application Server for z/OS customization task. You should also use customization save files (described below) to provide long-term, more easily documented backup of customization variable values.

To delete all information from an ISPF variable pool, delete the xxxxEDIT and xxxxPROF members from the ISPPROF data set.

Note: Ensure when you do this that the ISPF application that uses the variable pool is not active.

If an ISPF Dialog such as the WebSphere Application Server for z/OS Customization Dialog terminates abnormally, the variable pool members in the ISPPROF data set will not be updated.

Customization Dialog save files

The Customization Dialog provides its own method for saving and reusing Dialog variable values. From most task menus, you can type `S` on the command line to save the current Dialog variables in a file, and type `R` on the command line to

restore Dialog variables from a previously-created save file. These files are ordinary z/OS sequential files (RECFM=VB, LRECL=255) with one variable/value pair per line.

Because values in save files are explicitly saved and restored, you are somewhat less likely to use them accidently when customizing a different cell than values stored in the ISPF profile. Also, you can use the save file name to identify the WebSphere Application Server for z/OS cell or node for which they are created, the WebSphere Application Server for z/OS version and release, and so on. You can use a Customization Dialog save file, together with the WebSphere Application Server for z/OS product files, to recreate the WebSphere Application Server for z/OS customization data sets (.CNTL and .DATA) at any time.

One save file that is particularly important is the security domain save file. You should save the security domain values during the creation of each WebSphere Application Server for z/OS security domain, and import them whenever you create a new stand-alone application server or Network Deployment cell in that security domain. Within the Dialog panels for cell creation, you can view the security domain settings but not change them. The panels have specific numbered options for saving the newly created security domain values and restoring them during cell creation, so you don't forget.

Once you have finishing setting customization variable values for a new stand-alone application server or Network deployment cell, save the variable values before proceeding with customization. This allows you to more easily restart the customization process if you discover you have made an error during customization. You can also import the appropriate saved values when performing tasks, such as federating a stand-alone application server or creating a new managed node, on an existing Network Deployment cell.

The following are some cautions you should take when working with Customization Dialog save files:

- Customization Dialog save files may contain unencrypted passwords and other sensitive information. Make sure that access to the save files is restricted.
- Save files created with the S command include the security domain variables as well. Thus, if you need to restore values from both a regular save file and a security domain save file, restore the security domain save file last.
- Do not edit Customization Dialog save files directly. Instead, restore them in a Customization Dialog session, make the appropriate changes, and resave the values.

You can also use Customization Dialog save files to check for typing errors and similar problems by sorting the save file on the columns containing the Dialog variable values. This can help detect typographic errors in product data set names, etc., and also detect ports or UNIX UID/GID values that are accidently reused within a particular configuration.

The instructions for each configuration task provide guidance on restoring and saving Customization Dialog variable values.

Customization Dialog commands and function keys

The WebSphere Application Server for z/OS Customization Dialog uses only a small number of application-specific primary commands which are listed on the panels on which you can enter them. Of these, the most important are the S (Save) and R (RESTORE) commands for working with WebSphere Application Server for

z/OS customization variable save files. See “Working with Customization Dialog variables” on page 33 for more information.

The following default function keys are used throughout the Customization Dialog:

PF1
Displays a help screen for the current panel.

Note: Within the help panels, slightly different PF key settings are used. Press PF1 from within any help panel for details.

PF2
Sets split screen mode. See “Starting the Customization Dialog” on page 31 for restrictions on split screen mode in the Customization Dialog.

PF3
Exits the current panel.

PF4
Exits the Customization Dialog.

PF7
Scrolls up.

PF8
Scrolls down.

PF9
Swaps screens in split screen mode.

PF10
Scrolls left.

PF11
Scrolls right.

PF12
Retrieves the previous command.

You may also find the following ISPF primary commands useful:

EPDF
Allows you to browse or edit a z/OS data set from the ISPF command line

MSGID
Turns the display of message identifiers on and off.

PANELID
Turns the display of panel names on and off.

PFSHOW OFF
Turns off the display of PF key settings.

Note: This is necessary when using a 32-row display.

PFSHOW ON
Turns on the display of PF key settings.

TSO
Executes a TSO command, CLIST or REXX command procedure from the command line.

ZKEYS
Allows you to display or change the current function key settings.

Creating a security domain

Perform this task to set up the operating system security prerequisites for a WebSphere Application Server for z/OS cell. This ensures that all servers in the cell are using the same operating system security definitions.

Install the WebSphere Application Server for z/OS product code and review the instructions for using the Customization Dialog. Have available a copy of the worksheet that you completed as part of planning a security domain. (See the *Securing applications and their environment* PDF for more details.

You must perform this task before configuring any application serving environment that uses the security domain. If a new WebSphere Application Server for z/OS cell or server on a z/OS system will use the exact same security domain definitions as an existing server or cell on the same z/OS system, you do not need to repeat this task.

You need to run the jobs generated as part of this task once per security database. If z/OS systems which do not share a RACF or other security database, you are responsible for making sure identical security definitions are in place for all WebSphere Application Server for z/OS user IDs, groups, and profiles. See the *Securing applications and their environment* PDF for more information.

1. Log on to TSO on the z/OS system on which you intend to configure the security domain. Use a user ID that has READ access to the WebSphere Application Server for z/OS product data sets.
2. Start the Customization Dialog. See "Starting the Customization Dialog" on page 31 for details.
3. Choose the configuration data sets in which you will store your customization jobs and data. See "Choosing configuration data sets" on page 38 for details.
4. Set the customization variables according to the values recorded on your security domain worksheet. See "Setting the customization variables: Security domain" on page 38 for details.
5. Save the security domain customization variables in a data set that you will use in later customization steps. See "Saving the security domain variables" on page 40 for details.
6. Create the customization jobs and files, based on the customization variable values you entered. See "Creating the customization jobs and files" on page 41 for details.
7. Follow the generated customization instructions. See the *Securing applications and their environment* PDF for details, and a sample set of customization instructions.

You are done when you have successfully completed the steps in the generated instructions. The security domain is in place on the chosen z/OS system. If any z/OS systems that interoperate with or host your planned application serving environment do not share the security database you updated as part of this task, update the other systems' security databases accordingly.

Note: In the case of SSL certificates, this may require transporting certificates created on the initially configured z/OS system to the other z/OS systems rather than creating new SSL certificates on each system.

Proceed with the configuration of the application serving environments that use this security domain.

Choosing configuration data sets

This article leads you through the ″Allocate target data sets″ option in the Customization Dialog.

You must start the Customization Dialog and select the ″Configure a security domain″ option.

Each option in the Customization Dialog saves customization jobs and files in a pair of customization data sets. While is it possible to reuse these data sets, it is safest to create separate data sets for each WebSphere Application Server for z/OS configuration. We recommend that you use the customization data set name prefix (sometimes referred to as ″config_hlq″) to indicate the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring. For example, you might use the following data set name prefix for configuring a WebSphere Application Server for z/OS 6.0.1 security domain for cell PRODCELL:

SYSPROG1.WAS601.PRODCELL.SECD

Complete this task before generating the customization jobs and files.

1. On the main Dialog panel, type ″1″ in the *Option* field to select ″Allocate target data sets″.
2. Press Enter. **Result:** You see a panel that looks similar to the following:

```
---- WebSphere Application Server for z/OS Customization ------------
Option  ===>

 Allocate Target Data Sets

 Specify a high level qualifier (HLQ) and press Enter to allocate the
 data sets to contain the generated jobs and instructions. You can
 specify multiple qualifiers (up to 39 characters).

 High level qualifier:                                          .CNTL
                                                                .DATA

 The Dialog will display data set allocation panels. You can make
 changes to the default allocations, however you should not change
 the DCB characteristics of the data sets.

    .CNTL  - a PDS with fixed block 80-byte records to
             contain customization jobs.

    .DATA  - a PDS with variable length data to contain
             other data produced by the Customization Dialog.
```

3. Fill in your chosen configuration data set name prefix value (config_hlq). If the data sets ″config_hlq.CNTL″ and ″config_hlq.DATA″ do not exist, you will be prompted for data set allocation information. If the data sets already exist, a message will inform you that they will be reused.

The data sets ″config_hlq.CNTL″ and ″config_hlq.DATA″ are allocated and will store customization jobs and files. These data set names will also be saved along with the customization variables.

Setting the customization variables: Security domain

This article describes how to complete the ″Define variables″ option for a WebSphere Application Server for z/OS security domain.

You must start the Customization Dialog and select the 'Configure Security Domain' option. Have the `Customization Dialog worksheet: Security domain` completed and at hand. (See the *Securing applications and their environment* PDF for more details.)

1. On the 'Configure Security Domain' panel, type "2" in the *Option* field to select "Define variables" and press **Enter**.
2. Fill in the security domain Define Variables panels using the following screen shots and tips as your guides. When you are done with each panel, press **Enter**.

   Security Domain Define Variables Panel (1 of 2)

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Security Domain Configuration (1 of 2)

   Specify the following to customize the security domain to be selected
   when configuring one or more servers or cells, then press Enter
   to continue.

 Use security domain identifier in RACF definitions:  N
     Security domain identifier.....................:

 WebSphere Application Server Configuration Group Information
   Group....:  WSCFG1         GID..:  2500

 WebSphere Application Server Administrator Information
   User ID..:  WSADMIN        UID..:  2403
   Password.:  WSADMIN

 Unauthenticated User Definitions for Base Servers
   User ID..:  WSGUEST        UID..:  2402
   Group....:  WSCLGP         GID..:  2502

 WebSphere Application Server Asynchronous Administration Task
   User ID..:  WSADMSH        UID..:  2504

 WebSphere Application Server Servant Group Information
   Group....:  WSSR1          GID..:  2501

 Configure for local OS security registry..........:  Y
```

   Tips:

   - If you have selected SecurityDomainType = 'cell-qualified', then set 'Use security domain identifier in RACF definitions:' to 'Y', and enter your chosen security domain name on the next line. If you have selected SecurityDomainType = 'none', then set 'Use security domain identifier in RACF definitions:' to 'N'.
   - Enter the user ID and UID value for each user ID on the security domain worksheet, and the group name and GID value for each group on the security domain worksheet.
   - Set 'Configure for local OS security registry' to 'Y' if you intend to use RACF (or equivalent) as your WebSphere user registry; set it to 'N' if you plan to use LDAP or a custom user registry instead.

   Security Domain Define Variables Panel (2 of 2)

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Security Domain Configuration (2 of 2)

   Specify the following to customize the security domain to be selected
   when configuring one or more servers or cells, then press Enter
```

```
  to continue.

SSL Customization

  Certificate authority keylabel..........:  WebSphereCA
  Generate certificate authority (CA) certificate:  Y
  Expiration date for CA authority:  2010/12/31
  Default RACF keyring name.........:  WASKeyring
  Enable SSL on location service daemon:  N

Additional z/OS Security Customization Options
  Generate default RACF realm name:  N
     Default RACF realm name ....:  MCLXCF01

  Use SAF EJBROLE profiles to enforce J2EE roles:  N

  Enable SAF authentication using LTPA or ICSF login tokens:  Y
```

Saving the security domain variables

You must start the Customization Dialog and fill in the security domain variables.

The security domain settings are used in the customization of every WebSphere Application Server for z/OS cell. By saving the security domain variables, you create a saved security configuration that you can use consistently across all nodes in the cells you create for a given security domain.

Complete this task after setting the security domain variables and before moving on to later customization steps. If you encounter problems during customization and change the security domain variable values, be sure to re-save them.

Note: This procedure applies to only the ″Configure a security domain″ Dialog option. For information about saving variables for all the other Dialog options, see “Saving the cell variables” on page 73

1. On the Configure Security Domain panel, type ″3″ in the *Option* field to select ″Save security domain variables″ and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Save Security Domain Variables

  Specify the name of a sequential data set to contain the security domain
  variables, then press Enter to continue. If the data set does not exist,
  the Dialog displays the Allocate New Data Set panel, with which you
  can allocate a data set.


 Data set name:
```

2. Fill in the name of the sequential data set you will use to hold the security domain variable values. Choose a data set name that identifies the sysplex, cell or group of cells for which this security domain is defined. Enclose the data set name in single quotes. If the data set does not exist, you will be prompted for data set allocation information.
3. Record the name of the security domain variable data set on your security domain worksheet.

The security domain settings are saved in the data set you selected.

Creating the customization jobs and files

You must select configuration data sets to use and complete the process of defining variables for this task. See “Choosing configuration data sets” on page 38 and “Setting the customization variables: Security domain” on page 38 for more information.

The Customization Dialog creates customization batch jobs and data files, based on the variable values you specified in the Dialog. The batch jobs and data sets will be written to the config_hlq.CNTL and config_hlq.DATA configuration data sets that you created with the 'Allocate target data sets' option.

1. Ensure the configuration data sets are allocated and not in use.

   Note: Editing a member in config_hlq.CNTL or config_hlq.DATA will cause this task to fail.

2. On the 'Configure Security Domain' panel, type "4" in the *Option* field to select "Generate customization jobs" and press **Enter**. You will have one of two results:

   - **Result A:** If all variables are defined correctly, you see the 'Specify Job Cards' panel, which looks similar to this:

     ```
     ------------  WebSphere Application Server for z/OS Customization     --------
     Option  ===>

     Generate Customization Jobs

      This portion of the Customization Dialog generates the jobs you must
      run after you complete this Dialog process. You must complete the
      customization process before you generate the jobs with this step.
      If you have not done this, please return to that step.

      Jobs and data files will get generated into data sets:
        'hlq.CNTL'
        'hlq.DATA'
      If you wish to generate customization jobs using other data sets, then
      exit from this panel and select option 1 (Allocate target data sets).

      All the jobs that will be tailored for you will need a job card.
      Please enter a valid job card for your installation below. The
      file tailoring process will update the jobname for you in all the
      generated jobs, so you need not be concerned with that portion of
      the job cards below. If continuations are needed, replace the
      comment cards with continuations.

      Specify the job cards, then press Enter to continue.

      //jobname  JOB (ACCTNO,ROOM),'userid',CLASS=A,REGION=0M
      //*
      //*
      //*
     ```

     Note: Pay particular attention to the displayed target data sets. Make sure they are the ones you intend to use.

   - **Result B:** If the variables are not defined correctly, you will see the 'Verification' panel. Decide whether the warnings or errors are serious enough to warrant returning to the "Define variables" option.

     Note: If the return code is "8" or greater, return to the "Define variables" option and fix the uncovered problems. If you saved the variables previously, be sure to re-save them after making any updates.

3. Fill in the job card information, according to your installation requirements. For each job, the Dialog generates a jobname and the ″JOB″ keyword to match the member name of the PDS, but you specify the rest.

   Note: If you need to run these jobs on a particular system in the sysplex (for example, JES2 MAS or JES3 complex), you should specify the necessary Scheduling Environment (SCHENV), JES2 JOBPARM, or JES3 //*MAIN statement at this time.

   Example of a job card entry:

   ```
   //jobname JOB 1234,USER1,NOTIFY=????,MSGCLASS=O,REGION=0M
   //*            USER=SYSADM1,PASSWORD=SYSADM1
   /*JOBPARM SYSAFF=SYSB
   ```

   Note: This example is useful for jobs that require a user ID other than that of the logged-on TSO user. (This is typically a user ID with UID=0.) In that case, you can just put a comma at the end of the first line, put in the correct user ID on the second line, then uncomment that second line.

   You may wish to use RACF SUBMIT authority to avoid having to keep passwords in your configuration data sets.
4. Fix any errors. If there are errors anywhere, you will see the 'Error' panel. Press PF3 to exit the error panel, then enter the correct panel to fix the errors. Then return to the ″Generate Customization Jobs″ option and pick up where you left off. If necessary, you can update the variables and rerun this option. The generation process will delete and re-tailor all the members.

   Note: Compress the configuration data sets before you rerun this option.

You are done when all the jobs are generated. You may then move ahead to viewing the generated jobs. See "Following the generated customization instructions: Security domain" for more information.

Following the generated customization instructions: Security domain

You must generate the customization jobs and files for this task.

The Customization Dialog creates a set of instructions for each customization task. Follow these instructions to tailor and customize a security domain on your system.

Note: Do not attempt to fix a typo or make a change by modifying the generated output. Many of the variables are used in multiple members of the target data sets, so, if you do not change them all, you will run into problems that are very difficult to diagnose.

1. On the 'Configure Security Domain' panel, type ″5″ in the *Option* field to select ″View instructions″ and press **Enter**. ISPF Browse will open and you will see the BBOSDINS member of config_hlq.CNTL.
2. Read the instructions carefully, both to preview the customization process and to find any typographical or other errors you may have made while entering the customization variable values.
3. Follow the instructions as given. There are two ways to follow the directions:
   - Follow the instructions while remaining in ISPF Browse.

- Record the data set name and member at the top of the screen and either print the instructions or use ISPF split screen and browse or edit the instructions while you follow them.

4. Fix any problems. If you encounter problems caused by your Customization Dialog values, modify your variables using the Dialog, regenerate the instructions, and restart the customization process.

   Note: Remember that you cannot generate new customization jobs while either configuration data set is open!

You are done with this customization task when you have successfully followed the generated instructions.

Sample generated instructions: Security domain

This article presents a sample of what the Customization Dialog's generated instructions may look like. This is a sample only--you must use the instructions generated from your own variables when configuring your system.

```
-----------------------------------------------
Instructions for customizing a WebSphere for z/OS security domain.

The customization dialog has created jobs based on the information you
provided. These instructions tell you how to modify the operating
system and run the jobs to customize WebSphere for z/OS.

RULES:

1.  If you created the target data sets (*.CNTL and *.DATA) on another
    (driving) system, you must copy them to the target system and give
    them the same data set names.

2.  You must perform these instructions on your target system.

3.  You will have saved the security domain definition values
    in a data set. These values will need to be loaded and used
    when creating a stand-alone application server or a Network
    Deployment environment.

    ------------------------------------------------------------------

Running the customized jobs
---------------------------

The customization dialog built a number of batch jobs with the
variables you supplied. You must run the jobs in the order listed
below using user IDs with the appropriate authority.

The customization dialog for WebSphere for z/OS does not attempt to
update configuration data for your base operating system or existing
subsystems.

BEFORE YOU BEGIN: You must copy the target data sets (*.CNTL and
*.DATA) to your target system and give them the same data set names,
and you must be running on your target system.

Follow the table below, which lists in order the jobs you must submit
and the commands you must enter. Special handling notes are included
in the table. All jobs are members of

DATASET.CNTL.

Attention: After submitting each job, carefully check the output.
Errors may exist even when all return codes are zero.
```

BBOSBRAJ	User ID requirement: Authority to update data set
Done: By:	DATASET.DATA. This job builds (but does not execute) the RACF commands for the WebSphere for z/OS security domain and places them into member BBOSBRAK of data set DATASET.DATA. Carefully review these definitions with your security administrator.
BBOSBRAK	**User ID requirement: RACF special authority.**
Done: By:	This job instantiates the security rules set up in the previous job by invoking RACF commands. RESULT: You may receive errors, such as INVALID USER messages, from this job because a user ID, group or profile is already defined. Make sure the existing user ID, group or profile has the same characteristics as the user ID, group or profile being created by BBOSBRAK. If not, then change the values in the customization dialog which are causing the conflict, regenerate the customization jobs, and restart the process.
---	Activating the APPL class (optional)
Done: By:	The following APPL profile is created by BBOSBRAK with UACC(READ) and permitted to the unauthenticated user group WSCLGP: CB390 However, in order to make use of APPL profile security with WebSphere for z/OS, you must also activate the RACF APPL class globally with the command: SETROPTS CLASSACT(APPL) Consult with your site security administrator before issuing this command, as it affects all RACF-controlled applications that check the APPL class. See the WAS InfoCenter for more information on implementing security using APPL class profiles.

Creating a stand-alone application server cell

This article leads you through the tasks involved in setting up a WebSphere Application Server for z/OS stand-alone application server environment.

Ensure the security domain was successfully created on the z/OS target system for the new stand-alone application server. Have available a copy of the worksheet that you completed as a part of planning for a stand-alone application cell. (See the *Installing your application serving environment* PDF for more details.)

Follow the steps below to set up a new WebSphere Application Server for z/OS stand-alone application server cell.

Note: You need to complete these steps and have a stand-alone application server cell in place before you migrate your system. See the *Migrating, coexisting, and interoperating* PDF for further information on migration.

1. Log on to TSO on the z/OS system on which you intend to configure the stand-alone application server cell. Use a user ID that has READ access to the WebSphere Application Server for z/OS product data sets. You will also need access to a user ID with authority to make security system updates and a user ID with UID 0. (These can all be the same user ID.)
2. Start the Customization Dialog. See "Starting the Customization Dialog" on page 31 for details.
3. Choose the configuration data sets in which you will store your customization jobs and data. See "Choosing configuration data sets" on page 46 for details.
4. Load the security domain variables saved from the security domain you intend to use for this cell. See "Loading the security domain variables" for details.
5. Set the customization variables according to the values recorded on your stand-alone application server worksheet. See "Setting the customization variables: Stand-alone application server cell" on page 47 for details.
6. (Optional but recommended.) Save the stand-alone application server customization variables in a data set. See "Saving the cell variables" on page 73 for details.
7. Create the customization jobs and files, based on the customization variable values you entered. See "Creating the customization jobs and files" on page 52 for details.
8. Follow the generated customization instructions. See "Following the generated customization instructions: Stand-alone application server cell" on page 54 for details, and a sample set of customization instructions.

You are done when you have successfully completed the steps in the generated instructions. The new stand-alone application server is up and running on the chosen z/OS system. See "Working with your new server" on page 64 for more information.

Deploy and test applications on your new stand-alone application server.

Loading the security domain variables

This article describes how to complete the "Load security domain variables" option for a WebSphere Application Server for z/OS stand-alone application server cell.

Create the security domain you will use for the new stand-alone application server node and know the name of the saved security domain configuration variable file that you recorded on the security domain worksheet.

The security domain settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the security domain variables at the start of node or cell creation, you ensure that the security domain configuration you use is consistent and matches the RACF definitions that have already been set as part of security domain configuration.

Complete this task as the first step in configuring a new stand-alone application server node. If you encounter problems during customization and change the security domain variable values, be sure to re-save them.

1. On the 'Create a stand-alone application server node' panel, type "1" in the *Option* field to select "Load security domain variables" and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Load Security Domain Variables

 Specify the name of a data set containing the security domain variables,
 then press Enter to continue.

 IBM-supplied defaults are in ''


 Data set name:


 If this data set is not cataloged, specify the volume.

 Volume:
```

2. Fill in the name of the sequential data set you used to hold the security domain variable values and press **Enter**.

   Note: Ensure you enclose the data set name in single quotes.
   The security domain variables will load.

The security domain settings are loaded. You can display these variables, but not change them.

Choosing configuration data sets

This article leads you through the "Allocate target data sets" option in the Customization Dialog.

You must start the Customization Dialog and select the "Create stand-alone application server nodes" option.

Each option in the Customization Dialog saves customization jobs and files in a pair of customization data sets. While is it possible to reuse these data sets, it is safest to create separate data sets for each WebSphere Application Server for z/OS configuration. We recommend that you use the customization data set name prefix (sometimes referred to as "config_hlq") to indicate the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring.

For example, you might use the following data set name prefix for configuring a WebSphere Application Server for z/OS 6.0.1 stand-alone application server with cellname SYSA and servername BB6QA1:

```
SYSPROG.WAS601.SYSA.BB6QA1.SAPPSRVR
```

Complete this task before generating the customization jobs and files.

1. On the main Dialog panel, type "2" in the *Option* field to select "Allocate target data sets".
2. Press Enter. **Result:** You see a panel that looks similar to the following:

```
-----    WebSphere Application Server for z/OS Customization   -----
Option  ===>

 Allocate Target Data Sets

 Specify a high level qualifier (HLQ) and press Enter to allocate the
 data sets to contain the generated jobs and instructions. You can
 specify multiple qualifiers (up to 39 characters).

 High level qualifier:                                        .CNTL
                                                              .DATA

 The Dialog will display data set allocation panels. You can make
 changes to the default allocations, however you should not change
 the DCB characteristics of the data sets.

    .CNTL  - a PDS with fixed block 80-byte records to
             contain customization jobs.

    .DATA  - a PDS with variable length data to contain
             other data produced by the Customization Dialog.
```

3. Fill in your chosen configuration data set name prefix value (config_hlq). If the data sets ″config_hlq.CNTL″ and ″config_hlq.DATA″ do not exist, you will be prompted for data set allocation information. If the data sets already exist, a message will inform you that they will be reused.

The data sets ″config_hlq.CNTL″ and ″config_hlq.DATA″ are allocated and will store customization jobs and files. These data set names will also be saved along with the customization variables.

Setting the customization variables: Stand-alone application server cell

This article describes how to complete the ″Define variables″ option for a WebSphere Application Server for z/OS stand-alone application server node.

You must start the Customization Dialog and select the 'Create stand-alone application server nodes' option. Have the `Customization Dialog worksheet: Stand-alone application server cell` completed and at hand. (See the *Installing your application serving environment* PDF for more details.)

1. On the 'Create stand-alone application server nodes' panel, type ″3″ in the *Option* field to select ″Define variables″ and press **Enter**.
2. On the 'Define Variables to Configure stand-alone application server Node' panel, type ″1″ in the *Option* field to select ″System Locations (directories, HLQs, etc.)″ and press **Enter**.
3. Fill in the System Locations panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

   System Locations

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (1 of 2)

   Specify the following for the system on which you are installing
   WebSphere Application Server for z/OS, then press Enter to continue.
   For some data sets, specify "Y" if they are in STEPLIB.

   System name.:  AQTS      Sysplex name :  MCLXCF01

 Full Names of Data Sets
```

```
   PROCLIB:  SYS1.PROCLIB
   PARMLIB:  SYS1.PARMLIB
   SYSEXEC:

   Run WebSphere Application Server from STEPLIB (Y/N)?  Y
   SBBOLPA.:  BOSS.VICOM.W000170.SBBOLPA
   SBBOLOAD:  BOSS.VICOM.W000170.SBBOLOAD
   SBBOLD2.:  BOSS.VICOM.W000170.SBBOLD2
   SBBOEXEC:  BOSS.VICOM.W000170.SBBOEXEC
   SBBOMSG.:  BOSS.VICOM.W000170.SBBOMSG

                                                      Use STEPLIB?
   SCEERUN.:  CEE.SCEERUN                                   N
   SCEERUN2:  CEE.SCEERUN2                                  N
   SGSKLOAD:  GSK.SGSKLOAD                                  N
              (leave SGSKLOAD blank if all systems are at z/OS 1.6 or above)
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (2 of 2)

   Specify the following for your customization, then press Enter
   to continue.

 Locations of HFS Resident Components

   WebSphere Application Server product directory:
     /usr/lpp/zWebSphere/V6R0
```

4. On the 'Define Variables to Configure stand-alone application server Node' panel, type "2" in the *Option* field to select "System Environment Customization" and press **Enter**.
5. Fill in the System Environment Customization panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

   System Environment Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (1 of 4)

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /WebSphere/V6R0
   Name...........:  OMVS.WAS.CONFIG.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  250
   Secondary allocation in cylinders.:  100
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (2 of 4)

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Error Log Stream Information

        Name.................:  WAS.ERROR.LOG
        Data class ..........:  STANDARD
        Storage class........:
        HLQ for data sets....:  IXGLOGR
```

```
   Is log stream CF resident (Y|N):  Y

     If yes, specify structure name.:  WAS_STRUCT
     If no, specify: log stream size:  3000
                     staging size...:  3000

 RRS Log Stream Information

        Group name...........:  MCLXCF01
        Data class...........:  STANDARD
        Storage class........:
        HLQ for data sets....:  IXGLOGR

   Is log stream CF resident (Y|N):  Y

 Create RRS PROC (Y|N)........:  Y
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (3 of 4)

   Specify the following to customize your system environment, then
   press Enter to continue.

 CTRACE Writer Definitions

   Procedure name:  BBOWTR
   User ID.......:  STCRACF
   Group.........:  SYS1

   Trace Data Set Information

     Name...................:  SYS1.AQTS.WAS390.CTRACE
     Volume, or "*" for SMS.:  *
     Primary space in cylinders...:  10
     Secondary space in cylinders.:  0

   Trace Parmlib member suffix...:  60
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (4 of 4)

   Specify the following to customize your server, then press Enter
   to continue.

 Logging Details for Transaction XA Partner Log

 Use Log stream (Y|N):  N

   Log Stream Information

     Name HLQ.....................:  WASTXA
     Data class ..................:
     Storage class................:
     HLQ for data sets............:  IXGLOGR

     Is log stream CF resident (Y|N):  Y

        If yes, specify structure name.:  WAS_STRUCT
        If no, specify: log stream size:  256
                        staging size...:  256
```

6. On the 'Define Variables to Configure stand-alone application server Node' panel, type "3" in the *Option* field to select "Server Customization" and press **Enter**.
7. Fill in the Server Customization panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

   Server Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (1 of 6)

   Specify the following to customize your server, then press Enter
   to continue.

 application server Definitions

   WebSphere Application Server home directory:
     /WebSphere/V6R0
         / AppServer

   Cell name (short)......:  AQTS
   Cell name (long).......:  AQTS

   Node name (short)......:  AQTS
   Node name (long).......:  AQTS

   Server name (short)....:  BBOS001
   Server name (long).....:  server1

   Cluster transition name:  BBOC001

   Admin asynch operations procedure name:  BBOW6SH

   Install samples?  (Y/N):  Y
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (2 of 6)

   Specify the following to customize your server, then press Enter
   to continue.

 application server Definitions

   Controller Information

     Jobname.......:  BBOS001
     Procedure name:  BBO6ACR
     User ID.......:  ASCR1
     UID...........:  2431

   Servant Information

     Jobname.......:  BBOS001S
     Procedure name:  BBO6ASR
     User ID.......:  ASSR1
     UID...........:  2432

   Control Region Adjunct

     Jobname.......:  BBOS001A
     Procedure name:  BBO6CRA
     User ID.......:  ASCRA1
     UID...........:  2433
```

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (3 of 6)

   Specify the following to customize your server, then press Enter
   to continue.

 application server Definitions

   Node host name..........:

     SOAP JMX Connector port.............:  8880

   ORB Listener host name..:  *

     ORB port............................:  2809
     ORB SSL port........................:  0

   HTTP transport host name:  *

     HTTP port...........................:  9080
     HTTP SSL port.......................:  9443

   Service Integration port.........................:  7276
   Service Integration Secure port..................:  7286
   Service Integration MQ Interoperability port.....:  5558
   Service Integration MQ Interoperability Secure port:  5578
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (4 of 6)

   Specify the following to customize your server, then press Enter
   to continue.

 application server Definitions

 Specify your High Availability Manager Host here.  This MUST
 resolve to a single IP address; it can not be a multihomed host

 High Availability Manager Host:

 High Availability Manager Communication Port:  9353
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (5 of 6)

   Specify the following to customize your server, then press Enter
   to continue.

 Location Service Daemon Definitions

   Daemon home directory:
     /WebSphere/V6R0/Daemon

   Daemon jobname:  BBODMNB

   Procedure name.:  BBO6DMN
   User ID........:  WSDMNCR1
   UID............:  2411

   IP name........:  sdf
```

```
   Port...........:  5655
   SSL port.......:  5656

   Register daemon with WLM DNS:  N
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (6 of 6)

   (Note:  This panel is optional if you are not configuring a
    database for the Scheduler component)

   Specify the following for the system on which you wish to
   configure your Scheduler database, then press Enter to continue.

 Full Names of Datasets

   SBPXEXEC...........:  SYS1.SBPXEXEC
   DB2 RUNLIB Location:  DB2HLQ.RUNLIB.LOAD

 Scheduler Database Definitions

   DB2 Subsystem Name.:  DSN
   Plan Name..........:  DSNTIA81

   Scheduler Database Name:  SCHEDDB

   Storage Group Name.....:  SYSDEFLT
   Tablespace Name........:  SCHEDTS
   Table Prefix...........:  TBLPRFIX
```

8. On the 'Define Variables to Configure stand-alone application server Node' panel, type "4" in the *Option* field to select "View Security Domain Configuration Panels" and press **Enter**. These panels display values you previously set in the "Configure security domain" option--you can not change any of the values here. If you do wish to make changes, you must go back to the main Dialog panel and run through the "Configure security domain" option again.

Creating the customization jobs and files

You must select configuration data sets to use and complete the process of defining variables for this task. See "Choosing configuration data sets" on page 46 and "Setting the customization variables: Stand-alone application server cell" on page 47 for more information.

The Customization Dialog creates customization batch jobs and data files, based on the variable values you specified in the Dialog. The batch jobs and data sets will be written to the config_hlq.CNTL and config_hlq.DATA configuration data sets that you created with the 'Allocate target data sets' option.

1. Ensure the configuration data sets are allocated and not in use.

   Note: Editing a member in config_hlq.CNTL or config_hlq.DATA will cause this task to fail.

2. On the 'Create a stand-alone application server node' panel, type "4" in the *Option* field to select "Generate customization jobs" and press **Enter**. You will have one of two results:
   - **Result A:** If all variables are defined correctly, you see the 'Specify Job Cards' panel, which looks similar to this:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Generate Customization Jobs

 This portion of the Customization Dialog generates the jobs you must
 run after you complete this Dialog process. You must complete the
 customization process before you generate the jobs with this step.
 If you have not done this, please return to that step.

 Jobs and data files will get generated into data sets:
   'hlq.CNTL'
   'hlq.DATA'
 If you wish to generate customization jobs using other data sets, then
 exit from this panel and select option 1 (Allocate target data sets).

 All the jobs that will be tailored for you will need a job card.
 Please enter a valid job card for your installation below. The
 file tailoring process will update the jobname for you in all the
 generated jobs, so you need not be concerned with that portion of
 the job cards below. If continuations are needed, replace the
 comment cards with continuations.

 Specify the job cards, then press Enter to continue.

 //jobname  JOB (ACCTNO,ROOM),'userid',CLASS=A,REGION=0M
 //*
 //*
 //*
```

Note: Pay particular attention to the displayed target data sets. Make sure they are the ones you intend to use.

- **Result B:** If the variables are not defined correctly, you will see the 'Verification' panel. Decide whether the warnings or errors are serious enough to warrant returning to the "Define variables" option.

  Note: If the return code is "8" or greater, return to the "Define variables" option and fix the uncovered problems. If you saved the variables previously, be sure to re-save them after making any updates.

3. Fill in the job card information, according to your installation requirements. For each job, the Dialog generates a jobname and the "JOB" keyword to match the member name of the PDS, but you specify the rest.

   Note: If you need to run these jobs on a particular system in the sysplex (for example, JES2 MAS or JES3 complex), you should specify the necessary Scheduling Environment (SCHENV), JES2 JOBPARM, or JES3 //*MAIN statement at this time.

   Example of a job card entry:

```
//jobname JOB 1234,USER1,NOTIFY=????,MSGCLASS=O,REGION=0M
//*             USER=SYSADM1,PASSWORD=SYSADM1
/*JOBPARM SYSAFF=SYSB
```

   Note: This example is useful for jobs that require a user ID other than that of the logged-on TSO user. (This is typically a user ID with UID=0.) In that case, you can just put a comma at the end of the first line, put in the correct user ID on the second line, then uncomment that second line.

   You may wish to use RACF SUBMIT authority to avoid having to keep passwords in your configuration data sets.

4. Fix any errors. If there are errors anywhere, you will see the 'Error' panel. Press PF3 to exit the error panel, then enter the correct panel to fix the errors. Then

return to the ″Generate Customization Jobs″ option and pick up where you left off. If necessary, you can update the variables and rerun this option. The generation process will delete and re-tailor all the members.

Note: Compress the configuration data sets before you rerun this option.

You are done when all the jobs are generated. You may then move ahead to viewing the generated jobs. See “Following the generated customization instructions: Stand-alone application server cell” for more information.

Following the generated customization instructions: Stand-alone application server cell

You must generate the customization jobs and files for this task.

The Customization Dialog creates a set of instructions for each customization task. Follow these instructions to tailor and customize a stand-alone application server node on your system.

Note: Do not attempt to fix a typo or make a change by modifying the generated output. Many of the variables are used in multiple members of the target data sets, so, if you do not change them all, you will run into problems that are very difficult to diagnose.

1. On the 'Create a stand-alone application server node' panel, type ″5″ in the *Option* field to select ″View instructions″ and press **Enter**. ISPF Browse will open and you will see the BBOSSINS member of config_hlq.CNTL.
2. Read the instructions carefully, both to preview the customization process and to find any typographical or other errors you may have made while entering the customization variable values.
3. Follow the instructions as given. There are two ways to follow the directions:
   - Follow the instructions while remaining in ISPF Browse.
   - Record the data set name and member at the top of the screen and either print the instructions or use ISPF split screen and browse or edit the instructions while you follow them.
4. Fix any problems. If you encounter problems caused by your Customization Dialog values, modify your variables using the Dialog, regenerate the instructions, and restart the customization process.

   Note: Remember that you cannot generate new customization jobs while either configuration data set is open!

You are done with this customization task when you have successfully followed the generated instructions.

Sample generated instructions: Stand-alone application server cell

This article presents a sample of what the Customization Dialog's generated instructions may look like. This is a sample only--you must use the instructions generated from your own variables when configuring your system.

```
-----------------------------------------------
Instructions for customizing WebSphere for z/OS
for a stand-alone application server node.
```

```
The customization dialog has created jobs based on the information you
provided. These instructions tell you how to modify the operating
system and run the jobs to customize WebSphere for z/OS.

RULES:

1.  If you created the target data sets (*.CNTL and *.DATA) on another
    (driving) system, you must copy them to the target system and give
    them the same data set names.

2.  You must perform these instructions on your target system.

Doing manual configuration updates
----------------------------------

The customization dialog for WebSphere for z/OS does not attempt to
update configuration data for your base operating system or existing
subsystems. You must do the following manual steps prior to running
the WebSphere for z/OS configuration jobs.

Perform these steps to do manual configuration updates:

1.  Update the workload management application environment as follows.

    ATTENTION: If you have already installed the WLM-DAE support PTF
    (APAR OW54622) on z/OS 1.4, or are running a higher level of
    z/OS, you may skip this step.

    Using these parameters, run IWMARIN0 to create the BBOC001
    application environment:

    Appl Environment Name    BBOC001
    Description              WebSphere for z/OS V6 servant
    Subsystem type           CB
    Procedure name           BBO6ASR
    Start parameters         JOBNAME=&IWMSSNM.S;,
                               ENV=AQTS.AQTS.&IWMSSNM;
    Limit on starting server address spaces for a subsystem instance:
         Single address space per system

    NOTE: For the start parameter string, you must continue typing with
    ENV on the same line as JOBNAME. We show the parameters on separate
    lines for aesthetics only.

    For information about IWMARIN0, the ISPF dialog application for MVS
    workload management, see z/OS MVS Planning: Workload Management
    (SA22-7602).

    For more information about workload management and WebSphere for
    z/OS, see related topics in the WebSphere for z/OS Information
    Center at
    http://www.ibm.com/software/webservers/appserv/zos_os390/library/.

    ---------------------------------------------------------------------

2.  Update BLSCUSER. Refer to member BBOIPCSP in

    'DATASET.CNTL'

    In order to use the IPCS support provided by the product, append
    the contents of this member to the BLSCUSER member in your IPCSPARM
    or system PARMLIB datasets.

    ---------------------------------------------------------------------

3.  Update SCHEDxx. Refer to member BBOSCHED in
```

'DATASET.CNTL'

In order to set the correct program properties for the WebSphere for z/OS run-time executables, append the contents of this member to the SCHEDxx member in your system PARMLIB concatenation.

Note: When you are finished, issue the command SET SCH=(xx,xx) to activate SCHEDxx and load a new program properties table.

---

4. Make sure the following data sets are APF-authorized:

    BOSS.VICOM.W000170.SBBOLPA
    BOSS.VICOM.W000170.SBBOLOAD
    BOSS.VICOM.W000170.SBBOLD2
    CEE.SCEERUN
    CEE.SCEERUN2
    GSK.SGSKLOAD

   Add these datasets to your PROGxx or IEAAPFxx parmlib members, as appropriate, ensuring you specify the correct volsers.

---

5. If you want to collect the SMF120 records created by the run-time servers, update SMFPRMxx via the following:

   a. Update the SYS or SUBSYS(STC,...) statement for started tasks to include the 120 record.
   b. (Optional) You can specify designated subtypes 1-6.

   EXAMPLE:

```
   SUBSYS(STC,EXITS(IEFU29,IEFACTRT),INTERVAL(SMF,SYNC),
                      TYPE(0,30,70:79,88,89,120,245))
                                            ---
```

   For details on the SMF records, see related topics in the WebSphere for z/OS Information Center at http://www.ibm.com/software/webservers/appserv/zos_os390/library/

---

6. Update your active BPXPRMxx member to have the following WebSphere for z/OS configuration HFS:

   OMVS.WAS.CONFIG.HFS

   mounted at:

   /WebSphere/V6R0

   in read/write mode.

   EXAMPLE:

```
   MOUNT FILESYSTEM('OMVS.WAS.CONFIG.HFS')
     MOUNTPOINT('/WebSphere/V6R0')
      TYPE(HFS)
      MODE(RDWR)
```

---

7. Update TCP/IP by reserving the following ports for WebSphere for z/OS:

```
   SOAP JMX Connector port                              - 8880
   ORB port                                             - 2809
   HTTP port                                            - 9080
   HTTP SSL port                                        - 9443

   High availability manager communication port         - 9353
   Service Integration port                             - 7276
   Service Integration Secure port                      - 7286
   Service Integration MQ Interoperability port         - 5558
   Service Integration MQ Interoperability Secure port - 5578

   Daemon IP port                                       - 5655
   Daemon SSL port                                      - 5656

View member BBOTCPIP in

'DATASET.CNTL'

Add the contents of this member to the PORT section of the file
referenced by the DD statement for the TCP/IP profile in the
TCP/IP start procedure. Cut and paste from this member into the
data set used by your installation.

ATTENTION: If another application has already reserved any of these
ports for its own use, you must resolve the resulting conflict
before you continue. If you update the WebSphere for z/OS
customization dialog with new port specifications, be sure to
regenerate the customization jobs, data, and instructions.

-----------------------------------------------------------------
```

8. The WebSphere product libraries will be placed in STEPLIB as needed, rather than in the system link pack area and system link list.

```
SBBOLPA:
=========
The following data set will be placed in the STEPLIB concatentation
for the location service daemon.  The data set's load modules will
be loaded in the MVS common storage area when the location service
daemon is started.  Make sure the target MVS system has at least
8MB free storage in extended CSA before starting the daemon.

BOSS.VICOM.W000170.SBBOLPA


SBBOLOAD and SBBOLD2:
====================
The following data sets will be placed in the STEPLIB concatenation
for the location service daemon, controller and servant regions,
and in the setupCmdLine.sh script in the WebSphere Configuration
HFS. You must not remove these STEPLIB statements.

BOSS.VICOM.W000170.SBBOLOAD
BOSS.VICOM.W000170.SBBOLD2

BBORTSS5:
=========

The BBORTSS5 module is used by WebSphere Application Server V6
for component trace support. A copy of this module must be in
the system link pack area in order for CTRACE to work correctly.

If a copy of BBORTSS5 (any release) is currently loaded into LPA,
you need take no further action.
```

Otherwise, issue the following MVS console command to load BBORTSS5 into dynamic LPA:

```
SETPROG LPA,ADD,MODNAME=BBORTSS5,
           DSNAME=BOSS.VICOM.W000170.SBBOLPA
```

Alternatively, you can place the following statement in a parmlib PROGxx member which is activated with the SET PROG= command after system IPL is complete:

```
LPA ADD MODNAME(BBORTSS5) DSNAME(BOSS.VICOM.W000170.SBBOLPA)
```

Make sure that the BBORTSS5 module is loaded into LPA after each system IPL.

---

9. Make sure the following Language Environment data set is in the system link list:

   CEE.SCEERUN

   ---

10. Make sure the following Language Environment data set is in the system link list:

    CEE.SCEERUN2

    ---

11. Make sure the following System SSL data set is in the system link pack area or the system link list on all MVS systems at or below z/OS Version 1.5:

    GSK.SGSKLOAD

    ---

12. WebSphere for z/OS regions open a large number of files (more than 1024). Make sure your BPXPRMxx parmlib member(s) specify a value of MAXFILEPROC that is greater than or equal to 2000. Use the following MVS console command to see your current MAXFILEPROC setting:

    ```
    D OMVS,OPTIONS
    ```

    ---

13. Update the CFRM Policy. Prior to using log streams that have been indicated as CF-resident, you must update the CFRM policy to define the structures to be used. Tailor member BBOWCFRM in the following data set to define the log streams:

    'DATASET.CNTL'

Running the customized jobs

The customization dialog built a number of batch jobs with the variables you supplied. You must run the jobs in the order listed below using user IDs with the appropriate authority.

BEFORE YOU BEGIN: Complete the section above entitled "Doing manual configuration updates."

Follow the table below, which lists in order the jobs you must submit and the commands you must enter. Special handling notes are included in the table. All jobs are members of

DATASET.CNTL

Attention: After submitting each job, carefully check the output. Errors may exist even when all return codes are zero.

BBOMSGC	User ID requirement: Update authority for data set SYS1.MSGENU and/or SYS1.MSGJPN. Done: ATTENTION: This is optional unless you require message translation. By: This job sets up MMS to translate messages for WebSphere for z/OS. There are two steps to update SYS1.MSGENU and SYS1.MSGJPN. Remove the unneeded step and change the target libraries, if necessary.
BBOERRLG	User ID requirement: Authority to define a log stream. Done: ATTENTION: If your installation already has a WebSphere for z/OS error log stream named the following, skip this step. WAS.ERROR.LOG This job defines the error log stream. By: The customization dialog supplied the required parameters on the define log stream command. Review and supply any options you require. For more information, see z/OS MVS Setting Up a Sysplex (SA22-7625). RESULT: Upon successful completion of this job, you will see the following message in SYSPRINT: IXG004I LOGR POLICY PROCESSING ENDED WITHOUT ERROR
BBORRSLS	User ID requirement: Authority to define a log stream. Done: ATTENTION: If your installation already has Resource Recovery Services (RRS) active, skip this job. (To check if RRS is active, go to SDSF or the operator console and look for an address space named "ATRRRS"). By: This job defines the RRS log streams. The RRS group name is, by default, the resource sharing group name. RESULT: Upon successful completion of this job, you will see the following message in SYSPRINT: IXG004I LOGR POLICY PROCESSING ENDED WITHOUT ERROR
BBOWCTR	User ID requirement: Authority to allocate data set

Done:	SYS1.AQTS.WAS390.CTRACE
	This job allocates the CTRACE data set used by
By:	BBOWTR
	ATTENTION: Skip this step if the CTRACE data set is already allocated.
BBOCBRAJ	User ID requirement: Authority to update data set
Done:	DATASET.DATA
By:	This job builds (but does not execute) the RACF commands for the WebSphere for z/OS run-time clusters and places them into member BBOWBRAK of data set
	DATASET.DATA
	Carefully review these definitions with your security administrator.
BBOCBRAK	User ID requirement: RACF special authority.
Done:	This job executes the RACF commands set up in the previous job.
By:	RESULT: You may receive errors, such as INVALID USER messages, from this job because a user ID, group or profile is already defined. Make sure the existing user ID, group or profile has the same characteristics as the user ID, group or profile being created by BBOCBRAK. If not, then change the values in the customization dialog which are causing the conflict, regenerate the customization jobs, and restart the process.
--------	Check user ID authorizations.
Done:	Make sure the WSCFG1 group has read access to all WebSphere product data sets, as well as to CEE.SCEERUN
By:	Make sure the following user IDs have read access to the resolver configuration file in use on your system. Depending on your IP setup, this file may be /etc/resolv.conf, SYS1.TCPPARMS(TCPDATA), or another data set. ASCR1 ASSR1 See the z/OS eNetwork Communication Server IP Configuration manual for the resolver search order.
	Ensure the following user ID has read access to the data sets in your system parmlib concatenation: WSDMNCR1
	ATTENTION: If operator commands are protected by the z/OS security server at your installation, you must ensure that sufficient authority is given to WebSphere tasks to

	control operations. The application server Controller user ID (ASCR1), the application server Servant user id (ASSR1), and the Service Daemon user ID (WSDMNCR1) need the ability to perform operations on started tasks belonging to WebSphere Application Server for z/OS in the same cell (or security domain). The RACF commands to authorize these user IDs are not generated by the previous job (BBOCBRAJ). The following example commands show what needs to be done, substituting your profile names: PERMIT START_profile_name CLASS(OPERCMDS) ID (ASCR1 ASSR1 WSDMNCR1) ACCESS(UPDATE) PERMIT STOP_profile_name CLASS(OPERCMDS) ID (ASCR1 ASSR1 WSDMNCR1) ACCESS(UPDATE) PERMIT MODIFY_profile_name CLASS(OPERCMDS) ID (ASCR1 ASSR1 WSDMNCR1) ACCESS(UPDATE) PERMIT CANCEL_profile_name CLASS(OPERCMDS) ID (ASCR1 ASSR1 WSDMNCR1) ACCESS(UPDATE) PERMIT FORCE_profile_name CLASS(OPERCMDS) ID (ASCR1 ASSR1 WSDMNCR1) ACCESS(UPDATE)
BBOWCHFS	User ID requirement: UID=0 and authority to allocate
Done: By:	OMVS.WAS.CONFIG.HFS This job: o Creates a mount point directory /WebSphere/V6R0 o Allocates the configuration HFS OMVS.WAS.CONFIG.HFS and mounts it at the above mount point. BEFORE YOU BEGIN: The BBOWCHFS job assumes your root HFS is mounted in read/write mode. If the root HFS is not in read/write mode, manually create the directory /WebSphere/V6R0 and any needed higher directories, set file permissions to 775, and set the owning user ID and group to WSADMIN and WSCFG1 before running BBOWCHFS. EXAMPLE: If you plan to use /WebSphere/V6R0 as your directory, issue the following commands from within the OMVS shell: mkdir -p -m 775 /WebSphere/V6R0 chown -R WSADMIN:WSCFG1 /WebSphere

BBOWHFSA	User ID requirement: UID=0.
Done: By:	This job populates the previously-created HFS. Upon completion, examine the job output. Success is indicated with a RC=0 in the job output.
BBOWCPY1	User ID requirement: update authority for:
Done: By:	SYS1.PROCLIB SYS1.PARMLIB ATTENTION: This job modifies SYS1.PROCLIB. Because master subsystem address spaces like ATRRRS and BBOWTR must have their cataloged procedures in a PROCLIB listed in the master scheduler JCL, we copy members to SYS1.PROCLIB. You may have a private master subsystem PROCLIB to which you want to copy the members. This job copies the tailored start procedures, parameters, and EXECs to the run-time libraries. ATTENTION: Be aware that you may overlay existing members in the above data sets.
BBOWWPFA	User ID requirement: UID=0.
Done: By:	This job sets up the runtime HFS. Upon completion, examine the job output. Success is indicated by rc=0.
BBOWHFSB	User ID requirement: UID=0.
Done: By:	This job will complete the HFS initialization. Upon completion, examine the job output. Success is indicated with a RC=0 in the job output.
--------	If Resource Recovery Services (RRS) is not active, issue the following MVS command. (To check if RRS is active, go to SDSF or the operator console and see if "RRS" is listed as active).
Done: By:	 START ATRRRS,SUB=MSTR Note: You know you were successful if the following appears in the SYSLOG: ASA2011I RRS INITIALIZATION COMPLETE. COMPONENT ID=SCRRS
--------	Issue the MVS command:
Done:	TRACE CT,WTRSTART=BBOWTR This command starts the CTRACE writer used by WebSphere

By:	for z/OS. RESULT: Check the SYSLOG for the following: ITT110T INITIALIZATION OF CTRACE WRITER COMPLETE.
--------	Start the application server
Done: By:	Issue the MVS command START BBO6ACR,JOBNAME=BBOS001, ENV=AQTS.AQTS.BBOS001 This command starts the application server. Wait until the server is finished initializing before proceeding. RESULT: The following message appears on the console and in the job log of BBOS001 BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR z/OS CONTROL PROCESS BBOS001
BBOWIVT	User ID requirement: UID=0 or WSADMIN
Done: By:	This job runs the IVT application. See related topics in the WebSphere for z/OS Information Center at http://www.ibm.com/software/webservers/appserv/zos_os390/library/ for information about how to run this job.

The product is now configured and verified.

To start the application server, issue the MVS command:

```
START BBO6ACR,JOBNAME=BBOS001,
      ENV=AQTS.AQTS.BBOS001
```

To stop the application server, enter the MVS command:

```
STOP BBODMNB
```

Use the following jobs to configure a database for the Scheduler component. These are optional if you do not plan to use the Scheduler.

	On the z/OS machine that will host the database:
Done: By:	- Log on to the native z/OS environment. - If multiple DB2 systems are installed, then decide which subsystem you want to use. Note the subsystem. - Using the DB2 administration menu, create a new database (SCHEDDB, for example). Note the database name - Create a storage group (or use an existing one) and note the name - Decide which user ID will connect to the database from the remote machine running the product. Normally, for security reasons, this user ID is not the one you used to create the database. - Grant the user ID rights to access the database and storage group. The user ID must also have permission

	to create new tables for the database.	
BBOCRTTS	User ID requirement: WSADMIN	
Done:	This job creates the tablespaces in the Scheduler database	
By:		
BBOCRTSC	User ID requirement: WSADMIN	
Done:	This job creates the schema for the Scheduler database. This job assumes that BBOCRTTS has already been run to create the necessary tablespaces in the Scheduler database	
By:		
BBODRPTS	User ID requirement: WSADMIN	
Done:	This job drops the previously created Scheduler tablespaces. This job is customized to drop tablespaces which were created with the matching BBOCRTTS job	
By:		
BBODRPSC	User ID requirement: WSADMIN	
Done:	This job drops the previously created Scheduler schema. This job is customized to drop the schema which was created with the matching BBOCRTSC job	
By:		

The following is a useful script that helps you define security controls for clusters. It is in data set

'DATASET.DATA'

BBOWBRAC	This is a sample exec you can modify to include installation-specific RACF controls. This exec defines all the user IDs and groups that are necessary and sufficient for installing WebSphere for z/OS.
Done:	
By:	Additionally, there are commented sections for other components that might be used (SSL, for example).

Working with your new server

Once you complete the customization instructions, you will have a WebSphere Application Server for z/OS stand-alone application server. The application server includes a (simple) cell and node structure. This article provides useful information for when you work with your new server.

Before starting your server

Make sure the WebSphere Application Server for z/OS product HFS and configuration HFS are mounted. If you chose to load the SBBOLPA (and possibly SBBOLOAD) into the system link pack area, make sure these libraries are loaded into LPA before starting the server.

Starting the stand-alone application server

To start your stand-alone application server, issue the following MVS console command:

```
START server_proc,JOBNAME=server_name,ENV=cell_name.node_name.server_name
```

where:

- *server_proc* is the stand-alone application server controller cataloged procedure.
- *server_name* is the server short name.
- *node_name* is the node short name.
- *cell_name* is the cell short name.

For example, if you chose default values and your system is named MVSA, you would enter the following START command:

```
START BBO6ACR,JOBNAME=BBOS001,ENV=MVSA.MVSA.BBOS001
```

The START command brings up the controller. The controller starts the location service daemon, then uses WLM to start the servant. You should see a message like the following when the entire server is up and running:

BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR z/OS CONTROL PROCESS BBOS001

Accessing the server administrative console

Once the server is successfully started, access the administrative console by pointing a Web browser to the following URL:

http://*hostname*:*http_port*/ibm/console

where:

- *hostname* is the HTTP transport host name you specified during customization.

  Note: If you specified '*' for the HTTP host name, this is actually the node host name.
- *http_port* is the HTTP port you specified during customization.

  Note: The default HTTP port for the stand-alone application server is 9080.

Until global security is enabled, you will see a sign-on screen that asks you for a user ID but no password.

The user ID can be anything and is used only to provide basic tracking of changes. Be aware that, until you enable global security, anyone with a Web browser and access to the HTTP port can modify your application serving environment.

You can use the administrative console, scripting, or both to manage the application server and deploy and manage J2EE applications. See the *Administering applications and their environment* PDF for more information.

Accessing the Samples Gallery

If you chose to install the sample applications, you can access them by pointing a Web browser to the following URL:

http://*hostname*:*http_port*/WSsamples/

where:

- *hostname* is the HTTP transport host name you specified during customization.

  Note: If you specified '*' for the HTTP host name, this is actually the node host name.
- *http_port* is the HTTP port that you specified during customization.

  Note: The default HTTP port for the stand-alone application server is 9080.

See "Accessing the Samples (Samples Gallery)" in the product Information Center for more information.

Stopping your stand-alone application server

The easiest way to stop the stand-alone application server is to stop the location service daemon. The location service daemon holds pointers to modules in common storage, and stopping it forces the rest of the cell to shut down. To stop the location service daemon, enter the following MVS console command:

```
STOP daemon_jobname
```

where *daemon_jobname* is the location service daemon jobname. The default location service daemon jobname for a stand-alone application server is BBODMNB.

Using the Installation Verification Test

You initially run the installation verification test (IVT), which verifies that WebSphere Application Server is configured correctly for your system, during ISPF customization of each of your systems. If you want to run the IVT at a time other than during initial customization, however, there are two methods from which you can choose.

Note: These options are available only if the user is running a stand-alone application server configuration.

Select either method to invoke the IVT:

- "Running the Installation Verification Test with a job"
- "Running the Installation Verification Test from a command line" on page 67

Running the Installation Verification Test with a job

The application server must be running.

Follow these steps to run the Installation Verification Test using the BBOWIVT job.

1. Verify that the application server is running.

   Note: The IVT will end unsuccessfully if the server is not running.

2. Confirm that the ivtApp application is installed and started.
3. Submit the job BBOWIVT.

After initialization, the IVT runs its series of verification tests and reports pass or fail status for each in the messages generated by the BBOWIVT job. Once it finishes, check the results in the install_root/logs/ivt.log file.

Running the Installation Verification Test from a command line

The application server must be running.

Follow these steps to run the Installation Verification Test from a command line.

1. Verify that the application server is running.

   Note: The IVT will end unsuccessfully if the application server is not running.
2. Confirm that the ivtApp application is installed and started.
3. From a command line, navigate to the `/WebSphere/V6R0/AppServer/bin` directory.
4. Issue the following command:

   ```
   ivt.sh [-p port_number] [-host host_name]
   ```

   where

   - `-p port_number` is an optional argument that specifies your port number. If you do not specify a port number, the program will use the default port number value of 9080.
   - `-host host_name` is an optional argument that specifies your host name. If you do not specify a host name, the program will use the host name value that is set in your TCP/IP hosts file.

   Example:

   ```
   /WebSphere/V6R0/AppServer/bin> ivt.sh -p 9090 -host myhost
   ```

The IVT will run, producing a series of verification tests and reporting pass or fail status in the messages generated by the BBOWIVT job. It will also log results to the `install_root\logs\ivt.log` file.

Note: After command line initialization, the IVT runs its series of verification tests and reports pass or fail status for each. Once it finishes, check the results in the install_root/logs/ivt.log file.

Creating a Network Deployment cell

This article leads you through the tasks involved in setting up a WebSphere Application Server for z/OS Network Deployment environment.

Ensure the security domain was successfully created on the z/OS target system(s) for the new Network Deployment cell. Have available a copy of the worksheet that you completed as a part of Planning for a Network Deployment cell.

Perform this task to set up a new WebSphere Application Server for z/OS Network Deployment cell. These steps will set up the cell and the deployment manager node.

1. Log on to TSO on the z/OS system on which you intend to configure the Network Deployment cell's deployment manager. Use a user ID that has READ access to the WebSphere Application Server for z/OS product data sets. You

will also need access to a user ID with authority to make security system updates and a user ID with UID 0. (These can all be the same user ID.)

2. Start the Customization Dialog. See “Starting the Customization Dialog” on page 31 for details.
3. Choose the configuration data sets in which you will store your customization jobs and data. See “Choosing configuration data sets” on page 69 for details.
4. Load the security domain variables saved from the security domain you intend to use for this cell. See “Loading the security domain variables” for details.
5. Set the customization variables according to the values recorded on your Network Deployment cell worksheet. See “Setting the customization variables: Network Deployment cell” on page 70 for details.
6. Save the Network Deployment cell customization variables in a data set. See “Saving the cell variables” on page 73 for details. You will use these variables when creating managed nodes for the cell or federating stand-alone application servers into it.
7. Create the customization jobs and files, based on the customization variable values you entered. See “Creating the customization jobs and files” on page 74 for details.
8. Follow the generated customization instructions. See “Following the generated customization instructions: Network Deployment cell” on page 75 for details, and a sample set of customization instructions.

You are done when you have successfully completed the steps in the generated instructions. The new deployment manager is up and running on the chosen z/OS system. See “Working with your new deployment manager” on page 83 for more information.

Add application server nodes to your cell using one of two methods:

- Create a new managed node using the Customization Dialog and add application servers to it using the administrative console or scripting.
- Federate existing stand-alone application server(s) into your Network Deployment cell to create managed nodes with application servers.

Loading the security domain variables

This article describes how to complete the ″Load security domain variables″ option for a WebSphere Application Server for z/OS Network Deployment cell.

Create the security domain you will use for the new Network Deployment cell and know the name of the saved security domain configuration variable file that you recorded on the security domain worksheet.

The security domain settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the security domain variables at the start of node or cell creation, you ensure that the security domain configuration you use is consistent and matches the RACF definitions that have already been set as part of security domain configuration.

Complete this task as the first step in configuring a new Network Deployment cell. If you encounter problems during customization and change the security domain variable values, be sure to re-save them.

1. On the ’Configure Deployment Manager Node’ panel, type ″1″ in the *Option* field to select ″Load security domain variables″ and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Load Security Domain Variables

 Specify the name of a data set containing the security domain variables,
 then press Enter to continue.

 IBM-supplied defaults are in ''


 Data set name:


 If this data set is not cataloged, specify the volume.

 Volume:
```

2. Fill in the name of the sequential data set you used to hold the security domain variable values and press **Enter**.

   Note: Ensure you enclose the data set name in single quotes.
   The security domain variables will load.

The security domain settings are loaded. You can display these variables, but not change them.

Choosing configuration data sets

This article leads you through the ″Allocate target data sets″ option in the Customization Dialog.

You must start the Customization Dialog and select the ″Create a Network Deployment cell″ option.

Each option in the Customization Dialog saves customization jobs and files in a pair of customization data sets. While is it possible to reuse these data sets, it is safest to create separate data sets for each WebSphere Application Server for z/OS configuration. We recommend that you use the customization data set name prefix (sometimes referred to as ″config_hlq″) to indicate the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring.

For example, you might use the following data set name prefix for configuring a WebSphere Application Server for z/OS 6.0.1 Network Deployment cell with cellname AZCELL:

```
SYSPROG.WAS601.AZCELL.NDCONFIG
```

Complete this task before generating the customization jobs and files.

1. On the main Dialog panel, type ″2″ in the *Option* field to select ″Allocate target data sets″.
2. Press Enter. **Result:** You see a panel that looks similar to the following:

```
-------     WebSphere Application Server for z/OS Customization      ---------
Option  ===>

 Allocate Target Data Sets

 Specify a high level qualifier (HLQ) and press Enter to allocate the
 data sets to contain the generated jobs and instructions. You can
```

```
specify multiple qualifiers (up to 39 characters).

High level qualifier:                                          .CNTL
                                                               .DATA

The Dialog will display data set allocation panels. You can make
changes to the default allocations, however you should not change
the DCB characteristics of the data sets.

   .CNTL  - a PDS with fixed block 80-byte records to
            contain customization jobs.

   .DATA  - a PDS with variable length data to contain
            other data produced by the Customization Dialog.
```

3. Fill in your chosen configuration data set name prefix value (config_hlq). If the data sets "config_hlq.CNTL" and "config_hlq.DATA" do not exist, you will be prompted for data set allocation information. If the data sets already exist, a message will inform you that they will be reused.

The data sets "config_hlq.CNTL" and "config_hlq.DATA" are allocated and will store customization jobs and files. These data set names will also be saved along with the customization variables.

Setting the customization variables: Network Deployment cell

This article describes how to complete the "Define variables" option for a WebSphere Application Server for z/OS Network Deployment cell.

You must start the Customization Dialog and select the 'Create Network Deployment cells and nodes' option then the 'Create a Network Deployment cell' option. Have Customization Dialog worksheet: Network Deployment cell completed and at hand.

1. On the 'Configure Deployment Manager Node' panel, type "3" in the *Option* field to select "Define variables" and press **Enter**.
2. On the 'Define Variables to Configure stand-alone application server Node' panel, type "1" in the *Option* field to select "System Locations (directories, HLQs, etc.)" and press **Enter**.
3. Fill in the System Locations panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

   System Locations

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (1 of 2)

   Specify the following for the system on which you are installing
   WebSphere Application Server for z/OS, then press Enter to continue.
   For some data sets, specify "Y" if they are in STEPLIB.

   System name.:  AQTS      Sysplex name :  MCLXCF01

 Full Names of Data Sets

   PROCLIB:  SYS1.PROCLIB
   PARMLIB:  SYS1.PARMLIB
   SYSEXEC:

   Run WebSphere Application Server from STEPLIB (Y/N)?  Y
   SBBOLPA.:  BOSS.VICOM.W000170.SBBOLPA
   SBBOLOAD:  BOSS.VICOM.W000170.SBBOLOAD
   SBBOLD2.:  BOSS.VICOM.W000170.SBBOLD2
```

```
   SBBOEXEC:  BOSS.VICOM.W000170.SBBOEXEC
   SBBOMSG.:  BOSS.VICOM.W000170.SBBOMSG

                                                        Use STEPLIB?
   SCEERUN.:  CEE.SCEERUN                                    N
   SCEERUN2:  CEE.SCEERUN2                                   N
   SGSKLOAD:  GSK.SGSKLOAD                                   N
              (leave SGSKLOAD blank if all systems are at z/OS 1.6 or above)
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (2 of 2)

   Specify the following for your customization, then press Enter
   to continue.

 Locations of HFS Resident Components

   WebSphere Application Server product directory:
     /usr/lpp/zWebSphere/V6R0
```

4. On the 'Define Variables to Configure Deployment Manager Node' panel, type "2" in the *Option* field to select "System Environment Customization" and press **Enter**.
5. Fill in the System Environment Customization panel using the following screen shot as your guide. When you are done, press **Enter**.

   System Environment Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (1 of 1)

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /WebSphere/V6R0
   Name...........:  OMVS.WAS.CONFIG.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  250
   Secondary allocation in cylinders.:  100
```

6. On the 'Define Variables to Configure Deployment Manager Node' panel, type "3" in the *Option* field to select "Server Customization" and press **Enter**.
7. Fill in the Server Customization panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

   Server Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (1 of 4)

   Specify the following to customize your server, then press Enter
   to continue.

 Deployment Manager Definitions

   WebSphere Application Server home directory:
     /WebSphere/V6R0
         / DeploymentManager

   Cell name (short)......:  MCLXCF01
   Cell name (long).......:  MCLXCF01Network
```

```
   Node name (short)......:  MCLXCF01
   Node name (long).......:  MCLXCF01Manager

   Server name (short)....:  BBODMGR
   Server name (long).....:  dmgr

   Cluster transition name:  BBODMGR
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (2 of 4)

   Specify the following to customize your server, then press Enter
   to continue.

 Deployment Manager Definitions

   Controller Information

     Jobname.......:  BBODMGR
     Procedure name:  BBO6DCR
     User ID.......:  DMCR1
     UID...........:  2421

   Servant Information

     Jobname.......:  BBODMGRS
     Procedure name:  BBO6DSR
     User ID.......:  DMSR1
     UID...........:  2422
------------  WebSphere Application Server for z/OS Customization     -------
Option  ===>

Server Customization (3 of 4)

   Specify the following to customize your server, the press Enter
   to continue.

 Deployment Manager Definitions

   Node host name..........:

     SOAP JMX connector port...................:  8879
     Cell Discovery Address port...............:  7277
     DRS CLIENT Address port...................:  7989

   ORB Listener host name..:  *

     ORB port..................................:  9809
     ORB SSL port..............................:  0

   HTTP transport host name:  *

     HTTP port.................................:  9060
     HTTP SSL port.............................:  9043

 The High Availability Manager Host MUST resolve to a single
 IP address.  It can not be a multihomed host.

   High Availability Manager Host:
   High Availability Manager Communication Port:  9352
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (4 of 4)
```

```
  Specify the following to customize your server, then press Enter
  to continue.

Location Service Daemon Definitions

  Daemon home directory:
    /WebSphere/V6R0/Daemon

  Daemon jobname:  BBODMNC

  Procedure name.:  BBO6DMN
  User ID........:  WSDMNCR1
  UID............:  2411

  IP name........:
  Port...........:  5755
  SSL port.......:  5756

  Register daemon with WLM DNS:  N
```

8. On the 'Define Variables to Configure Deployment Manager Node' panel, type "4" in the *Option* field to select "View Security Domain Configuration Panels" and press **Enter**. These panels display values you previously set in the "Configure security domain" option--you can not change any of the values here. If you do wish to make changes, you must go back to the main Dialog panel and run through the "Configure security domain" option again.

Saving the cell variables

You must start the Customization Dialog and fill in the variables for the chosen task.

Saving your Network Deployment cell variables allows you to load the same consistent set of values when configuring a new managed node for the cell.

Complete this task after setting the variables for your chosen task. If you encounter problems during customization and change the variable values, be sure to re-save them.

Note: This procedure applies to all Dialog options except "Configure a security domain." For information about saving those variables, see "Saving the security domain variables" on page 40

1. On the main panel for your chosen task, type "S" in the *Option* field to select "Save customization variables" and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Save Customization Variables

  Specify the name of a sequential data set to contain the
  customization variables, then press Enter to continue. If the
  data set does not exist, the Dialog displays the Allocate New
  Data Set panel, with which you can allocate a data set.


 Data set name:
```

2. Fill in the name of the sequential data set you will use to hold the variable values. Choose a data set name that identifies the sysplex, cell or group of cells

affected by your chosen task. Enclose the data set name in single quotes. If the data set does not exist, you will be prompted for data set allocation information.

3. Record the name of the data set on the applicable worksheet.

The settings are saved in the data set you selected.

Creating the customization jobs and files

You must select configuration data sets to use and complete the process of defining variables for this task. See "Choosing configuration data sets" on page 69 and "Setting the customization variables: Network Deployment cell" on page 70 for more information.

The Customization Dialog creates customization batch jobs and data files, based on the variable values you specified in the Dialog. The batch jobs and data sets will be written to the config_hlq.CNTL and config_hlq.DATA configuration data sets that you created with the 'Allocate target data sets' option.

1. Ensure the configuration data sets are allocated and not in use.

   Note: Editing a member in config_hlq.CNTL or config_hlq.DATA will cause this task to fail.

2. On the 'Configure Deployment Manager Node' panel, type "4" in the *Option* field to select "Generate customization jobs" and press **Enter**. You will have one of two results:

   - **Result A:** If all variables are defined correctly, you see the 'Specify Job Cards' panel, which looks similar to this:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Generate Customization Jobs

 This portion of the Customization Dialog generates the jobs you must
 run after you complete this Dialog process. You must complete the
 customization process before you generate the jobs with this step.
 If you have not done this, please return to that step.

 Jobs and data files will get generated into data sets:
   'hlq.CNTL'
   'hlq.DATA'
 If you wish to generate customization jobs using other data sets, then
 exit from this panel and select option 1 (Allocate target data sets).

 All the jobs that will be tailored for you will need a job card.
 Please enter a valid job card for your installation below. The
 file tailoring process will update the jobname for you in all the
 generated jobs, so you need not be concerned with that portion of
 the job cards below. If continuations are needed, replace the
 comment cards with continuations.

 Specify the job cards, then press Enter to continue.

 //jobname  JOB (ACCTNO,ROOM),'userid',CLASS=A,REGION=0M
 //*
 //*
 //*
```

     Note: Pay particular attention to the displayed target data sets. Make sure they are the ones you intend to use.

- **Result B:** If the variables are not defined correctly, you will see the 'Verification' panel. Decide whether the warnings or errors are serious enough to warrant returning to the "Define variables" option.

  Note: If the return code is "8" or greater, return to the "Define variables" option and fix the uncovered problems. If you saved the variables previously, be sure to re-save them after making any updates.

3. Fill in the job card information, according to your installation requirements. For each job, the Dialog generates a jobname and the "JOB" keyword to match the member name of the PDS, but you specify the rest.

   Note: If you need to run these jobs on a particular system in the sysplex (for example, JES2 MAS or JES3 complex), you should specify the necessary Scheduling Environment (SCHENV), JES2 JOBPARM, or JES3 //*MAIN statement at this time.

   Example of a job card entry:

   ```
   //jobname JOB 1234,USER1,NOTIFY=????,MSGCLASS=O,REGION=0M
   //*            USER=SYSADM1,PASSWORD=SYSADM1
   /*JOBPARM SYSAFF=SYSB
   ```

   Note: This example is useful for jobs that require a user ID other than that of the logged-on TSO user. (This is typically a user ID with UID=0.) In that case, you can just put a comma at the end of the first line, put in the correct user ID on the second line, then uncomment that second line.

   You may wish to use RACF SUBMIT authority to avoid having to keep passwords in your configuration data sets.

4. Fix any errors. If there are errors anywhere, you will see the 'Error' panel. Press PF3 to exit the error panel, then enter the correct panel to fix the errors. Then return to the "Generate Customization Jobs" option and pick up where you left off. If necessary, you can update the variables and rerun this option. The generation process will delete and re-tailor all the members.

   Note: Compress the configuration data sets before you rerun this option.

You are done when all the jobs are generated. You may then move ahead to viewing the generated jobs. See "Following the generated customization instructions: Network Deployment cell" for more information.

Following the generated customization instructions: Network Deployment cell

You must generate the customization jobs and files for this task.

The Customization Dialog creates a set of instructions for each customization task. Follow these instructions to tailor and customize a Network Deployment cell on your system.

Note: Do not attempt to fix a typo or make a change by modifying the generated output. Many of the variables are used in multiple members of the target data sets, so, if you do not change them all, you will run into problems that are very difficult to diagnose.

1. On the 'Configure Deployment Manager Node' panel, type "5" in the *Option* field to select "View instructions" and press **Enter**. ISPF Browse will open and you will see the BBOCCINS member of config_hlq.CNTL.

2. Read the instructions carefully, both to preview the customization process and to find any typographical or other errors you may have made while entering the customization variable values.
3. Follow the instructions as given. There are two ways to follow the directions:
   - Follow the instructions while remaining in ISPF Browse.
   - Record the data set name and member at the top of the screen and either print the instructions or use ISPF split screen and browse or edit the instructions while you follow them.
4. Fix any problems. If you encounter problems caused by your Customization Dialog values, modify your variables using the Dialog, regenerate the instructions, and restart the customization process.

   Note: Remember that you cannot generate new customization jobs while either configuration data set is open!

You are done with this customization task when you have successfully followed the generated instructions.

Sample generated instructions: Network Deployment cell

This article presents a sample of what the Customization Dialog's generated instructions may look like. This is a sample only--you must use the instructions generated from your own variables when configuring your system.

```
-----------------------------------------------
Instructions for customizing WebSphere for z/OS
for a Deployment Manager node.

The customization dialog has created jobs based on the information you
provided. These instructions tell you how to modify the operating
system and run the jobs to customize WebSphere for z/OS.

RULES:

1.  If you created the target data sets (*.CNTL and *.DATA) on another
    (driving) system, you must copy them to the target system and give
    them the same data set names.

2.  You must perform these instructions on your target system.

Doing manual configuration updates
----------------------------------

The customization dialog for WebSphere for z/OS does not attempt to
update configuration data for your base operating system or existing
subsystems. You must do the following manual steps prior to running
the WebSphere for z/OS configuration jobs.

Perform these steps to do manual configuration updates:

1.  Update the workload management application environment as follows.

    ATTENTION: If you have already installed the WLM-DAE support PTF
    (APAR OW54622) on z/OS 1.4, or are running a higher level of
    z/OS, you may skip this step.

    Using these parameters, run IWMARIN0 to create the BBODMGR
    application environment:

    Appl Environment Name    BBODMGR
    Description              WebSphere for z/OS V6 servant
    Subsystem type           CB
    Procedure name           BBO6DSR
```

```
Start parameters         JOBNAME=&IWMSSNM.S;,
                           ENV=MCLXCF01.MCLXCF01.&IWMSSNM;
Limit on starting server address spaces for a subsystem instance:
     Single address space per system
```

NOTE: For the start parameter string, you must continue typing with ENV on the same line as JOBNAME. We show the parameters on separate lines for aesthetics only.

For information about IWMARIN0, the ISPF dialog application for MVS workload management, see z/OS MVS Planning: Workload Management (SA22-7602).

For more information about workload management and WebSphere for z/OS, see related topics in the WebSphere for z/OS Information Center at http://www.ibm.com/software/webservers/appserv/zos_os390/library/.

---

2. Update your active BPXPRMxx member to have the following WebSphere for z/OS configuration HFS:

   OMVS.WAS.CONFIG.HFS

   mounted at:

   /WebSphere/V6R0

   in read/write mode.

   EXAMPLE:

   ```
   MOUNT FILESYSTEM('OMVS.WAS.CONFIG.HFS')
     MOUNTPOINT('/WebSphere/V6R0')
      TYPE(HFS)
      MODE(RDWR)
   ```

   ---

3. Update TCP/IP by reserving the following ports for WebSphere for z/OS:

   ```
   SOAP JMX Connector port                      - 8879
   CELL DISCOVERY ADDRESS port                  - 7277
   ORB port                                     - 9809
   HTTP port                                    - 9060
   HTTP SSL port                                - 9043

   High Availability Manager Communications port - 9352

   Daemon IP port                               - 5755
   Daemon SSL port                              - 5756
   ```

   View member BBOTCPID in

   'DATASET.CNTL'

   Add the contents of this member to the PORT section of the file referenced by the DD statement for the TCP/IP profile in the TCP/IP start procedure. Cut and paste from this member into the data set used by your installation.

   ATTENTION: If another application has already reserved any of these ports for its own use, you must resolve the resulting conflict before you continue. If you update the WebSphere for z/OS customization dialog with new port specifications, be sure to

regenerate the customization jobs, data, and instuctions.

---

4. The WebSphere product libraries will be placed in STEPLIB as needed, rather than in the system link pack area and system link list.

   SBBOLPA:
   =========
   The following data set will be placed in the STEPLIB concatentation for the location service daemon. The data set's load modules will be loaded in the MVS common storage area when the location service daemon is started. Make sure the target MVS system has at least 8MB free storage in extended CSA before starting the daemon.

   BOSS.VICOM.W000170.SBBOLPA

   SBBOLOAD and SBBOLD2:
   ====================
   The following data sets will be placed in the STEPLIB concatenation for the location service daemon, controller and servant regions, and in the setupCmdLine.sh script in the WebSphere Configuration HFS. You must not remove these STEPLIB statements.

   BOSS.VICOM.W000170.SBBOLOAD
   BOSS.VICOM.W000170.SBBOLD2

   BBORTSS5:
   =========

   The BBORTSS5 module is used by WebSphere Application Server V6 for component trace support. A copy of this module must be in the system link pack area in order for CTRACE to work correctly.

   If a copy of BBORTSS5 (any release) is currently loaded into LPA, you need take no further action.

   Otherwise, issue the following MVS console command to load BBORTSS5 into dynamic LPA:

   ```
   SETPROG LPA,ADD,MODNAME=BBORTSS5,
              DSNAME=BOSS.VICOM.W000170.SBBOLPA
   ```

   Alternatively, you can place the following statement in a parmlib PROGxx member which is activated with the SET PROG= command after system IPL is complete:

   ```
   LPA ADD MODNAME(BBORTSS5) DSNAME(BOSS.VICOM.W000170.SBBOLPA)
   ```

   Make sure that the BBORTSS5 module is loaded into LPA after each system IPL.

   ---

5. Make sure the following Language Environment data set is in the system link list:

   CEE.SCEERUN

   ---

6. Make sure the following Language Environment data set is in the system linklist:

```
    CEE.SCEERUN2

    ---------------------------------------------------------------

7.  Make sure the following System SSL data set is in the system link
    pack area or the system link list on all MVS systems at or below
    z/OS Version 1.5:

    GSK.SGSKLOAD


    ---------------------------------------------------------------

8.  WebSphere for z/OS regions open a large number of files (more than
    1024). Make sure your BPXPRMxx parmlib member(s) specify a value of
    MAXFILEPROC that is greater than or equal to 2000. Use the
    following MVS console command to see your current MAXFILEPROC
    setting:

      D OMVS,OPTIONS

    ---------------------------------------------------------------

9.  Grant WSDMNCR1 read access to
    SYS1.PARMLIB
    and any other parmlibs that precede
    SYS1.PARMLIB
    in the parmlib concatenation of the LOADzz member in IPLPARM. Use
    the D IPLINFO command to determine which LOADzz member is current
    and where it resides.


Running the customized jobs
---------------------------

The customization dialog built a number of batch jobs with the
variables you supplied. You must run the jobs in the order listed
below using user IDs with the appropriate authority.

BEFORE YOU BEGIN: Complete the section above entitled "Doing manual
configuration updates."

Follow the table below, which lists in order the jobs you must submit
and the commands you must enter. Special handling notes are included
in the table. All jobs are members of

DATASET.CNTL

Attention: After submitting each job, carefully check the output.
Errors may exist even when all return codes are zero.

+-----------+-------------------------------------------------------+
| BBODBRAJ  | User ID requirement: Authority to update data set     |
+-----------+                                                       |
| Done:     | DATASET.DATA                                          |
|           |                                                       |
|           | This job builds (but does not execute) the RACF commands
|           | for the WebSphere for z/OS run-time clusters and places
| By:       | them into member BBODBRAK of data set                 |
|           |                                                       |
|           | DATASET.DATA                                          |
|           |                                                       |
|           | Carefully review these definitions with your security |
|           | administrator.                                        |
+-----------+-------------------------------------------------------+
| BBODBRAK  | User ID requirement: RACF special authority.          |
+-----------+                                                       |
```

Done:	This job executes the RACF commands set up in the previous job. RESULT: You may receive errors, such as INVALID USER messages, from this job because a user ID, group or profile is already defined. Make sure the existing user ID, group or profile has the same characteristics as the user ID, group or profile being created by BBOCBRAK. If not, then change the values in the customization dialog which are causing the conflict, regenerate the customization jobs, and restart the process.
By:	ATTENTION: If operator commands are protected by the z/OS security server at your installation, you must ensure that sufficient authority is given to WebSphere tasks to control operations. The Deployment Manager Location Service Daemon user ID, (WSDMNCR1), the Deployment Manager Controller user ID (DMCR1), and the Deployment Manager Servant user ID (DMSR1) need the ability to perform operations on started tasks belonging to WebSphere Application Server for z/OS in the same cell (or security domain). The RACF commands to authorize these user IDs are not generated by the previous job (BBODBRAJ). The following example commands show what needs to be done, substituting your profile names: PERMIT START_profile_name CLASS(OPERCMDS) ID(WSDMNCR1 DMCR1 DMSR1) ACC(UPDATE) PERMIT STOP_profile_name CLASS(OPERCMDS) ID(WSDMNCR1 DMCR1 DMSR1) ACC(UPDATE) PERMIT MODIFY_profile_name CLASS(OPERCMDS) ID(WSDMNCR1 DMCR1 DMSR1) ACC(UPDATE) PERMIT CANCEL_profile_name CLASS(OPERCMDS) ID(WSDMNCR1 DMCR1 DMSR1) ACC(UPDATE) PERMIT FORCE_profile_name CLASS(OPERCMDS) ID(WSDMNCR1 DMCR1 DMSR1) ACC(UPDATE)
BBODCHFS	User ID requirement: UID=0 and authority to allocate
Done:	OMVS.WAS.CONFIG.HFS ATTENTION: Skip this step if the mount point is already created, such as for the stand-alone application server
By:	This job: o Creates a mount point directory /WebSphere/V6R0

	o Allocates the configuration HFS OMVS.WAS.CONFIG.HFS and mounts it at the above mount point. BEFORE YOU BEGIN: The BBODCHFS job assumes your root HFS is mounted in read/write mode. If the root HFS is not in read/write mode, manually create the directory /WebSphere/V6R0 and any needed higher directories. Set file permissions to 775 and set the owning user ID and group to WSADMIN and WSCFG1 before running BBODCHFS. EXAMPLE: If you plan to use /WebSphere/V6R0 as your directory, issue the following commands from within the OMVS shell: mkdir -p -m 775 /WebSphere/V6R0 chown -R WSADMIN:WSCFG1 /WebSphere
BBODHFSA	User ID requirement: UID=0.
Done:	This job populates the previously-created HFS. Upon completion, examine the job output. Success is indicated with a RC=0 in the job output.
By:	
BBODCPY1	User ID requirement: update authority for:
Done:	 SYS1.PROCLIB
By:	 ATTENTION: This job modifies SYS1.PROCLIB. Because master subsystem address spaces like ATRRRS and BBOWTR must have their cataloged procedures in a PROCLIB listed in the master scheduler JCL, we copy members to SYS1.PROCLIB. You may have a private master subsystem PROCLIB to which you want to copy the members. This job copies the tailored start procedures, parameters, and EXECs to the run-time product libraries ATTENTION: Be aware that you may overlay existing members in the above data set.
BBOWWPFD	User ID requirement: UID=0.
Done:	This job sets up the runtime HFS.
By:	Upon completion, examine the job output. Success is indicated by rc=0.
BBODHFSB	User ID requirement: UID=0.
Done:	This job will complete the HFS initialization.

```
|           | Upon completion, examine the job output. Success is      |
| By:       | indicated with a RC=0 in the job output.                 |
|           |                                                          |
+-----------+----------------------------------------------------------+
| --------  | If Resource Recovery Services (RRS) is not active, issue |
+-----------+ the following MVS command. (To check if RRS is active,   |
| Done:     | go to SDSF or the operator console and see if "RRS" is   |
|           | listed as active).                                       |
|           |                                                          |
| By:       |   START ATRRRS,SUB=MSTR                                  |
|           |                                                          |
|           | Note: You know you were successful if the following      |
|           | appears in the SYSLOG:                                   |
|           |                                                          |
|           |   ASA2011I RRS INITIALIZATION COMPLETE. COMPONENT        |
|           |     ID=SCRRS                                             |
+-----------+----------------------------------------------------------+
| --------  | Start the Deployment Manager.                            |
+-----------+                                                          |
| Done:     | Issue the MVS command                                    |
|           |                                                          |
|           |   START BBO6DCR,JOBNAME=BBODMGR,                         |
|           |         ENV=MCLXCF01.MCLXCF01.BBODMGR                    |
|           |                                                          |
| By:       | This command starts the Deployment Manager and also      |
|           | starts the location service daemon. Wait until the       |
|           | server is finished initializing before proceeding.       |
|           |                                                          |
|           | RESULT: The following message appears on the console and |
|           | in the job log of BBODMGR                                |
|           |                                                          |
|           |   BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR    |
|           |     z/OS CONTROL PROCESS BBODMGR                         |
|           |                                                          |
+-----------+----------------------------------------------------------+
|                                                                      |
| The product is now configured for a Deployment Manager.              |
|                                                                      |
+----------------------------------------------------------------------+
```

Note: If you want to federate a managed node or stand-alone Application Server node, you will need to run option 2 or 3 of the dialog, respectively.

```
+----------------------------------------------------------------------+
| To start the Deployment Manager, issue the following MVS command:    |
|                                                                      |
|   START BBO6DCR,JOBNAME=BBODMGR,                                     |
|         ENV=MCLXCF01.MCLXCF01.BBODMGR                                |
|                                                                      |
| To stop the WebSphere for z/OS servers, enter the MVS command:       |
|                                                                      |
|   STOP BBODMNC                                                       |
|                                                                      |
+----------------------------------------------------------------------+
```

The following is a useful script that helps you define security controls. It is in data set

'DATASET.DATA'

```
+-----------+----------------------------------------------------------+
| BBODBRAC  | This is a sample exec that you may modify to include     |
+-----------+ installation-specific RACF controls. This exec defines   |
| Done:     | all the user IDs and groups that are necessary and       |
|           | sufficient for installing WebSphere for z/OS.            |
|           |                                                          |
```

By:	Additionally, there are commented sections for other components that might be used (SSL, for example).

Working with your new deployment manager

Once you complete the customization instructions, you will have a WebSphere Application Server for z/OS Network Deployment cell. The Network Deployment cell consists of a deployment manager and a location service daemon. (To run J2EE applications, you must add application server nodes. See below for details.) This article provides useful information for working with your new Network Deployment cell.

Before starting your server

Make sure the WebSphere Application Server for z/OS product HFS and configuration HFS are mounted. If you chose to load the SBBOLPA (and possibly SBBOLOAD) into the system link pack area, make sure these libraries are loaded into LPA before starting the server.

Starting the deployment manager

To start your deployment manager, issue the following MVS console command:

```
START server_proc,JOBNAME=dmgr_name,ENV=cell_name.node_name.dmgr_name
```

where:

- *server_proc* is the deployment manager controller cataloged procedure.
- *dmgr_name* is the deployment manager short name.
- *node_name* is the deployment manager node short name.
- *cell_name* is the cell short name.

For example, if you chose default values and your system is named MVSA, you would enter the following START command:

```
START BBO6DCR,JOBNAME=BBODMGR,ENV=MVSA.MVSA.BBODMGR
```

The START command brings up the deployment manager controller. The controller starts the location service daemon, then uses WLM to start the deployment manager servant. You should see a message like the following when the deployment manager is up and running:

BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR z/OS CONTROL PROCESS BBODMGR

Accessing the server administrative console

Once the deployment manager is successfully started, access the administrative console by pointing a Web browser to the following URL:

http://*hostname*:*http_port*/ibm/console

where:

- *hostname* is the deployment manager HTTP transport host name you specified during customization.

Note: If you specified '*' for the deployment manager HTTP host name, this is actually the deployment manager node host name.

- *http_port* is the deployment manager HTTP port you specified during customization.

  Note: The default HTTP port for the deployment manager is 9060.

Until global security is enabled, you will see a sign-on screen that asks you for a user ID but no password.

The user ID can be anything and is used only to provide basic tracking of changes. Be aware that, until you enable global security, anyone with a Web browser and access to the HTTP port can modify your application serving environment.

You can use the administrative console, scripting, or both to manage the Network Deployment cell and deploy and manage J2EE applications. Before you can deploy applications, however, you need to add application server nodes to your Network Deployment cell.

Adding application server nodes

Application server nodes (also called managed nodes) in a Network Deployment cell consist of a node agent and any number of application servers per node.

Note: Each z/OS system also needs one location service daemon for each stand-alone or Network Deployment cell hosted on the system.

Add an application server node to a Network Deployment cell using one of two methods:

- Create an (empty) managed node using the Customization Dialog. The new node can reside on the same or a different z/OS system as the deployment manager. The new managed node, consisting of just a node agent and perhaps a location service daemon, is federated into the Network Deployment cell. Once this is done, you can use the administrative console or scripting to add application servers and deploy and manage J2EE applications in the node. See the *Installing your application serving environment* PDF for more information.
- Federate an existing stand-alone application server into the Network Deployment cell. The stand-alone server node becomes a managed node in the Network Deployment cell, along with any J2EE applications that have been deployed on it. See the *Installing your application serving environment* PDF for more information.

Stopping your deployment manager

Use one of the following two methods to stop your deployment manager:

- Stop the location service daemon, which also stops the deployment manager and any of the cell's managed nodes on the same z/OS system. The location service daemon holds pointers to modules in common storage, and stopping it forces the cell's nodes on the same z/OS system as the location service daemon to shut down. To stop the location service daemon, enter the following MVS console command:

  `STOP daemon_jobname`where *daemon_jobname* is the location service daemon jobname. The default location service daemon jobname for a Network Deployment cell is BBODMNC.

Note: This is the easiest way to stop the deployment manager.

- Stop just the deployment manager, leaving the location service daemon and any managed nodes on the z/OS system still running. This works because the deployment manager is used to administer only the cell--it does not need to be up for J2EE applications in the cell to run. To stop the deployment manager, enter the following MVS console command:

  `STOP dmgr_name`where *dmgr_name* is the deployment manager short name. The default deployment manager short name is BBODMGR.

Creating a managed server node

This article leads you through the tasks involved in creating a WebSphere Application Server for z/OS managed server node.

Perform this task to create a new WebSphere Application Server for z/OS managed node.

1. Log on to TSO on the z/OS system on which you intend to configure the managed node. Use a user ID that has READ access to the WebSphere Application Server for z/OS product data sets. You will also need access to a user ID with authority to make security system updates and a user ID with UID 0. (These can all be the same user ID.)
2. Start the Customization Dialog. See “Starting the Customization Dialog” on page 31 for details.
3. Choose the configuration data sets in which you will store your customization jobs and data. See “Choosing configuration data sets” on page 86 for details.
4. Load the security domain variables saved from the security domain you intend to use for this cell. See “Loading the security domain variables” for details. (Optionally, you can use the L command to load the Network Deployment cell variables if you saved them during Network Deployment cell setup. Network Deployment cell variables include the security domain variables--you do not need to load both.)
5. Set the customization variables according to the values recorded on your managed node worksheet. See “Setting the customization variables: Managed node” on page 87 for details.
6. (Optional but recommended.) Save the managed node customization variables in a data set. See “Saving the cell variables” on page 73 for details.
7. Create the customization jobs and files, based on the customization variable values you entered. See “Creating the customization jobs and files” on page 90 for details.
8. Follow the generated customization instructions. See “Following the generated customization instructions: Managed node” on page 92 for details, and a sample set of customization instructions.

You are done when you have successfully completed the steps in the generated instructions. The new managed node is up and running on the chosen z/OS system. See “Working with your new managed server node” on page 98 for more information.

Loading the security domain variables

This article describes how to complete the ″Load security domain variables″ option for a WebSphere Application Server for z/OS managed node.

Create the security domain you will use for the new managed node and know the name of the saved security domain configuration variable file that you recorded on the security domain worksheet.

The security domain settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the security domain variables at the start of node or cell creation, you ensure that the security domain configuration you use is consistent and matches the RACF definitions that have already been set as part of security domain configuration.

Complete this task as the first step in configuring a new managed node. If you encounter problems during customization and change the security domain variable values, be sure to re-save them.

1. On the 'Configure Managed Node' panel, type "1" in the *Option* field to select "Load security domain variables" and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Load Security Domain Variables

 Specify the name of a data set containing the security domain variables,
 then press Enter to continue.

 IBM-supplied defaults are in ''


 Data set name:


 If this data set is not cataloged, specify the volume.

 Volume:
```

2. Fill in the name of the sequential data set you used to hold the security domain variable values and press **Enter**.

   Note: Ensure you enclose the data set name in single quotes.
   The security domain variables will load.

The security domain settings are loaded. You can display these variables, but not change them.

Choosing configuration data sets

This article leads you through the "Allocate target data sets" option in the Customization Dialog.

You must start the Customization Dialog and select the "Create an empty managed node and add it to an existing Network Deployment cell." option.

Each option in the Customization Dialog saves customization jobs and files in a pair of customization data sets. While is it possible to reuse these data sets, it is safest to create separate data sets for each WebSphere Application Server for z/OS configuration. We recommend that you use the customization data set name prefix (sometimes referred to as "config_hlq") to indicate the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring. For example, you might

use the following data set name prefix for configuring a WebSphere Application Server for z/OS 6.0.1 managed node named MA6N01 for cell MAINT1:

JULIA.WASV6R1.MAINT1.MA6N01.MANAGED

Complete this task before generating the customization jobs and files.

1. On the main Dialog panel, type ″2″ in the *Option* field to select ″Allocate target data sets″.
2. Press Enter. **Result:** You see a panel that looks similar to the following:

```
-----     WebSphere Application Server for z/OS Customization      --------
Option  ===>

 Allocate Target Data Sets

 Specify a high level qualifier (HLQ) and press Enter to allocate the
 data sets to contain the generated jobs and instructions. You can
 specify multiple qualifiers (up to 39 characters).

 High level qualifier:                                          .CNTL
                                                                .DATA

 The Dialog will display data set allocation panels. You can make
 changes to the default allocations, however you should not change
 the DCB characteristics of the data sets.

    .CNTL  - a PDS with fixed block 80-byte records to
             contain customization jobs.

    .DATA  - a PDS with variable length data to contain
             other data produced by the Customization Dialog.
```

3. Fill in your chosen configuration data set name prefix value (config_hlq). If the data sets ″config_hlq.CNTL″ and ″config_hlq.DATA″ do not exist, you will be prompted for data set allocation information. If the data sets already exist, a message will inform you that they will be reused.

The data sets ″config_hlq.CNTL″ and ″config_hlq.DATA″ are allocated and will store customization jobs and files. These data set names will also be saved along with the customization variables.

Setting the customization variables: Managed node

This article describes how to complete the ″Define variables″ option for a WebSphere Application Server for z/OS managed node.

You must start the Customization Dialog and select the 'Create Network Deployment cells and nodes' option then the 'Create an empty managed node and add it to an existing Network Deployment cell' option. Have the `Customization Dialog worksheet: Managed node` completed and at hand.

1. On the 'Configure Managed Node' panel, type ″3″ in the *Option* field to select ″Define variables″ and press **Enter**.
2. On the 'Define Variables to Configure stand-alone application server Node' panel, type ″1″ in the *Option* field to select ″System Locations (directories, HLQs, etc.)″ and press **Enter**.
3. Fill in the System Locations panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

   System Locations

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (1 of 2)

   Specify the following for the system on which you are installing
   WebSphere Application Server for z/OS, then press Enter to continue.
   For some data sets, specify "Y" if they are in STEPLIB.

   System name.:  AQTS      Sysplex name :  MCLXCF01

 Full Names of Data Sets

   PROCLIB:  SYS1.PROCLIB
   PARMLIB:  SYS1.PARMLIB
   SYSEXEC:

   Run WebSphere Application Server from STEPLIB (Y/N)?  Y
   SBBOLPA.:  BOSS.VICOM.W000170.SBBOLPA
   SBBOLOAD:  BOSS.VICOM.W000170.SBBOLOAD
   SBBOLD2.:  BOSS.VICOM.W000170.SBBOLD2
   SBBOEXEC:  BOSS.VICOM.W000170.SBBOEXEC
   SBBOMSG.:  BOSS.VICOM.W000170.SBBOMSG

                                                        Use STEPLIB?
   SCEERUN.:  CEE.SCEERUN                                    N
   SCEERUN2:  CEE.SCEERUN2                                   N
   SGSKLOAD:  GSK.SGSKLOAD                                   N
              (leave SGSKLOAD blank if all systems are at z/OS 1.6 or above)
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Locations (2 of 2)

   Specify the following for your customization, then press Enter
   to continue.

 Locations of HFS Resident Components

   WebSphere Application Server product directory:
     /usr/lpp/zWebSphere/V6R0
```

4. On the 'Define Variables to Configure Managed Node' panel, type "2" in the *Option* field to select "System Environment Customization" and press **Enter**.
5. Fill in the System Environment Customization panel using the following screen shot as your guide. When you are done, press **Enter**.

   System Environment Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

System Environment Customization (1 of 1)

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /WebSphere/V6R0
   Name...........:  OMVS.WAS.CONFIG.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  250
   Secondary allocation in cylinders.:  100
```

6. On the 'Define Variables to Configure Managed Node' panel, type "3" in the *Option* field to select "Server Customization" and press **Enter**.

7. Fill in the Server Customization panels using the following screen shots as your guides. When you are done with each panel, press **Enter**.

Server Customization

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (1 of 4)

   Specify the following to customize your node, then press Enter
   to continue.

 Cell and Node Definitions

   WebSphere Application Server home directory:
     /WebSphere/V6R0
         / AppServer

   Node Host Name:  QWEQ


   Cell name (short)......:  AQTS
   Cell name (long).......:  AQTS

   Node name (short)......:  AQTS
   Node name (long).......:  AQTS

   Admin asynch operations procedure name:  BBOW6SH
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (2 of 4)

   Specify the following to customize your node, then press Enter
   to continue.

 Procedure Name Definitions

   Controller Information

     Procedure name:  BBO6ACR
     User ID.......:  ASCR1
     UID...........:  2431

   Servant Information

     Procedure name:  BBO6ASR
     User ID.......:  ASSR1
     UID...........:  2432

   Control Region Adjunct

     Procedure name:  BBO6CRA
     User ID.......:  ASCRA1
     UID...........:  2433
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (3 of 4)

   Specify the following to customize daemon definitions for your
   node, then press Enter to continue

 Location Service Daemon Definitions

   Daemon home directory:
```

```
     /WebSphere/V6R0/Daemon

   Daemon jobname:  BBODMNB

   Procedure name.:  BBO6DMN
   User ID........:  WSDMNCR1
   UID............:  2411
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Server Customization (4 of 4)
   Specify the following, which will be used in a job to
   federate your node into the specified Deployment Manager cell.

 WebSphere Application Server home directory:
     /WebSphere/V6R0
         / AppServer
 Deployment Manager Access
   Node host name...........:
   JMX SOAP port............:  8879
   Deployment manager security is enabled:  N
      User ID..............:  WSADMIN

 Node group name.......:  DefaultNodeGroup

 Node Agent Definitions
   Server name (short)...:  BBON001
   Server name (long)....:  nodeagent
   JMX SOAP connector port.........:  9360
   Node Discovery port.............:  7272
   Node Multicast Discovery port..:  5000

 The High Availability Manager Host must not be multihomed
 High Availability Manager Host:
   High availability manager communication port:  9354

 ORB listener host name...:  *
   ORB port.......................:  2809
   ORB SSL port...................:  0
```

8. On the 'Define Variables to Configure Managed Node' panel, type "4" in the *Option* field to select "View Security Domain Configuration Panels" and press **Enter**. These panels display values you previously set in the "Configure security domain" option--you can not change any of the values here. If you do wish to make changes, you must go back to the main Dialog panel and run through the "Configure security domain" option again.

Creating the customization jobs and files

You must select configuration data sets to use and complete the process of defining variables for this task. See "Choosing configuration data sets" on page 86 and "Setting the customization variables: Managed node" on page 87 for more information.

The Customization Dialog creates customization batch jobs and data files, based on the variable values you specified in the Dialog. The batch jobs and data sets will be written to the config_hlq.CNTL and config_hlq.DATA configuration data sets that you created with the 'Allocate target data sets' option.

1. Ensure the configuration data sets are allocated and not in use.

   Note: Editing a member in config_hlq.CNTL or config_hlq.DATA will cause this task to fail.

2. On the 'Configure Managed Node' panel, type "4" in the *Option* field to select "Generate customization jobs" and press **Enter**. You will have one of two results:
   - **Result A:** If all variables are defined correctly, you see the 'Specify Job Cards' panel, which looks similar to this:

```
--------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Generate Customization Jobs

 This portion of the Customization Dialog generates the jobs you must
 run after you complete this Dialog process. You must complete the
 customization process before you generate the jobs with this step.
 If you have not done this, please return to that step.

 Jobs and data files will get generated into data sets:
   'hlq.CNTL'
   'hlq.DATA'
 If you wish to generate customization jobs using other data sets, then
 exit from this panel and select option 1 (Allocate target data sets).

 All the jobs that will be tailored for you will need a job card.
 Please enter a valid job card for your installation below. The
 file tailoring process will update the jobname for you in all the
 generated jobs, so you need not be concerned with that portion of
 the job cards below. If continuations are needed, replace the
 comment cards with continuations.

 Specify the job cards, then press Enter to continue.

 //jobname  JOB (ACCTNO,ROOM),'userid',CLASS=A,REGION=0M
 //*
 //*
 //*
```

     Note: Pay particular attention to the displayed target data sets. Make sure they are the ones you intend to use.
   - **Result B:** If the variables are not defined correctly, you will see the 'Verification' panel. Decide whether the warnings or errors are serious enough to warrant returning to the "Define variables" option.

     Note: If the return code is "8" or greater, return to the "Define variables" option and fix the uncovered problems. If you saved the variables previously, be sure to re-save them after making any updates.
3. Fill in the job card information, according to your installation requirements. For each job, the Dialog generates a jobname and the "JOB" keyword to match the member name of the PDS, but you specify the rest.

   Note: If you need to run these jobs on a particular system in the sysplex (for example, JES2 MAS or JES3 complex), you should specify the necessary Scheduling Environment (SCHENV), JES2 JOBPARM, or JES3 //*MAIN statement at this time.

   Example of a job card entry:

```
//jobname JOB 1234,USER1,NOTIFY=????,MSGCLASS=O,REGION=0M
//*             USER=SYSADM1,PASSWORD=SYSADM1
/*JOBPARM SYSAFF=SYSB
```

   Note: This example is useful for jobs that require a user ID other than that of the logged-on TSO user. (This is typically a user ID with UID=0.) In that case, you can just put a comma at the end of the first line, put in the

correct user ID on the second line, then uncomment that second line. You may wish to use RACF SUBMIT authority to avoid having to keep passwords in your configuration data sets.

4. Fix any errors. If there are errors anywhere, you will see the 'Error' panel. Press PF3 to exit the error panel, then enter the correct panel to fix the errors. Then return to the "Generate Customization Jobs" option and pick up where you left off. If necessary, you can update the variables and rerun this option. The generation process will delete and re-tailor all the members.

   Note: Compress the configuration data sets before you rerun this option.

You are done when all the jobs are generated. You may then move ahead to viewing the generated jobs. See "Following the generated customization instructions: Managed node" for more information.

Following the generated customization instructions: Managed node

You must generate the customization jobs and files for this task.

The Customization Dialog creates a set of instructions for each customization task. Follow these instructions to tailor and customize a managed node on your system.

Note: Do not attempt to fix a typo or make a change by modifying the generated output. Many of the variables are used in multiple members of the target data sets, so, if you do not change them all, you will run into problems that are very difficult to diagnose.

1. On the 'Configure Managed Node' panel, type "5" in the *Option* field to select "View instructions" and press **Enter**. ISPF Browse will open and you will see the BBOMDINS member of config_hlq.CNTL.
2. Read the instructions carefully, both to preview the customization process and to find any typographical or other errors you may have made while entering the customization variable values.
3. Follow the instructions as given. There are two ways to follow the directions:
   - Follow the instructions while remaining in ISPF Browse.
   - Record the data set name and member at the top of the screen and either print the instructions or use ISPF split screen and browse or edit the instructions while you follow them.
4. Fix any problems. If you encounter problems caused by your Customization Dialog values, modify your variables using the Dialog, regenerate the instructions, and restart the customization process.

   Note: Remember that you cannot generate new customization jobs while either configuration data set is open!

You are done with this customization task when you have successfully followed the generated instructions.

Sample generated instructions: Managed node

This article presents a sample of what the Customization Dialog's generated instructions may look like. This is a sample only--you must use the instructions generated from your own variables when configuring your system.

```
---------------------------------------------
Instructions for customizing WebSphere for z/OS
for a Managed node.

The customization dialog has created jobs based on the information you
provided. These instructions tell you how to modify the operating
system and run the jobs to customize WebSphere for z/OS.

RULES:

1.  If you created the target data sets (*.CNTL and *.DATA) on another
    (driving) system, you must copy them to the target system and give
    them the same data set names.

2.  You must perform these instructions on your target system.

Doing manual configuration updates
---------------------------------

The customization dialog for WebSphere for z/OS does not attempt to
update configuration data for your base operating system or existing
subsystems. You must do the following manual steps prior to running
the WebSphere for z/OS configuration jobs.

Perform these steps to do manual configuration updates:

1.  Update the workload management application environment as follows.

    ATTENTION: If you have already installed the WLM-DAE support PTF
    (APAR OW54622) on z/OS 1.4, or are running a higher level of
    z/OS, you may skip this step.

    Using these parameters, run IWMARIN0 to create the
    <cluster_transition_name> application environment:

    Appl Environment Name    <cluster_transition_name>
    Description              WebSphere for z/OS V6 servant
    Subsystem type           CB
    Procedure name           BBO6ASR
    Start parameters         JOBNAME=&IWMSSNM.S;,
                               ENV=AQTS.AQTS.&IWMSSNM;
    Limit on starting server address spaces for a subsystem instance:
         Single address space per system

    NOTE: For the start parameter string, you must continue typing with
    ENV on the same line as JOBNAME. We show the parameters on separate
    lines for aesthetics only.

    For information about IWMARIN0, the ISPF dialog application for MVS
    workload management, see z/OS MVS Planning: Workload Management
    (SA22-7602).

    For more information about workload management and WebSphere for
    z/OS, see related topics in the WebSphere for z/OS Information
    Center at
    http://www.ibm.com/software/webservers/appserv/zos_os390/library/.

    ---------------------------------------------------------------

2.  Update BLSCUSER. Refer to member BBOIPCSP in

    'DATASET.CNTL'

    In order to use the IPCS support provided by the product, append
    the contents of this member to the BLSCUSER member in your IPCSPARM
    or system PARMLIB datasets.
```

```
---------------------------------------------------------------------

3.  Update SCHEDxx. Refer to member BBOSCHED in

    'DATASET.CNTL'

    In order to set the correct program properties for the WebSphere
    for z/OS run-time executables, append the contents of this member
    to the SCHEDxx member in your system PARMLIB concatenation.

    Note: When you are finished, issue the command SET SCH=(xx,xx)
    to activate SCHEDxx and load a new program properties table.

---------------------------------------------------------------------

4.  Make sure the following data sets are APF-authorized:

        BOSS.VICOM.W000170.SBBOLPA
        BOSS.VICOM.W000170.SBBOLOAD
        BOSS.VICOM.W000170.SBBOLD2
        CEE.SCEERUN
        CEE.SCEERUN2
        GSK.SGSKLOAD

    Add these datasets to your PROGxx or IEAAPFxx parmlib members, as
    appropriate, ensuring you specify the correct volsers.

---------------------------------------------------------------------

5.  If you want to collect the SMF120 records created by the run-time
    servers, update SMFPRMxx via the following:

    a.  Update the SYS or SUBSYS(STC,...) statement for started tasks
        to include the 120 record.
    b.  (Optional) You can specify designated subtypes 1-6.

    EXAMPLE:

       SUBSYS(STC,EXITS(IEFU29,IEFACTRT),INTERVAL(SMF,SYNC),
                          TYPE(0,30,70:79,88,89,120,245))
                                                 ---

    For details on the SMF records, see related topics in the
    WebSphere for z/OS Information Center at
    http://www.ibm.com/software/webservers/appserv/zos_os390/library/

---------------------------------------------------------------------

6.  Update your active BPXPRMxx member to have the following WebSphere
    for z/OS configuration HFS:

    OMVS.WAS.CONFIG.HFS

    mounted at:

    /WebSphere/V6R0

    in read/write mode.

    EXAMPLE:

       MOUNT FILESYSTEM('OMVS.WAS.CONFIG.HFS')
         MOUNTPOINT('/WebSphere/V6R0')
          TYPE(HFS)
          MODE(RDWR)

---------------------------------------------------------------------
```

7. The WebSphere product libraries will be placed in STEPLIB as needed, rather than in the system link pack area and system link list.

   SBBOLPA:
   ========
   The following data set will be placed in the STEPLIB concatentation for the location service daemon. The data set's load modules will be loaded in the MVS common storage area when the location service daemon is started. Make sure the target MVS system has at least 8MB free storage in extended CSA before starting the daemon.

   BOSS.VICOM.W000170.SBBOLPA

   SBBOLOAD and SBBOLD2:
   ====================
   The following data sets will be placed in the STEPLIB concatenation for the location service daemon, controller and servant regions, and in the setupCmdLine.sh script in the WebSphere Configuration HFS. You must not remove these STEPLIB statements.

   BOSS.VICOM.W000170.SBBOLOAD
   BOSS.VICOM.W000170.SBBOLD2

   BBORTSS5:
   =========

   The BBORTSS5 module is used by WebSphere Application Server V5 and V6 for component trace support. A copy of this module must be in the system link pack area in order for CTRACE to work correctly.

   If a copy of BBORTSS5 (any release) is currently loaded into LPA, you need take no further action.

   Otherwise, issue the following MVS console command to load BBORTSS5 into dynamic LPA:

   ```
   SETPROG LPA,ADD,MODNAME=BBORTSS5,
              DSNAME=BOSS.VICOM.W000170.SBBOLPA
   ```

   Alternatively, you can place the following statement in a parmlib PROGxx member which is activated with the SET PROG= command after system IPL is complete:

   ```
   LPA ADD MODNAME(BBORTSS5) DSNAME(BOSS.VICOM.W000170.SBBOLPA)
   ```

   Make sure that the BBORTSS5 module is loaded into LPA after each system IPL.

   ---

8. Make sure the following Language Environment data set is in the system link list:

   CEE.SCEERUN

   ---

9. Make sure the following Language Environment data set is in the system link list:

   CEE.SCEERUN2

   ---

10. Make sure the following System SSL data set is in the system link pack area or the system link list on all MVS systems at or below z/OS Version 1.5:

    GSK.SGSKLOAD

    ---

11. WebSphere for z/OS regions open a large number of files (more than 1024). Make sure your BPXPRMxx parmlib member(s) specify a value of MAXFILEPROC that is greater than or equal to 2000. Use the following MVS console command to see your current MAXFILEPROC setting:

      D OMVS,OPTIONS

Running the customized jobs

The customization dialog built a number of batch jobs with the variables you supplied. You must run the jobs in the order listed below using user IDs with the appropriate authority.

BEFORE YOU BEGIN: Complete the section above entitled "Doing manual configuration updates."

Follow the table below, which lists in order the jobs you must submit and the commands you must enter. Special handling notes are included in the table. All jobs are members of

DATASET.CNTL

Attention: After submitting each job, carefully check the output. Errors may exist even when all return codes are zero.

BBOMSGC	User ID requirement: Update authority for data set SYS1.MSGENU and/or SYS1.MSGJPN. ATTENTION: This is optional unless you require message translation. This job sets up MMS to translate messages for WebSphere for z/OS. There are two steps to update SYS1.MSGENU and SYS1.MSGJPN. Remove the unneeded step and change the target libraries, if necessary.
BBOWMNMT	User ID requirement: UID=0 and authority to allocate OMVS.WAS.CONFIG.HFS This job: o Creates a mount point directory /WebSphere/V6R0 o Allocates the configuration HFS OMVS.WAS.CONFIG.HFS and mounts it at the above mount point.

	BEFORE YOU BEGIN: The BBOWMNMT job assumes your root HFS is mounted in read/write mode. If the root HFS is not in read/write mode, manually create the directory /WebSphere/V6R0 and any needed higher directories, set file permissions to 775, and set the owning user ID and group to WSADMIN and WSCFG1 before running BBOWMNMT. EXAMPLE: If you plan to use /WebSphere/V6R0M0 as your directory, issue the following commands from within the OMVS shell: mkdir -p -m 775 /WebSphere/V6R0M0 chown -R WSADMIN:WSCFG1 /WebSphere
BBOMHFSA	User ID requirement: UID=0.
Done: By:	This job populates the previously-created HFS. Upon completion, examine the job output. Success is indicated with a RC=0 in the job output.
BBOWCPYM	User ID requirement: update authority for:
Done: By:	SYS1.PROCLIB This job copies the tailored start procedures, parameters, and EXECs to the run-time libraries. ATTENTION: Be aware that you may overlay existing members in the above data sets.
BBOWWPFM	User ID requirement: UID=0.
Done: By:	This job sets up the runtime HFS. Upon completion, examine the job output. Success is indicated by rc=0.
BBOMHFSB	User ID requirement: UID=0.
Done: By:	This job will complete the HFS initialization. Upon completion, examine the job output. Success is indicated with a RC=0 in the job output.
BBOWMNAN	User ID requirement: UID=0.
Done: By:	This job will federate your node into the specified Deployment Manager cell. Ensure that your Deployment Manager is running before submitting this job Upon completion, examine the job output. Success is indicated with a RC=0 in the job output.

```
+-----------+-------------------------------------------------------+
| --------  | Start the node agent server                           |
+-----------+                                                       |
|           | Note: The node agent is automatically started by the node|
|           | federation process.  This step is information, for    |
|           | starting the node agent at other times.               |
|           |                                                       |
|           | Issue the following MVS command to start your node agent|
|           | server:                                               |
| Done:     |                                                       |
|           |   START BBO6ACR,JOBNAME=BBON001,                      |
| By:       |         ENV=.AQTS.BBON001                             |
|           |                                                       |
|           |                                                       |
|           | RESULT: The following message appears on the console and|
|           | in the job log of BBON001.                            |
|           |                                                       |
|           |   BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR |
|           |   z/OS CONTROL PROCESS BBON001                        |
|           |                                                       |
+-----------+-------------------------------------------------------+
| The product is now configured.  You may create and manage         |
| application servers in the node using the administrative console or|
| scripting.                                                        |
|                                                                   |
+-------------------------------------------------------------------+
```

Working with your new managed server node

Once you complete the customization instructions, you will have a WebSphere Application Server for z/OS application server node (managed node) in your Network Deployment cell, with the same node name and server name(s) as your old stand-alone application server.

Before starting your server

Make sure the WebSphere Application Server for z/OS product HFS and configuration HFS are mounted. If you chose to load the SBBOLPA (and possibly SBBOLOAD) into the system link pack area, make sure these libraries are loaded into LPA before starting the server.

Starting the node agent

To start your node agent, issue the following MVS console command:

```
START
server_proc,JOBNAME=nodeagent_name,ENV=cell_name.node_name.nodeagent_name
```

where

- `server_proc` is the node agent cataloged procedure.
- `nodeagent_name` is the node agent short name.
- `node_name` is the node short name.
- `cell_name` is the cell short name.

For example, if you chose default values, your sysplex is named CELL1 and your system is named MVSA, you would enter the following START command:

```
START BBO6ACR,JOBNAME=BBON001,ENV=CELL1.MVSA.BBON001
```

The START command brings up the node agent. The node agent starts the location service daemon (if one is not already running). You should see a message like the following when the node is up and running:

BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR z/OS CONTROL PROCESS BBON001

The node agent must be running in order for the deployment manager to administer the node.

Administering the node through the administrative console

If the deployment manager for the cell is up and running, access the administrative console by pointing a Web browser to the following URL:

http://*hostname*:*http_port*/ibm/console

where

- *hostname* is the deployment manager HTTP transport host name you specified during customization.

  Note: If you specified '*' for the deployment manager HTTP host name, this is actually the deployment manager node host name.
- *http_port* is the deployment manager HTTP port you specified during customization.

  Note: The default HTTP port for the deployment manager is 9060.

Until global security is enabled, you will see a sign-on screen that asks you for a user ID but no password.

The user ID can be anything and is used only to provide basic tracking of changes. Be aware that, until you enable global security, anyone with a Web browser and access to the HTTP port can modify your application serving environment.

You can use the administrative console, scripting, or both to manage the node and deploy and manage J2EE applications. Before you can deploy applications, however, you need to add application servers to your managed node.

Adding application servers

Application servers can be added to the managed server node using the administrative console or scripting. Either of two methods can be used:

- Create a new application server directly using the administrative console or scripting. See "Creating application servers" on page 200 for more information. You can use the controller, servant and CRA cataloged procedures and user IDs created during the managed node setup process for any application servers you create in the managed node.
- Cluster an existing application server in another node, using this managed node as a target. This will create a "cloned" copy of the application server being clustered in your new managed node. See "Creating clusters" on page 333 for more information.

Starting application servers

To start one of your managed node's application servers, issue the following MVS console command:

```
START server_proc,JOBNAME=server_name,ENV=cell_name.node_name.server_name
```

where

- `server_proc` is the application server agent cataloged procedure (may be the same as the node agent cataloged procedure.
- `server_name` is the application server short name.
- `node_name` is the node short name.
- `cell_name` is the cell short name.

For example, if you chose the default procedure name, your sysplex is named CELL1, your node is named MVSA, and your server is named AZSR01A, you would enter the following START command:

```
START BBO6ACR,JOBNAME=AZSR01A,ENV=CELL1.MVSA.AZSR01A
```

The START command brings up the application server controller. The controller starts the location service daemon (if one is not already running), and then uses WLM to start the control region adjunct and the servant(s). You should see a message like the following when the node is up and running:

BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR z/OS CONTROL PROCESS AZSR01A

Stopping your managed node

Use one of the following two methods to stop your deployment manager:

- Stop the location service daemon, which also stops any of the cell's nodes on the same z/OS system. The location service daemon holds pointers to modules in common storage, and stopping it forces all cell members on the same z/OS system as the daemon to shut down. To stop the location service daemon, enter the following MVS console command:

  ```
  STOP daemon_jobname
  ```

  where `daemon_jobname` is the location service daemon jobname. The default location service daemon jobname for a Network Deployment cell is BBODMNC.
- Stop just the node agent and its application servers, leaving the location service daemon, the deployment manager (if present) and any other managed nodes on the z/OS system still running. To stop the node agent, enter the following MVS console command:

  ```
  STOP nodeagent_name
  ```

  where `nodeagent_name` is the node agent short name. The default deployment manager short name is BBON001.

Federating a stand-alone application server into a Network Deployment cell

This article leads you through the tasks involved in federating a WebSphere Application Server for z/OS stand-alone application server into a Network Deployment cell.

Perform this task to federate an existing WebSphere Application Server for z/OS stand-alone application server into a Network Deployment cell.

1. Log on to TSO on the z/OS system on which you intend to federate the server. Use a user ID that has READ access to the WebSphere Application Server for z/OS product data sets. You will also need access to a user ID with authority to make security system updates and a user ID with UID 0. (These can all be the same user ID.)
2. Start the Customization Dialog. See "Starting the Customization Dialog" on page 31 for details.
3. Choose the configuration data sets in which you will store your customization jobs and data. See "Choosing configuration data sets" on page 102 for details.
4. Load the security domain variables saved from the security domain you intend to use for this cell. See the *Securing applications and their environment* PDF for details.
5. Set the customization variables according to the values recorded on your federated application server node worksheet. See "Setting the customization variables: Federated application server node" on page 103 for details.
6. (Optional but recommended.) Save the federated application server node customization variables in a data set. See "Saving the cell variables" on page 73 for details. You will use these variables when creating managed nodes for the cell or federating stand-alone application servers into it.
7. Create the customization jobs and files, based on the customization variable values you entered. See "Creating the customization jobs and files" on page 104 for details.
8. Follow the generated customization instructions. See "Following the generated customization instructions: Federated application server node" on page 105 for details, and a sample set of customization instructions.

You are done when you have successfully completed the steps in the generated instructions. The new federated application server node is up and running on the chosen z/OS system. See "Working with your new federated server node" on page 109 for more information.

Loading the security domain variables

This article describes how to complete the ″Load security domain variables″ option for a WebSphere Application Server for z/OS federated application server node.

Create the security domain you will use for the new federated application server node and know the name of the saved security domain configuration variable file that you recorded on the security domain worksheet.

The security domain settings are used in the customization of every WebSphere Application Server for z/OS cell. By loading the security domain variables at the start of node or cell creation, you ensure that the security domain configuration you use is consistent and matches the RACF definitions that have already been set as part of security domain configuration.

Complete this task as the first step in configuring a new federated application server node. If you encounter problems during customization and change the security domain variable values, be sure to re-save them.

1. On the 'Federate stand-alone application server node' panel, type ″1″ in the *Option* field to select ″Load security domain variables″ and press **Enter**. You will see a panel that looks similar to the following:

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Load Security Domain Variables

 Specify the name of a data set containing the security domain variables,
 then press Enter to continue.

 IBM-supplied defaults are in ''


 Data set name:


 If this data set is not cataloged, specify the volume.

 Volume:
```

2. Fill in the name of the sequential data set you used to hold the security domain variable values and press **Enter**.

   Note: Ensure you enclose the data set name in single quotes.
   The security domain variables will load.

The security domain settings are loaded. You can display these variables, but not change them.

Choosing configuration data sets

This article leads you through the ″Allocate target data sets″ option in the Customization Dialog.

You must start the Customization Dialog and select the ″Federate an existing stand-alone application server node into an existing Network Deployment cell″ option.

Each option in the Customization Dialog saves customization jobs and files in a pair of customization data sets. While is it possible to reuse these data sets, it is safest to create separate data sets for each WebSphere Application Server for z/OS configuration. We recommend that you use the customization data set name prefix (sometimes referred to as ″config_hlq″) to indicate the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring.

Complete this task before generating the customization jobs and files.

1. On the main Dialog panel, type ″2″ in the *Option* field to select ″Allocate target data sets″.
2. Press Enter. **Result:** You see a panel that looks similar to the following:

```
-----    WebSphere Application Server for z/OS Customization      ------
Option  ===>

 Allocate Target Data Sets

 Specify a high level qualifier (HLQ) and press Enter to allocate the
 data sets to contain the generated jobs and instructions. You can
 specify multiple qualifiers (up to 39 characters).

 High level qualifier:                                        .CNTL
                                                              .DATA

 The Dialog will display data set allocation panels. You can make
```

```
changes to the default allocations, however you should not change
the DCB characteristics of the data sets.

   .CNTL  - a PDS with fixed block 80-byte records to
            contain customization jobs.

   .DATA  - a PDS with variable length data to contain
            other data produced by the Customization Dialog.
```

3. Fill in your chosen configuration data set name prefix value (config_hlq). If the data sets ″config_hlq.CNTL″ and ″config_hlq.DATA″ do not exist, you will be prompted for data set allocation information. If the data sets already exist, a message will inform you that they will be reused.

The data sets ″config_hlq.CNTL″ and ″config_hlq.DATA″ are allocated and will store customization jobs and files. These data set names will also be saved along with the customization variables.

Setting the customization variables: Federated application server node

This article describes how to complete the ″Define variables″ option for a WebSphere Application Server for z/OS federated node.

You must start the Customization Dialog and select the 'Create Network Deployment cells and nodes' option then the 'Federate an existing stand-alone application server node into an existing Network Deployment cell' option. Have the `Customization Dialog worksheet: Federated application server node` completed and at hand. See the *Installing your application serving environment* PDF for more information.

1. On the ″Federate stand-alone application server Node' panel, type ″3″ in the *Option* field to select ″Define variables″ and press **Enter**.
2. Fill in the federate node Define Variables panel using the following screen shots as your guides. When you are done with each, press **Enter**.

   Define Variables for Federate stand-alone application server Node (1 of 2)

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Define Variables for Federate stand-alone application server Node (1 of 2)
   Specify the following to customize your server, then press Enter
   to continue.

 WebSphere Application Server home directory:
     /WebSphere/V6R0
         / AppServer
 Deployment Manager Access
   Node host name...........:
   JMX SOAP port............:  8879
   Deployment manager security is enabled:  N
      User ID..............:  WSADMIN

 Include Apps..........:  Y
 Application server's ORB port..:  9810
 Node group name.......:  DefaultNodeGroup

 Node Agent Definitions
   Server name (short)...:  BBON001
   Server name (long)....:  nodeagent
   JMX SOAP connector port........:  9360
   Node Discovery port...........:  7272
   Node Multicast Discovery port..:  5000
   High availability manager communication port:  9354
```

```
 ORB listener host name...:  *
   ORB port.....................:  2809
   ORB SSL port.................:  0
```

Define Variables for Federate stand-alone application server Node (1 of 2)

```
------------  WebSphere Application Server for z/OS Customization     --------
Option  ===>

Define Variables for Federate stand-alone application server Node (2 of 2)
   Specify the following to customize your server, then press Enter
   to continue.

 Do you wish to federate service integration buses
   that exist on this node?" (Y/N):  N
```

Creating the customization jobs and files

You must select configuration data sets to use and complete the process of defining variables for this task. See "Choosing configuration data sets" on page 102 and "Setting the customization variables: Federated application server node" on page 103 for more information.

The Customization Dialog creates customization batch jobs and data files, based on the variable values you specified in the Dialog. The batch jobs and data sets will be written to the config_hlq.CNTL and config_hlq.DATA configuration data sets that you created with the 'Allocate target data sets' option.

1. Ensure the configuration data sets are allocated and not in use.

   Note: Editing a member in config_hlq.CNTL or config_hlq.DATA will cause this task to fail.

2. On the 'Federate stand-alone application server Node' panel, type "4" in the *Option* field to select "Generate customization jobs" and press **Enter**. You will have one of two results:

   - **Result A:** If all variables are defined correctly, you see the 'Specify Job Cards' panel, which looks similar to this:

     ```
     ------------  WebSphere Application Server for z/OS Customization     --------
     Option  ===>

     Generate Customization Jobs

      This portion of the Customization Dialog generates the jobs you must
      run after you complete this Dialog process. You must complete the
      customization process before you generate the jobs with this step.
      If you have not done this, please return to that step.

      Jobs and data files will get generated into data sets:
        'hlq.CNTL'
        'hlq.DATA'
      If you wish to generate customization jobs using other data sets, then
      exit from this panel and select option 1 (Allocate target data sets).

      All the jobs that will be tailored for you will need a job card.
      Please enter a valid job card for your installation below. The
      file tailoring process will update the jobname for you in all the
      generated jobs, so you need not be concerned with that portion of
      the job cards below. If continuations are needed, replace the
      comment cards with continuations.

      Specify the job cards, then press Enter to continue.
     ```

```
//jobname  JOB (ACCTNO,ROOM),'userid',CLASS=A,REGION=0M
//*
//*
//*
```

Note: Pay particular attention to the displayed target data sets. Make sure they are the ones you intend to use.

- **Result B:** If the variables are not defined correctly, you will see the 'Verification' panel. Decide whether the warnings or errors are serious enough to warrant returning to the "Define variables" option.

  Note: If the return code is "8" or greater, return to the "Define variables" option and fix the uncovered problems. If you saved the variables previously, be sure to re-save them after making any updates.

3. Fill in the job card information, according to your installation requirements. For each job, the Dialog generates a jobname and the "JOB" keyword to match the member name of the PDS, but you specify the rest.

   Note: If you need to run these jobs on a particular system in the sysplex (for example, JES2 MAS or JES3 complex), you should specify the necessary Scheduling Environment (SCHENV), JES2 JOBPARM, or JES3 //*MAIN statement at this time.

   Example of a job card entry:

   ```
   //jobname JOB 1234,USER1,NOTIFY=????,MSGCLASS=O,REGION=0M
   //*             USER=SYSADM1,PASSWORD=SYSADM1
   /*JOBPARM SYSAFF=SYSB
   ```

   Note: This example is useful for jobs that require a user ID other than that of the logged-on TSO user. (This is typically a user ID with UID=0.) In that case, you can just put a comma at the end of the first line, put in the correct user ID on the second line, then uncomment that second line.

   You may wish to use RACF SUBMIT authority to avoid having to keep passwords in your configuration data sets.

4. Fix any errors. If there are errors anywhere, you will see the 'Error' panel. Press PF3 to exit the error panel, then enter the correct panel to fix the errors. Then return to the "Generate Customization Jobs" option and pick up where you left off. If necessary, you can update the variables and rerun this option. The generation process will delete and re-tailor all the members.

   Note: Compress the configuration data sets before you rerun this option.

You are done when all the jobs are generated. You may then move ahead to viewing the generated jobs. See "Following the generated customization instructions: Federated application server node" for more information.

Following the generated customization instructions: Federated application server node

You must generate the customization jobs and files for this task.

The Customization Dialog creates a set of instructions for each customization task. Follow these instructions to tailor and customize a federated node on your system.

Note: Do not attempt to fix a typo or make a change by modifying the generated output. Many of the variables are used in multiple members of the target data sets, so, if you do not change them all, you will run into problems that are very difficult to diagnose.

1. On the 'Federate stand-alone application server Node' panel, type "5" in the *Option* field to select "View instructions" and press **Enter**. ISPF Browse will open and you will see the BBOANINS member of config_hlq.CNTL.
2. Read the instructions carefully, both to preview the customization process and to find any typographical or other errors you may have made while entering the customization variable values.
3. Follow the instructions as given. There are two ways to follow the directions:
   - Follow the instructions while remaining in ISPF Browse.
   - Record the data set name and member at the top of the screen and either print the instructions or use ISPF split screen and browse or edit the instructions while you follow them.
4. Fix any problems. If you encounter problems caused by your Customization Dialog values, modify your variables using the Dialog, regenerate the instructions, and restart the customization process.

   Note: Remember that you cannot generate new customization jobs while either configuration data set is open!

You are done with this customization task when you have successfully followed the generated instructions.

Sample generated instructions: Federated application server node

This article presents a sample of what the Customization Dialog's generated instructions may look like. This is a sample only--you must use the instructions generated from your own variables when configuring your system.

```
-----------------------------------------------
Instructions for customizing WebSphere for z/OS
for Federate stand-alone application server node.

The customization dialog has created jobs based on the information you
provided. These instructions tell you how to modify the operating
system and run the jobs to customize WebSphere for z/OS.

RULES:

1.  If you created the target data sets (*.CNTL and *.DATA) on another
    (driving) system, you must copy them to the target system and give
    them the same data set names.

2.  You must perform these instructions on your target system.

3.  Update TCP/IP by reserving the following ports for WebSphere for
    z/OS:

       SOAP JMX Connector port                     - 9360
       Node Discovery port                         - 7272
       Node Multicast Discovery Port               - 5000
       Node Agent's ORB port                       - 2809
       High Availability Manager Communication port - 9354
       application server's ORB Port               - 9810

    View member BBOTCPIA in
```

'DATASET.CNTL'.

Add the contents of this member to the PORT section of the file referenced by the DD statement for the TCP/IP profile in the TCP/IP start procedure. Cut and paste from this member into the data set used by your installation.

Note: The addNode process introduces a special utility server to the node. This utility server is called a nodeagent and exists to support administrative functions on the node. By default the nodeagent takes over ORB port 2809. Note on WebSphere z/OS the ORB port doubles as the INS CosNaming bootstrap port. By default, this port (2809) was assigned to the application server. Normally you want the nodeagent to be the INS CosNaming bootstrap point for the entire node, so that RMI/IIOP clients that do not override the INS CosNaming bootstrap defaults can locate within the namespace, EJBs installed on any server on that node. In order for the nodeagent to take over port 2809, the application server must be assigned a new ORB port. The default new ORB port for the application server is 9810. The nodeagent will take over a application server's ORB port if and only if the nodeagent's ORB port is equal to an application server's ORB port. You can specify the nodeagent's ORB port in the 'ORB port' field. You can specify the new ORB port for the application server in the 'Appplication Server's ORB Port' field.

ATTENTION: Skip this step if the ports are already defined in the TCP/IP profile.

4. You must first complete the customization of a stand-alone application server and the customization of a deployment manager server before starting these instructions. Also, ensure that the deployment manager server has been started before starting these instructions.

	User ID requirement: Authority to define a log stream.
Done:	NOTE: if you are not using logstream for transaction XA, you can skip this step.
By:	The process used to Federate an existing Application Server causes the user data of this application server to be changed. NOTE 2: STOP server prior to running BBOLOGSD. BBOLOGSD will fail if the server is active. Please run the following two jobs that were generated during stand-alone application server configuration: BBOLOGSD -- this will delete the logstreams that were allocated by BBOLOGSA earlier. BBOLOGSA -- this will create a new logstream with the application servers new user data. This is done so that IXGLOGR will not attempt to connect this new server to an old logstream which would produce an error.
BBOWADDN	User ID requirement: WSADMIN
Done:	Add the application server(s) associated with the stand-

	alone application server node to the deployment manager's cell.
By:	
	ATTENTION: Before you run this job (BBOWADDN) to federate the stand-alone application server node, you must have started the stand-alone Application server that you are federating at least once so the applyPTF step runs. Otherwise, the BBOWADDN job will fail with an error such as: java.lang.NullPointerException at com.ibm.ws.management.tools.NodeFederationUtility .createNDProductFile(NodeFederationUtility.java:1929)
	Note: Run this job only once for each node that you federate, regardless of how many application servers the node contains.
	RESULT: Upon successful completion of this job, you will see the following message in SYSPRINT: ADMU0003I: Node <nodename> has been successfully federated.
--------	Update the application server's WLM application environment.
Done:	
	ATTENTION: If you have already installed the WLM DAE APAR (APAR OW54622) on z/OS 1.4, or are running a higher level of z/OS, you may skip this step.
By:	
	If you do not have this WLM DAE APAR, you will need to update your static WLM definitions for the application server(s) on the stand-alone application server node you just federated into a Network Deployment cell. This is because the WLM APPLENV definitions for each server includes a reference to the server's cell short name in the ENV parameter that is part of the start parameters in APPLENV definition. The ENV parameter is encoded as follows: ENV=<CellShortName>.<NodeShortName>.<ServerShortName>
	After federating a stand-alone application server node into a Network Deployment cell, the cell identity of the stand-alone application server(s) changes from whatever it was to the cell identity of the Network Deployment cell. This means that you have to update the ENV parameter in the WLM APPLENV definition(s) for the server(s) on the newly federated node.
	To obtain the name(s) of the WLM application environments to be changed, access the WebSphere adminconsole application running on the deployment manager node by pointing a Web browser to the following address: http://1234: 9060/admin
	Select Servers, then application servers. For each application server, click on the server name, then scroll down to find the application server "short Name". Find and select the server's "Custom Properties". In the Custom Properties list, you will find a property named 'ClusterTransitionName', which is also the server's WLM application environment. Use the IWMARIN0

		application under TSO to modify these WLM application environment(s) so that the ENV= field begins with the deployment manager cell name instead of the stand-alone server cell name. Install and apply the updated WLM service definition before continuing.
--------	Start the node agent server	
	Note: The node agent is automatically started by the node federation process. This step is information, for starting the node agent at other times. If your application server proc name is XXXXXXXX and your application server node name (short) is YYYYYYYY, issue the MVS command	
Done: By:	START XXXXXXXX,JOBNAME=BBON001, ENV=.YYYYYYYY.BBON001 This command starts the node agent server. RESULT: The following message appears on the console and in the job log of BBON001. BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR z/OS CONTROL PROCESS BBON001.	
--------	Start the application server (optional)	
Done: By:	Note: You must start the node agent before you start the application server, if the node agent has not already been started Example: If your application server proc name is XXXXXXXX, your application server node name (short) is YYYYYYYY, and your server short name is BBOS001, issue the MVS command START XXXXXXXX,JOBNAME=BBOS001, ENV=.YYYYYYYY.BBOS001 This command starts the application server. RESULT: The following message appears on the console and in the job log of BBOS001. BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR z/OS CONTROL PROCESS BBOS001	

Working with your new federated server node

Once you have successfully federated an application server node:

- Check the default host alias list and all other cell-level documents to see if any need to be added in support of the application(s) and application server(s) on the newly federated node. Cell-level documents are **NOT** automatically updated by the federation process.
- Remove the location service daemon port definitions for the stand-alone application server cell, since these are not used after federation.

Once these tasks are accomplished, a federated application server node is just like any other application server node. The primary difference is that it already has an application server, and applications, if they were federated as well. See "Working with your new server" on page 64 for further information.

Configuring ports

This topic discusses configuring ports, particularly in coexistence scenarios.

1. Review the port number settings, especially when you are planning to coexist.

   See the *Installing your application serving environment* PDF for reference information for identifying port numbers used by present and past product versions.

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 *Installing your application serving environment* PDF.

   After installation, edit the *profiles_install_root*/*profile_name*/config/cells/*cell_name*/nodes/*node_name*/serverindex.xml file to change the port settings, or use scripting to change the values. See the *Administering applications and their environment* PDF for more information.

Communicating with Web servers

The WebSphere Application Server works with a Web server to route requests for dynamic content, such as servlets, from Web applications. The Web servers are necessary for directing traffic from browsers to the applications that run in WebSphere Application Server. The Web server plug-in uses the XML configuration file to determine whether a request is from the Web server or the Application Server.

- Install your Web server if it is not already installed.

  See the installation information provided with your Web server.

- Ensure that your Web server is configured to perform operations required by Web applications, such as GET and POST. Typically, this involves setting a directive in the Web server configuration file (such as the httpd.conf file for an IBM HTTP Server). Refer to the Web server documentation for instructions. If an operation is not enabled when a servlet or JSP file requiring the operation is accessed, an error message displays, such as this one from the IBM HTTP Server:

  ```
  IMW0093E Method POST is disabled on this server.
  ```

- Make sure the appropriate plug-in file has been installed on your Web server and the configureWeb_server_name script has been run to create and configure the Web server definition for this Web server.

  If you are using the IBM HTTP Server provided with the z/OS base operating system, see Installing the WebSphere HTTP Plug-in for z/OS.

  If you are using a distributed platform Web server with a WebSphere Application Server running on a z/OS platform, FTP the plug-in to the Web server and use the Plug-in installation wizard to install the appropriate plug-in file to your Web server.

The WebSphere HTTP Plug-in for z/OS and the Web server plug-ins for distributed platform Web servers are provided with the WebSphere Application Server product that runs on a z/OS platform. The *Installing your application serving*

environment PDF describes how to install these plug-ins and create corresponding Web server definitions to enable the plug-ins to communicate with an application server running on a z/OS platform.

The following steps are performed during the plug-in installation process. See the Plug-in Installation Roadmap for additional information.

1. A Web server definition is created.

   You can also use either use the administrative console or use the ConfigureWebServerDefintion.jacl script to create a Web server definition. If you use the administrative console:

   a. Select the node that was created in the preceding step, and in the Server name field, enter the local name of the Web server for which you are creating a Web server definition.

   b. Use the wizard to complete the Web server definition.

2. An application or modules are mapped to a Web server.

   If an application that you want to use with this Web server is already installed, the application is automatically mapped to the Web server. If the application is not installed, select this Web server during the Map modules to servers step of the application installation process.

3. Master repository is updated and saved.

When you install a plug-in, the configuration file for that plug-in is automatically created. You can change or fine tune the default settings for the properties in this configuration file. If change any of the settings, you must regenerate the file before your changes take affect.

Generating or regenerating the configuration file might take a while to complete. After it finishes, all objects in the administrative cell use their newest settings, which the Web server can access. If the Application Server is on the same physical machine as the Web server, the regeneration usually takes about 30 to 60 seconds to complete. The regeneration takes longer if they are not both on the same machine.

1. Use the administrative console to change the settings in the plug-in configuration file.

   When setting up your Web server plug-in, you must decide whether or not to have the configuration automatically generated in response to a configuration change. When the Web server plug-in configuration service is enabled and any of the following conditions occur, the plug-in configuration file is automatically generated:

   - When the Web server is created or saved.
   - When an application is installed.
   - When an application is uninstalled.
   - When the virtual host definition is updated

   You can either use the administrative console, or issue the GenPluginCfg command to regenerate your `plugin-cfg.xml` file. To use the administrative console:

   a. Select **Servers > Web Servers >** *webserver* **> plug-in properties**.

   b. Select **Automatically generate plug-in configuration file** or click on one or more of the following topics to manually configure the `plugin-cfg.xml` file:
      - Caching
      - Request and response
      - Request routing

- Service

Web server plug-in configuration properties maps each property to one of these topics.

Note: It is recommended that you do not manually update the `plugin-cfg.xml` file. Any manual updates you make for a given Web server are overridden whenever the `plugin-cfg.xml` file for that Web server is regenerated.

c. Click **OK**.

d. You might need to stop the application server and then start the application server again to enable the Web server to locate the `plugin-cfg.xml` file.

2. 6.1 + **Optional:** Edit the plug-in configuration file. You should not have to edit the configuration file. However, if you do, remember that the file is in ASCII format (ISO-98859-1). Issue the following command to convert the file to EBCDIC format:

```
> iconv  -f ISO8859-1 -t IBM-1047 plugin-cfg.xml.ASCII > plugin-cfg.xml.EBCDIC
```

Edit the file, and then issue the following command to convert it back to ASCII format:

```
> iconv  -f IBM-1047 -t ISO8859-1  plugin-cfg.xml.EBCDIC > plugin-cfg.xml.ASCII
```

Remember that any manual changes you make to the file will be overwritten the next time the file is regenerated.

3. **Optional:** If you want to enable the Web server plug-in to use private headers, define an SSL configuration repertoire that defines a trust file. Then in the administrative console, select **Application servers > server1 > Web Container Settings > Web Container Transport Chains >** *transport_chain* > **SSL Inbound Channel (SSL_2)** and specify this repertoire for that transport chain. If you try to use private headers without setting up an SSL configuration repertoire that does not include a trust file definition, the private headers will be ignored. If the private headers are ignored, the application server might not locate the requested application.

   After you enable the use of private headers, the transport chain's SSL inbound channel trusts all private headers it receives. Therefore, you must ensure that all paths to the transport chain's SSL inbound channel are trusted.

4. Propagate the plug-in configuration. The plug-in configuration file (plugin-cfg.xml) is automatically propagated to the Web server if the Web server plug-in configuration service is enabled, and one of the following is true:
   - The Web server is a local Web server. (It are located on the same machine as an application server.)
   - The Web server is a remote IBM HTTP Server Version 6.0 that has a running IBM HTTP Server Administrative server.

   If neither of these conditions is true, the plugin-cfg.xml file must be manually copied to the remote Web server's installation location.

   The remote Web server installation location is the location you specified when you created the node for this Web server.

The configuration is complete. To activate the configuration, stop and restart the Web server. If you encounter problems restarting your Web server, check the http_plugin.log file for information on what portion of the plugin-cfg.xml file contains an error. The log file states the line number on which the error occurred along with other details that might help you diagnose why the Web server did not start. You can then use the administrative console to update the plugin-cfg.xml file.

If applications are infrequently installed or uninstalled, which is usually the situation in a production environment, or if you can tolerate the performance impact of generating and distributing the plug-in configuration file each time any of the previously listed actions occur, you should consider enabling this service.

If you are making a series of simultaneous changes, like installing numerous applications, you might want the configuration service disabled until after you make the last change. The Web server plug-in configuration service is enabled by default. To disable this service, in the administrative console click elect **Servers > Application Servers >** *server_name* **> Administration Services >Web server plug-in configuration service** and then unselect the Enable automated Web server configuration processing option.

Tip: If your installation uses a firewall, make sure you configure the Web server plug-in to use a port that has been opened. (See your security administrator for information on how to obtain an open port.

Setting up the administrative architecture

After you set up the Network Deployment environment, 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.

1. Use the settings page for an administrative service to configure administrative services.
2. Configure cells.
3. Configure deployment managers.
4. Manage nodes.
5. Manage node agents.
6. Manage node groups.
7. Configure remote file services.
8. Configure location service daemons on the z/OS system.

Cells

Cells are logical groupings of one or more nodes in a WebSphere Application Server distributed network.

A cell is a configuration concept, a way for administrators to logically associate nodes with one another. Administrators define the nodes that make up a cell, according to the specific criteria that make sense in their organizational environments.

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.

Configuring cells

When you created a deployment manager profile, a cell was created. A cell provides a way to group one or more nodes of your Network Deployment product. You probably will not need to re-configure the cell. To view information about and manage a cell, use the settings page for a cell.

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 drop-down list for **Cell Discovery Protocol** on the settings page for the cell. It is unlikely that you will need to change the cell's protocol configuration from TCP.
3. Click **Custom Properties** and define any name-value pairs needed by your deployment manager.
4. When you installed the WebSphere Application Server Network Deployment product, a node was 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.

   Note: 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 details, see "IP version considerations for cells."

IP version considerations for cells

Internet Protocol Version 4 is no longer viable for many businesses. Because it is based on 32-bit architecture, there is a growing shortage of Internet Protocol Version 4 (IPv4) addresses. Internet Protocol Version 6 (IPv6) is based on 128-bit architecture, which allows a far greater number of addresses to be available for use over the Internet.

In response, WebSphere Application Server Version 6 now includes support for IPv6, in addition to continued support for IPv4. This means that nodes running WepSphere Application Server Version 6 can use IPv6. (However, note that nodes running WebSphere Application Server Version 5.x cannot use IPv6.)

WebSphere Application Server Version 6 supports a *dual mode* environment in which you can have older legacy applications running on IPv4 and IPv6-enabled applications running on IPv6. Note, however, that there are restrictions on using IPv4 and IPv6 in the same cell. This article documents those restrictions as well as outlines the ways in which you can set up your cells, depending on the version of IP that you will be using.

Note: IPv6 is not supported on native transports. If you need this function, you must configure a channel chain.

From an IP perspective, you must adhere to one of the following scenarios:

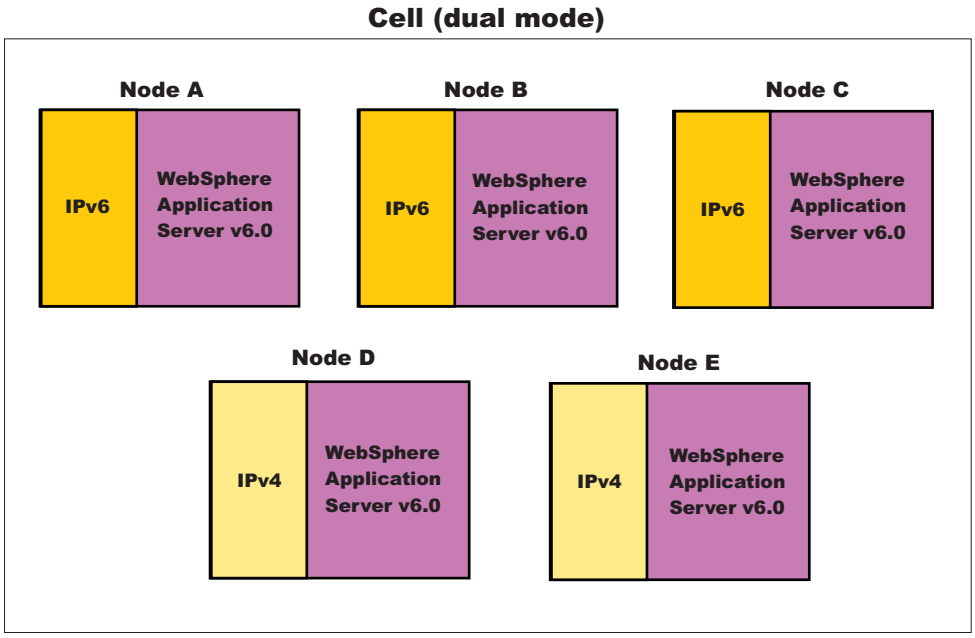
Dual mode cell

In a dual mode cell, mixed IPv4 and IPv6 communications are supported. By default, a cell is set to dual mode when it is created. Note, however, that only nodes running WebSphere Application Server Version 6 are valid in a dual mode cell.

IPv4 and IPv6 nodes cannot communicate with each other, so the purpose of the dual mode cell is to enable this communication, thereby allowing you to use your existing applications, running over IPv4, with newer applications that have been enabled for IPv6.

The following illustration shows a dual mode cell:

Dual mode cell

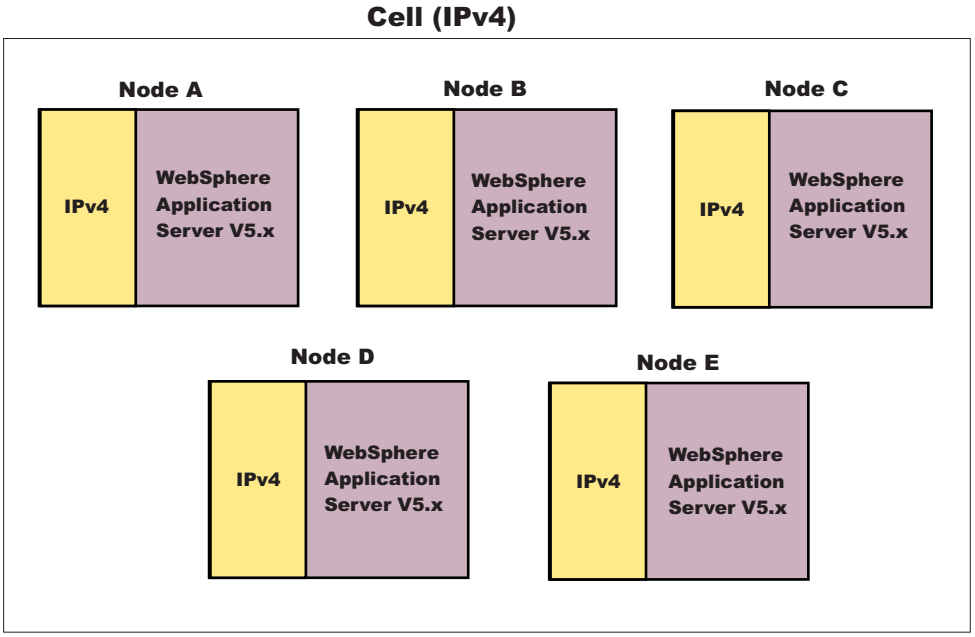


IPv4-only cell

In an IPv4-only cell, all nodes must:

- Use IPv4
- Run WebSphere Application Server Version 5.x
- Have host names defined as strings or 32-bit numerical addresses.

IPv4-only cell



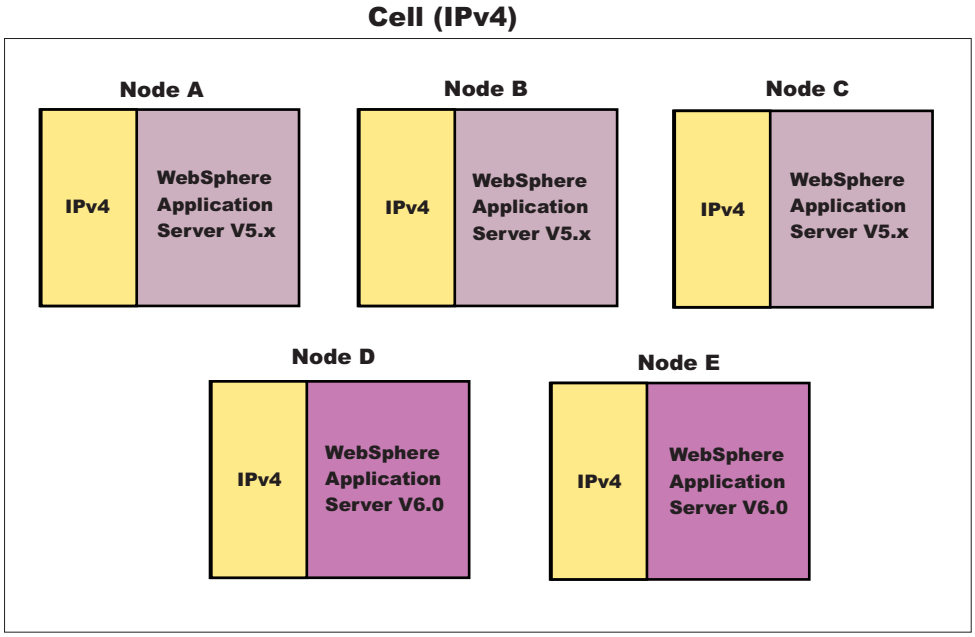
It is important to note that, by default, a cell is set to dual mode. However, in order to run in an IPv4-only environment, you will need to explicitly set the cell to IPv4. See the section on JVM settings, in this article, for more information.

Note: If you want to run a combination of WebSphere Application Server Version 5.x and WebSphere Application Server Version 6.0 nodes over IPv4, see the section on setting up a *mixed node cell*, below.

Mixed node cell

A mixed node cell consists of some nodes running WebSphere Application Server Version 5.x and other nodes running WebSphere Application Server Version 6. In a mixed node cell, all nodes must use IPv4. When defining a node that will be used in a mixed node cell, you must specify the host name as a string or as a 32-bit numerical address, regardless of whether the node is running WebSphere Application Server Version 5.x or WebSphere Application Server Version 6. 128-bit numerical addresses may not be specified.

Mixed node cell



In a mixed node cell, even though the WebSphere Application Server Version 6 nodes will be configured to use IPv4, the operating system running on them can still support both IPv4 and IPv6. This is true as long as the WebSphere Application Server Version 6 nodes are configured with string-based host names or 32-bit numerical addresses.

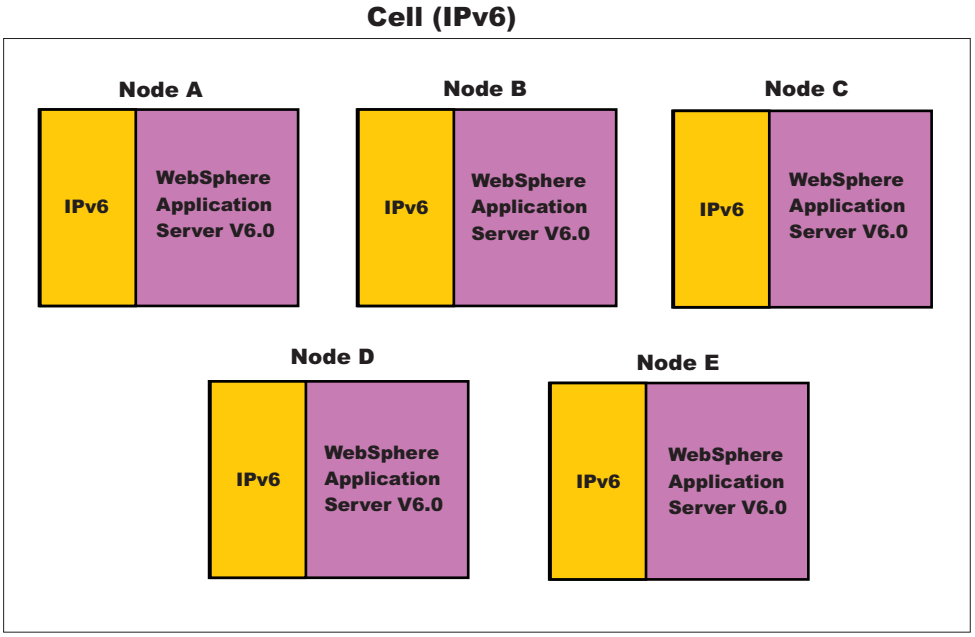
Note also, that you can only add Version 5.x nodes into a mixed node cell through migration. You first need to migrate from a Version 5.x Deployment Manager to a Version 6.0 Deployment Manager, and then either keep the Version 5.x nodes or migrate them to Version 6.0 nodes.

IPv6-only cell

In an IPv6-only cell, all nodes must:

- Use IPv6
- Run WebSphere Application Server Version 6
- Have host names defined as strings or 128-bit numerical addresses.

IPv6-only cell



Specifying host names

During the installation of WebSphere Application Server, you are asked to provide the host name or IP address of the machine on which the installation is being carried out in the *Host Name or IP address* field. The host name or IP address that you specify is used to advertise this installation to all other WebSphere Application Server installations in the cell configurations. All nodes in the cell will use the host names or IP addresses that are defined in this way to reach each other. In general, it is best to always use a host name to identify a WebSphere Application Server installation. By using a host name, you will not have to be concerned about which IP address is being used (32-bit versus 128-bit), whether it runs on IPv4 or IPv6, and so on. As long as the DNS service is properly configured, the nodes should all be able to work together.

However, if you prefer, you can control which IP stack or address is used. To do this, enter the specific IP address (32-bit for IPv4 or 128-bit for IPv6) into the *Host Name or IP Address* field. This installation will then be identified with this IP address and other WebSphere Application Server nodes will use this IP address to communicate with this node.

When specifying IPv6 addresses, it is good practice to always surround them with protective square brackets. For example, `[fe80::202:57ff:fec4:2334]`. The reason for this is that in system internal processing, IP addresses are often combined with port numbers in the form of `<IP address>:<port number>` , and the colons in IPv6 addresses could be confusing in such circumstances. Note that you cannot use IPv6 addresses that are surrounded by square brackets within the administrative console or the install wizard.

Note that 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.

Note that in scripting, the square brackets might have special meaning, depending on the language binding used (for example, Jacl). You can work around this problem by using a special escape character in front of the opening and closing brackets. Using the Jacl binding, for example, the same IPv6 address cited earlier can be entered as `\[fe80::202:57ff:fec4:2334\]`

Note: While you cannot use square brackets with IPv6 addresses within the administrative console, you must use square brackets to specify an IPv6 address as part of the adminstrative console's URL in a browser. This allows the browser to distinguish the IPv6 address from the port value.

Multicast configuration

WebSphere Application Server uses multicast broadcasting at the node level to allow a node agent to discover the managed processes in the node. IPv4 and IPv6 addresses are not compatible. Therefore, to allow a WebSphere Application Server node installation to run *out-of-the-box*, both IPv4 and IPv6 multicast addresses are initially defined in the node agent configuration, and when a node agent starts, both addresses will be tried in sequence. It is a good idea to delete either NODE_MULTICAST_DISCOVERY_ADDRESS or NODE_IPV6_MULTICAST_DISCOVERY_ADDRESS after installation. By that time, you should know whether the node is running IPv4 or IPv6, so by limiting multicast discovery to the known protocol, the node agent runs more efficiently.

To delete one of the multicast ports using the Administrative Console, do the following:

1. Click **System Administration --> Node Agents**.
2. Select the node agent.
3. On the next panel, under Additional Properties, select **Ports**. The next panel shows a list of existing ports.
4. Select either NODE_MULTICAST_DISCOVERY_ADDRESS (to delete IPv4) or NODE_IPV6_MULTICAST_DISCOVERY_ADDRESS (to delete IPv6).
5. Click **Delete**.

JVM settings

Two system properties are related to IPv6 support. They are:

- java.net.preferIPv4Stack=<true | false>
- java.net.preferIPv6Addresses=<true | false>

In general, setting these properties is not recommended. In particular, setting java.net.perferIPv4Stack to true will disable the dual mode support in the JVM which might, in turn, disrupt normal WebSphere Application Server functions. Therefore, it is important to understand the full implications if you are contemplating using this setting.

Cell settings

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.

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 cell follows to retrieve information from a network.

Select one of these protocol options:
UDP User Datagram Protocol (UDP)
TCP Transmission Control Protocol (TCP)

Default TCP

Deployment managers
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.

A deployment manager hosts the administrative console. A deployment manager provides a single, central point of administrative control for all elements of the entire WebSphere Application Server distributed cell. Each cell contains one deployment manager.

Configuring deployment managers

When you created a deployment manager profile, a deployment manager was created. A deployment manager provides a single, central point of administrative control for all elements in a WebSphere Application Server distributed cell. You can run the deployment manager with its default settings. However, you can follow this task to change the deployment manager configuration settings such as the ports the process uses, custom services, logging and tracing settings, and so on. To view information about and manage a deployment manager, use the settings page for a deployment manager.

1. Access the settings page for a deployment manager. Click **System Administration > Deployment Manager** from the navigation tree of the administrative console.
2. Configure the deployment manager as desired by clicking on a property such as **Custom Services** and specifying settings on the resulting pages.
3. If you specify the server short name as 8 characters, follow the directions toconvert the default 7 character short name to 8 characters.

Deployment manager settings
Use this page to stop the deployment manager from running, and to link to other pages which you can use to define additional properties for the deployment manager. A deployment manager provides a single, central point of administrative control for all elements of the entire 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, SAF (for example, RACF), started task control, and others.

The name is 1-8 characters, alpha numeric or national language. It cannot start with a numeric.

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

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

Process ID
Specifies a string identifying 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 it is not running.

Data type	String
Default	Started

Converting a 7 character server short name to 8 characters for the deployment manager

By default, WebSphere Application Server for z/OS assumes you will be 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 by performing the tasks below. **Before doing this, it is important to consider the following:**

- 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 WebSphere Application Server for z/OS 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.

  Note that this consideration does not apply to you 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 *Securing your applications and their environment* PDF for more information.

1. Change the 7 character server short name to the 8 character name you wish to use:
   a. Navigate to **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 servant's start command arguments to use the new 8-character name. Note that if you are reconfiguring a node agent, you can skip this step because it does not have an associated servant process.
   a. Navigate to **System administration > Deployment manager > Server infrastructure > Java and process management > Process definition > *Process type***.
   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**.

Node

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. Node names usually are identical to the host name for the computer.

Nodes in the network deployment topology can be managed or unmanaged. Two types of managed nodes exist while one type of unmanaged node exists.

One type of managed node has a node agent which manages all servers on a node, whether the servers are WebSphere Application Servers, Java Message Service (JMS) servers (on Version 5 nodes only), Web servers, or generic servers. The node agent represents the node in the management cell. A deployment manager manages this type of managed node. The other type of managed node has no node agent. This type of managed node is defined on a standalone Application Server. The deployment manager cannot manage this standalone Application Server. An standalone Application Server can be federated. When it is federated, a node agent is automatically created. The node becomes a managed node in the cell. The deployment manager manages this node.

An unmanaged node does not have a node agent to manage its servers. Unmanaged nodes can have server definitions such as Web servers in the WebSphere Application Server topology. Unmanaged nodes can never be federated. That is, a node agent can never be added to an unmanaged node.

A supported Web server can be on a managed node or an unmanaged node. You can define only one Web server to a standalone WebSphere Application Server node. This Web server is defined on an unmanaged node. You can define Web servers to the deployment manager. These Web servers can be defined on managed or unmanaged nodes.

WebSphere Application Server supports basic administrative functions for all supported Web servers. For example, 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 created 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 Server supports 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.

You can add managed and unmanaged nodes to a network deployment cell in one of the following ways:

- Administrative console
- Command line (managed nodes only)
- Administrative script
- Java program

Each of these methods for adding a managed node to a 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 node group of DefaultNodeGroup 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 or accept the default, the node group membership rules must be satisfied. If the node 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.

Managing nodes

This article describes how to 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.

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.

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

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.**
  1. Go to the Nodes page and click **Add Node**. Choose whether you want to add a managed or unmanaged node, and click **Next**.

2. For a managed node, verify that an application server is running on the remote host for the node that you are adding. On the Add Node page, specify a host name, connector type, and port for the application server at the node you are adding.
3. For a managed node, do one of the following sets of actions in the table:

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 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**.
The distributed 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	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.
4. For managed nodes, another administrative console panel is displayed if the node to federate is on a Windows operating system. Specify on the panel 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.
5. 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.

Both Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) are now supported by WebSphere Application Server, but restrictions do apply when using both IPv4 and IPv6 in the same cell. When you add a node to a cell, the format in which you specify the name is based on the version of IP that the node is using. For details, see IP version considerations for cells.

- **Select the discovery protocol.**

  If the discovery protocol that a node uses is not appropriate for the node, select the appropriate protocol. On the Nodes page, click the node to access the Settings for the node. Select a value for **Discovery Protocol**. 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 since 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 Settings for the node, click **Custom Properties**.
  3. On the Property collection page, click **New**.
  4. On the Settings page for a property instance, specify a name-value pair and a description for the property, and click **OK**.

- Synchronize the node configuration.

  If you added a managed node or changed a managed node's configuration, synchronize the node's configuration. On the Node Agents page, ensure that the node agent for the node is running. Then, on the Nodes page, put a check mark in the check box beside the node whose configuration files you want to synchronize and click **Synchronize** or **Full Resynchronize**.

  Clicking either button 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 is important if automatic configuration synchronization is disabled, or if the synchronization interval is set to a long time, and a configuration change has been made to the cell repository that needs to be replicated 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. So 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 will be 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, put a check mark in the check box beside the managed node whose servers you want to stop running and click **Stop**.

- **Remove a node.**

  On the Nodes page, put a check mark in the check box beside the node 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 .

Node collection

Use this page to manage nodes in the WebSphere Application Server environment. Nodes group managed servers.

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.

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 name is based on the version of IP the node will be using.

Version

Specifies the product version 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 unknown.

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 the state of the node.

The state can be**Started**, **Stopped**, **Unavailable**, **Unknown**, **Partial Start**, **Partial Stop**, **Not applicable**, **Synchronized**, or **Not synchronized**.

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

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 name is based on the version of IP the node will be using.

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 read only. It is defined during installation and customization.

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.

Host name

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

Date type String
Default None

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 nodes
A managed node is a node with an Application Server and a node agent that belongs to a cell. Use this page to add a managed node to a cell.

To view this administrative console page, click **System Administration > Nodes > Add 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

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.
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.

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.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.

This property applies to the z/OS operating system only.

com.ibm.websphere.deployed.features
A list of features that extends a profile. An example of a feature is an administrative console plug-in.

Node group

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.

Nodes that you organize 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 Application Server administrator. However, a node must be a member of a node group. Initially, all Application Server nodes are members of the default node group named DefaultNodeGroup.

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

On the z/OS platform, 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.

A node on a distributed platform and a node on a z/OS platform cannot 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.

Node group membership rules

Nodes can be members of node groups if they meet certain requirements.

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. If the node is on the z/OS platform, the node group must be a sysplex node group.
- Nodes on a distributed platform and nodes on the z/OS platform must be members of different node groups.
- Nodes on the z/OS platform that are in different sysplexes must be members of different groups.

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.

Examples: Using node groups

Use node groups to define groups of nodes capable of hosting members of the same cluster. An application deployed to a cluster must be capable of running on any of the cluster members. This means that the node that hosts each of the cluster members must be configured with software and settings necessary to support running of 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 1 to 8.
- Each node is a managed node, which means that each node is configured with an Application Server.

- Nodes 6, 7, and 8 are additionally configured as WebSphere Business Integration Server Foundation nodes.
- All nodes are either z/OS system nodes from the same sysplex, or distributed platform nodes.
- By default, all the nodes are in the default node group, DefaultNodeGroup.

Applications that exploit WebSphere Business Integration Server Foundation functions can run successfully only on nodes 6, 7, and 8. Therefore, clusters that host these applications can be formed only on nodes 6, 7, and 8. To define a clustering policy that guides users of your WebSphere cell into building clusters that can only span predetermined nodes, create an additional node group called WBINodeGroup, for example. Add to the node group nodes 6, 7, and 8. 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 1 to 6.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes 1 to 4 are on distributed platforms.
- Nodes 5 and 6 are nodes on the z/OS operating system and are in sysplex PLEX1.
- The deployment manager is on a distributed platform node.
- Nodes 1 to 4 are members of the DefaultNodeGroup node group by default.
- You created empty node group PLEX1NodeGroup 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. You did this because nodes on the z/OS operating system cannot be in the same node group as the distributed platform nodes.

Applications that exploit z/OS functions in the PLEX1 sysplex can run successfully only on nodes 5 and 6. Therefore, clusters that host these applications can be formed only on nodes 5 and 6. The required separation of distributed 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 only span 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.

Managing node groups

Read about Nodes groups if you are unfamiliar with them.

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.

- 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.

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.

On the z/OS platform, 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.

A node on a distributed platform and a node on a z/OS platform cannot 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.

Data type	String
Maximum length	64 characters

The name must contain alphanumeric or national language characters and cannot start with a number.

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.

Managing node group members

Read about Nodes groups and Node group membership rules if you are unfamiliar with them.

Group nodes that meet your application requirements into a node group.

- 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 bread crumb 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 bread crumb trail.

Node group member collection

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

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

To view this administrative console page, click **System Administration > Node groups >** *node group name* **> nodes > Node group members**.

Name

Specifies the name of a node group member.

Description

Specifies a description that you previously defined for the node group member when you created the 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* **> nodes > 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 cannot start with a number.

Administration service settings

Use this page to view and change the configuration for an administration service.

To view this administrative console page, click **Servers > Application Servers >***server_name* **> Administration > Administration Services**

Preferred Connector

Specifies the preferred JMX Connector type. Available options, such as SOAPConnector or RMIConnector, are defined using the JMX Connectors page.

Data type	String
Default	SOAP

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 > 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 > Application Servers >***server_name* **> Administration > Administration Services > Extension MBean Providers >***provider_library_name*

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

Name
The name used to identify the Extension MBean provider library.

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 > Application Servers >***server name* **> Administration > Administration Services > Extension MBean Providers >** *provider library name***> Extension MBean**

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 > Application Servers >***server name***> Administration > Administration Services > Extension MBean Providers >** *provider library name***> Extension MBean >***Extension MBean name*

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

Java Management Extensions connector properties

A Java Management Extensions (JMX) connector can either be a Remote Method Invocation (RMI) connector or a Simple Object Access Protocol (SOAP) connector.

Depending on the property, you can specify or set a property in the administrative console, the wsadmin tool, Application Server commands, scripts 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.

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. For specific information on how to code the JMX connector properties for a custom Java administrative client program, see the Java API documentation for Application Server.

For the administrative console, this article specifies the coding of the particular setting or property. Coding of properties in the `soap.client.props` file that are specific to JMX connectors is specified. These properties begin with com.ibm.SOAP. Other properties in the `soap.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, is specified.

Each profile has a property file at *`installation root`*`/profiles/`*`profile name`*`/properties/soap.client.props`. These property files allow you to set different properties, including security and timeout properties. These properties are the default for all administrative connections that use the SOAP JMX connector between processes executing in a particular profile. For instance, the wsadmin program executing under a particular profile uses the property values from that file for the SOAP connector behavior unless the properties are overridden by some other programmatic means.

To view the JMX connector custom properties administrative console panel that goes with this article, click one of the following paths:

- **Servers -> Application servers ->***server name* **-> Server Infrastructure -> Administration -> Administration Services -> Additional properties -> JMX Connectors->***connector type* **-> Additional Properties -> Custom properties**
- **System administration -> Deployment manager ->Additional Properties -> Administration Services -> Additional Properties -> JMX Connectors->***connector type***-> Additional Properties -> Custom properties**
- **System administration -> Node agents ->***node agent name* **-> Additional Properties -> Administration Services -> Additional Properties -> JMX Connectors->***connector type***-> Additional Properties -> Custom properties**

SOAP connector properties

This section discusses JMX connector properties that pertain to SOAP connectors.

SOAP Request timeout

Specifies the SOAP client 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 level of 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.

You can set the property by using one of the following options:

- Scripts 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.

Configuration URL

Specifies the Universal Resource Locator (URL) of the `soap.client.props` file. Specify the configuration 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. The property is AdminClient.CONNECTOR_SOAP_CONFIG.

Security context provider

Specifies the Secure Sockets Layer (SSL) implementation to use between the Application Server and the SOAP client. You can specify either IBM Java Secure Sockets Extension (IBMJSSE) or IBM Java Secure Sockets Extension that has undergone Federal Information Processing Standards certification (IBMJSSEFIPS). See the *Securing your applications and their environment* PDF for more information.

You can set the property by using the `soap.client.props` file.

Property	com.ibm.ssl.contextProvider
Data type	String

Valid Values	`IBMJSSE`
	`IBMJSSEFIPS`
	`IBMJSSE2`

Default	IBMJSSE2

Secure Sockets Layer (SSL) security

Use this property to enable SSL security between Application Server and the SOAP client. You can set the property by using one of the following options:

- Scripts 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. The property is AdminClient.CONNECTOR_SECURITY_ENABLED.

SOAP and RMI connector properties

This section discusses JMX connector properties that pertain to both SOAP connectors and RMI connectors.

Connector type

Specify a connector type of SOAP or RMI, depending on whether Application Server connects to a SOAP server or an RMI 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	Type
Data type	String
Valid values	`SOAPConnector` `RMIConnector`
Default	SOAP

- A Java administrative client. The property is AdminClient.CONNECTOR_TYPE. Specify by using the AdminClient.CONNECTOR_TYPE_RMI or the AdminClient.CONNECTOR_TYPE_SOAP constants.

Host

Use the host property to specify the host name or the IP address of the server to which Application Server connects. The server can be a SOAP server or an RMI 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	host
Data type	String
Valid values	Host name or IP address
Default	None

- A Java administrative client. The property is AdminClient.CONNECTOR_HOST.

Port

Use the port property to specify the port number of the server to which Application Server connects. The server can be a SOAP server or an RMI 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. The property is AdminClient.CONNECTOR_PORT.

User name

Specifies the user name that Application Server uses to access the SOAP server or the RMI 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.

Property	com.ibm.SOAP.loginUserid
Data type	String
Valid value	The value must match the global SSL settings for SOAP or RMI.
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 or RMI.
Default	None

- A Java administrative client. The property is AdminClient.USERNAME.

Password

Specifies the password that Application Server uses to access the SOAP server or the RMI 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.

Property	com.ibm.SOAP.loginPassword
Data type	String
Valid values	The value must match the global SSL settings for SOAP or RMI.
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 or RMI.
Default	None

- A Java administrative client. The property is AdminClient.PASSWORD.

Java Management Extensions connectors

Use this page to view and change the configuration for Java Management Extensions (JMX) connectors.

To view this administrative console page, click one of the following paths:
- **Servers > Application Servers >***server_name* **> Administration > Administration Services > JMX Connectors**
- **Servers > JMS 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 Simple Object Access Protocol (SOAP) connector and an appropriate port number. You can also use the Remote Method Invocation (RMI) 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, both the SOAP connector and the RMI 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 Enum

Default SOAPConnector

Range

SOAPConnector
For JMX connections using Simple Object Access Protocol (SOAP).

RMIConnector
For JMX connections using Remote Method Invocation (RMI).

JMX connector settings
Use this page to view the configuration for a Java Management Extensions (JMX) connector.

To view this administrative console page, click one of the following paths:
- **Servers > Application Servers >***server_name* **> Administration > Administration Services > JMX Connectors >***connector_type*
- **Servers > JMS Servers >***server_name* **> Administration > Administration Services > JMX Connectors >***connector_type*

On the z/OS platform, both the SOAP connector and the RMI 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 Enum

Default SOAPConnector

Range

SOAPConnector
For JMX connections using Simple Object Access Protocol (SOAP).

RMIConnector
For JMX connections using Remote Method Invocation (RMI).

Repository service custom properties on z/OS systems

This topic specifies the coding of custom properties for the administrative repository service on the z/OS system.

To view the Repository service custom properties on the administrative console panel, click **System administration > node agents > nodeagent > Administration services > Repository service > Custom properties**.

You can code any of the following custom properties for the repository service:

Disable thread pool

Disables the use of a thread pool when the transformer code runs. The transformer code generates the `was.env` file, the `control.jvm.options` file, the `servant.jvm.options` file, and the `trace.dat` file.

Specify the property and the value as a name-value pair on the Repository service custom properties panel of the administrative console:

Property	com.ibm.websphere.management.zos.transform.threadPoolDisable
Values	true or false
Default	false

Thread pool count

Specifies the number of threads that the transformer code can use when generating the `was.env` file, the `control.jvm.options` file, the `servant.jvm.options` file, and the `trace.dat` file. If the thread pool option is disabled, this value has no effect.

Specify the property and the value as a name-value pair on the Repository service custom properties panel of the administrative console:

Property	com.ibm.websphere.management.zos.transform.threadPoolCount
Data type	Integer
Value range	1 - 30
Default	20

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 > Application Servers >***server_name* **> 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

Node agents

Node agents are administrative agents that route administrative requests to servers.

A node agent is a server that runs on every host computer system that participates in the WebSphere Application Server Network Deployment product. It 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.

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. A node agent is created automatically when a node is added to a cell.

1. View information about a node agent. Use the Node Agents page. 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 on the node agent's name under **Name**.

   Note: 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 a node is added to a cell, the format in which the name is specified is based on the version of IP the node will be using. For details, see "IP version considerations for cells" on page 114.

2. Stop and then restart all application servers on the node managed by the node agent. On the Node Agents page, place a check mark in the check box beside the node agent that manages the node whose servers you want to restart, then click **Restart all Servers on Node**.

   Note that the node agent for the node must be processing in order to restart application servers on the node.

3. Stop the processing of a node agent. On the Node Agents page, place a check mark in the check box beside the node agent you want to stop processing, then click **Stop**.

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** .

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.

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 a node is added to a cell, the format in which the name is specified is based on the version of IP the node will be using.

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.

Status
Indicates whether the node agent server is started or stopped.

Note that if the status of servers such application servers is *Unavailable*, the node agent is not running in the servers' node and you must restart the node agent before you can start the servers.

Node agent server settings
Use this page to view information about and 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 Network Deployment environment.

To view this administrative console page, click **System Administraton > Node Agents >***node_agent_name*.

A node agent must be started on each node in order for the deployment manager node to be able to collect and control servers configured on that node. If you use configuration synchronization support, a node agent coordinates with the deployment manager server to synchronize the node's configuration data with the master copy managed by the deployment manager.

You must initially start a node agent outside the administrative console. For information on how to initially start a node agent, see the WebSphere Application Server Information Center.

The Runtime tab displays only when a node agent runs.

Name
Specifies a logical name for the node agent server.

Data type String
Default NodeAgent Server

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, SAF (for example, RACF), started task control, and others.

The name is 1-8 characters, alpha-numeric or national language. It 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.

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.

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 a node is added to a cell, the format in which the name is specified is based on the version of IP the node will be using.

Data type String

State
Indicates whether the node agent server is started or stopped.

Data type String
Default Started

Remote file services
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 file services do the following:

File transfer service

The file transfer service enables the moving of files between the network manager and the nodes. 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 defined in the Network Deployment server configuration under its WebContainer HTTP transports.

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.

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.

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.

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.

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.

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.

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.

For the z/OS platform, disablement of automatic synchronization is recommended when in a production environment or if you use the application rollout update capability.

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

z/OS location service daemons

Location service daemons provide the CORBA location service in support of Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP).

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).

Steps for stopping or canceling the z/OS location service daemon from the MVS console

If you cancel or stop the location service daemon, it cancels or stops all WebSphere Application Servers on the system. If you installed Network Deployment, the location service daemon also cancels or stops the deployment manager and the node agents.

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.

Determining if the z/OS location service daemon is running

This article describes what steps you must follow to determine if the location service daemon is running on a z/OS system.

The location service daemon starts automatically if it is not already running when you start an Application Server or a deployment manager server.

Perform the following step any time you want to know whether the location service daemon is running.

To determine if the location service daemon is running, issue one of the display commands, such as `d a,l`, from the MVS console.

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 the administrative console.

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.

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.

You can now modify the location service daemon settings in the administrative console.

1. Go to the settings page for the location service daemon to view the help page.
2. Click **System Administration > Node groups > *sysplex node group* > z/OS Location Service Daemon** 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

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 by the customization dialog 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 can be any of the following choices:

- IP name such as www.foobar.com
- IP address of the form n.n.n.n, where n is an integer
- Dynamic virtual IP address (DVIPA)

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 agents: Resources for learning

Use the following links to find relevant supplemental information about WebSphere Application Server administrative agents 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 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 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 developerWorks WebSphere

  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-3.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.

Configuring the environment

To assist in handling requests among Web applications, Web containers, and application servers, you can configure cell-widesettings for virtual hosts, variables and shared libraries.

1. Configure virtual hosts.
2. Configure variables.
3. If your deployed applications use shared library files, define the shared library files needed.

   See “Managing shared libraries” on page 182.

Virtual hosts

A virtual host is a configuration that enables a single host machine to resemble multiple host machines. 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.

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 and serve the servlet. If no match is found, an error is returned to the browser.

An application server provides a default virtual host with some common aliases, such as the machine's IP address, short host name, and fully qualified host name. The alias comprises the first part of the path for accessing a resource such as a servlet. For example, it is localhost:80 in the request http://localhost:80/myServlet.

A virtual host is not associated with a particular node (machine). It is a configuration, rather than a "live object," explaining why you can create it, but cannot start or stop it. For many users, creating virtual hosts is unnecessary because the default_host is provided.

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.

Why you would use virtual hosting

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. 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.

Virtual hosts allow the administrator to isolate and independently manage multiple sets of resources on the same physical machine.

Suppose 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.

Further suppose that 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.

The servlets on one virtual host do not share their context with the servlets on the other virtual host. Requests for the servlet on VirtualHost1 can continue as usual. This is true even though VirtualHost2 is refusing to fill requests for the servlet with the same name.

You associate a servlet or other application with a virtual host instead of the actual DNS address.

The default virtual host (default_host)

The product provides a default virtual host (named default_host).

The virtual host configuration uses wildcard entries with the ports for its virtual host entries.
- The default alias is *:80, using an internal port that is not secure.
- Aliases of the form *:9080 use the secure internal port.
- Aliases of the form *:9443 use the external port that is not secure.
- Aliases of the form *:443 use the secure external port.

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.

How requests map to virtual host aliases

Virtual hosts let you manage a single application server on a single machine as if the application server were multiple application servers that are each on their own host machine. Resources associated with one virtual host cannot share data with resources associated with another virtual host, even though the virtual hosts share the same application server on the same physical machine.

When you request a resource, WebSphere Application Server tries to map the request to an alias of a defined virtual host.

Mappings are both case sensitive and insensitive. For example, the portion "http://host:port/" is case insensitive, but the URL that follows is case sensitive. The match must be alphanumerically exact. Also, 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 due to 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 was 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 WebSphere Application Server, *server1*.
3. The server1 Application Server 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.

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 when WebSphere Application Server 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.

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.

Configuring virtual hosts

Virtual hosts enable you to isolate and independently manage multiple sets of resources on the same physical machine.

1. Create a virtual host using the "Virtual host collection" on page 161 of the administrative console. Click **Environment > Virtual Hosts** from the navigation tree of the console. Click **New**. On the "Virtual host settings" on page 162 page that displays, specify an administrative name for the virtual host. When you create a virtual host, a default set of 90 MIME entries is created for the virtual host.
2. There must be a virtual host alias corresponding to each port used by an HTTP transport. There is one HTTP transport in each Web container, usually assigned to the virtual host named default_host. You can change the default assignment to any valid virtual host.

   You must create a virtual host for each HTTP port in the following cases:
   - You are using the internal HTTP transport with a port other than the default of 9080. You must define the port that you are using.
   - You are using the default port 9080, but the port is no longer defined. You must define port 9080.
   - You have created multiple Application Servers (either stand-alone servers or cluster members) that use the same virtual host. Because each server must be listening on a different HTTP transport port, you must define a virtual host alias for the transport port of each server.

   If you define new virtual host aliases, identify the port values that the aliases use on the "HTTP transport collection" on page 262 page.
3. If necessary, create a virtual host alias for each HTTP transport port.

   From the virtual host collection "Virtual host collection" on page 161 page, click your virtual host. On the "Virtual host settings" on page 162 page for the virtual host, click **Host aliases**. To define a virtual host alias on the "Host alias collection" on page 162 page, click **New**. On the "Host alias settings" on page 162 page for the virtual host alias, specify a host name and a port. Configure the virtual host to contain an alias for the port number. For example, specify an alias of `*:9082` if 9082 is the port number in use by the transport.
4. When you enter the URL for the application into a Web browser, include the port number in the URL.

   For example, if 9082 is the port number, specify a URL such as `http://localhost:9082/wlm/SimpleServlet`
5. If MIME entries are not specified at the Web module level, define MIME object types and their file name extensions. For each needed MIME entry on the "MIME type collection" on page 163 page, click **New**. On the "MIME type settings" on page 164 page, specify a MIME type and extension.

6. After you configure a virtual host alias or change a configuration, you must regenerate the Web server plug-in configuration and restart WebSphere Application Server.

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 (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.

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 default reflects the value specified at product setup. The default might be `80`, `81`, `9060` or a similar value.

Data type	Integer
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

Variables

A variable is a configuration property that can be used to provide a parameter for some values in the system. A variable has a name and a value.

Not all WebSphere components support the use of a variable that you can define using this function. Test your application to verify that variables that you define are being used correctly.

WebSphere variables are used for:

- Configuring WebSphere Application Server path names, such as JAVA_HOME, and APP_INSTALL_ROOT.
- Configuring certain cell-wide or cluster-widecustomization values.
- Configuring the WebSphere Application Server for z/OS location service.

Each variable has a scope. A scope is the range of locations in the WebSphere Application Server 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.

You can use variables in configuration values such as file system path settings. Use the following syntax to refer to a variable:

`${variable_name}`

The value of a variable can contain a reference to another variable. The value of the variable is computed by substituting the value of the referenced variable recursively.

Variables are useful when concatenating two path variables when the specification does not accept the *AND* operator. For example, suppose that the following variables exist:

Variable name	Variable value
ROOT_DIR	/
HOME_DIR	${ROOT_DIR}home
USER_DIR	${HOME_DIR}/myuserdir

The variable reference ${USER_DIR} resolves to the value /home/myuserdir.

Configuring WebSphere variables

This topic describes how to create a WebSphere Application Server variable.

You can define a WebSphere Application Server variable to provide a parameter for some values in the system. After you define the name and value for a variable, the value is used in place of the variable name. Variables most often specify file paths. However, some system components also support the use of variables.

WebSphere variables are used for:

- Configuring WebSphere Application Server path names, such as JAVA_HOME, and APP_INSTALL_ROOT.
- Configuring certain cell-wide or cluster-wide customization values.
- Configuring the WebSphere Application Server for z/OS location service.

The scope of a variable can be cell-wide, cluster-wide, node-wide, or applicable to only one server process.

Define variables on the **Environment > WebSphere Variables** console page.

Define the scope to apply a variable cell-wide, cluster-wide, node-wide or to only one server process. A variable resolves to its new value when used in a component that supports the use of variables.

1. Click **Environment > WebSphere Variables** in the administrative console to define a new variable.
2. Specify the scope of the variable. Declare the new variable for the **Cell**, **Cluster**,**Node** or **Server** and click **Apply**.

   The variable exists at the level you specify. Define a variable at multiple levels to use multiple values. The more granular definition overrides the higher level setting.

For instance, if you specify the same variable on a node and a server, the server setting overrides the node setting. Similarly, a node level setting overrides a cluster or cell setting.

Scoping variables is particularly important when 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.

See Test connection service for more information.

3. Click **New** on the WebSphere Variables page.
4. Specify a name, a value, and a description on the Variable page. Click **OK**. You can use the following options for setting WebSphere variables:
   - Use "WebSphere variables predefined for the z/OS platform" on page 169.
   - 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.
   - WebSphere variables support substitution. The name of a variable can be formed by substituting the value of another variable. If you enter `${<variable name>}` in the **Name** field, the value of `<variable name>` will be the name of your new WebSphere variable. For example `${JAVA_HOME}` will create a WebSphere variable with a name that is equal to the Java home directory.
   - The application server uses WebSphere 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.
5. Verify that the variable is displayed in the list of variables. The administrative console does not pick up typing errors. The variable is ignored if it is referred to incorrectly.
6. Save your configuration.
7. Stop the server and start the server again to put the variable configuration into effect.

WebSphere variables collection

Use this page to view and change a list of substitution variables with their values and scope.

To view this administrative console page, click **Environment > WebSphere Variables**.

For information on a variable, click the variable and read the value in the **Description** field.

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 Application Server variable is visible on the administrative console panel.

A resource can be visible in the administrative console collection table at the cell, cluster, node, or server scope.

Variable settings

Use this page to define the name and value of a WebSphere Application Server substitution variable.

To view this administrative console page, click **Environment > WebSphere Variables >***WebSphere_variable_name*.

Name:

Specifies the symbolic name for a WebSphere Application Server variable. For example, a variable name might represent a physical path or URL root that is used by WebSphere Application Server.

WebSphere variables are used for:

- Configuring WebSphere Application Server path names, such as JAVA_HOME, and APP_INSTALL_ROOT.
- Configuring certain cell-widecustomization values.
- Configuring the WebSphere Application Server for z/OS location service.

WebSphere Application Server substitutes the symbolic name wherever its value displays in the system.

For example, ″JAVA_HOME″ is the symbolic name representing the file system path to the installation directory for the Java Virtual Machine (JVM). For example, the value is `/opt/IBM/WebSphere/AppServer/java` for the WebSphere Application Server product on a Linux machine.

You can create new variables for use in WebSphere Application Server components that support the use of variables.

Data type String

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.

For example, `/opt/IBM/WebSphere/AppServer/java` is the value on a Linux machine for a variable named JAVA_HOME.

Data type String

Description:

Documents the purpose of a variable.

Data type String

IBM Toolbox for Java JDBC driver

WebSphere Application Server supports the **IBM Toolbox for Java** JDBC driver. The IBM Toolbox for Java JDBC driver is included with the IBM Toolbox for Java product.

IBM Toolbox for Java is a library of Java classes that are optimized for accessing iSeries and AS/400 data and resources. You can use the IBM Toolbox for Java JDBC driver to access local or remote **DB2 UDB for iSeries 400** databases from server-side and client Java applications that run on any platform that supports Java.

IBM Toolbox for Java is available in these versions:

IBM Toolbox for Java licensed program
The licensed program is available with every OS/400 release, starting with Version 4 Release 2 (V4R2). You can install the licensed program on your iSeries 400 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 iSeries information center: http://publib.boulder.ibm.com/iseries/v5r1/ic2924/index.htm 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-1.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 JDBC driver for both versions supports JDBC 2.0. For more information about IBM Toolbox for Java and JTOpen, see the product Web site at http://www-1.ibm.com/servers/eserver/iseries/toolbox/index.html.

Note: If you are using WebSphere Application Server on platforms other than iSeries, use the **JTOpen** version of the Toolbox JDBC driver.

Configure and use the jt400.jar file

1. Download the *jt400.jar* file from the **JTOpen** URL at http://www-1.ibm.com/servers/eserver/iseries/toolbox/downloads.htm. Place it in a directory on your workstation such as *C:\JDBC_Drivers\Toolbox*.
2. Open the administrative console.
3. Select **Environment**.
4. Select **Managed WebSphere Variables**.
5. Set the managed variable *OS400_TOOLBOX_JDBC_DRIVER_PATH* at the **Node** level.

6. Double click **OS400_TOOLBOX_JDBC_DRIVER_PATH**.
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,

   ```
   OS400_TOOLBOX_JDBC_DRIVER_PATH == "C:\JDBC_Drivers\Toolbox"
   ```

   When you choose a Toolbox driver from the list of possible resource providers the **Classpath** field looks like:

   ```
   Classpath == ${OS400_TOOLBOX_JDBC_DRIVER_PATH}/jt400.jar
   ```

WebSphere variables predefined for the z/OS platform

The following WebSphere variables are predefined for the z/OS platform.

protocol_http_defaultIdentity:

This variable is pre-configured at the cell level. It specifies the user identity associated with unauthenticated HTTP requests. Use `protocol_http_defaultIdentity` only if unauthenticated users have been permitted through the Global Security settings. This value is initially configured through the WebSphere Application Server for z/OS ISPF Customization Dialog.

To change the value specified for this variable, in the administrative console click **Environment > WebSphere Variables > New** and enter `protocol_http_defaultIdentity` in the Name field and the appropriate user identity in the Value field.

Data type	String
Default	WSADMIN

protocol_https_defaultIdentity:

This variable is pre-configured at the cell level. It specifies the user identity associated with unauthenticated HTTPS requests. Use `protocol_https_defaultIdentity` only if unauthenticated users have been permitted through the Global Security settings. This value is initially configured through the WebSphere Application Server for z/OS ISPF Customization Dialog.

To change the value specified for this variable, in the administrative console click **Environment > WebSphere Variables > New** and enter `protocol_https_defaultIdentity` in the Name field and the appropriate user identity in the Value field.

Data type	String
Default	WSADMIN

protocol_https_cert_mapping_file:

This variable can be set at the cell, node, or server level. It specifies the name of a file containing entries that map IP addresses to server certificate labels. When an HTTP SSL connection request is received, the application server checks the IP address against entries in the file specified on this variable. If a match is made, the certificate mapped to the IP address is used for the connection. If no match is found, the application server checks for the existence of the `protocol_https_default_cert_label` variable. If a value has been specified on the `protocol_https_default_cert_label` variable, the certificate specified will be used to establish the connection. If no value has been specified on the

protocol_https_default_cert_label variable, the default server certificate specified in the RACF SSL keyring owned by the application server will be used to establish the HTTP SSL connection.

To set this variable at the cell level, in the administrative console click **Environment > WebSphere Variables > New** and enter protocol_https_cert_mapping_file in the Name field and the appropriate file name in the Value field.

To set this variable at the node level, in the administrative console click **System administration > Nodes >** *node_name* **> Custom properties > New** and enter protocol_https_cert_mapping_file in the Name field and the appropriate file name in the Value field.

To set this variable at the server level, in the administrative console click **Servers > Application servers >** *server_name* **> Web container settings > Web container > Custom properties > New** and enter protocol_https_cert_mapping_file in the Name field and the appropriate file name in the Value field.

Data type String

protocol_https_default_cert_label:

This variable can be set at the cell, node, or server level. It specifies the label of the server certificate that should be used when establishing HTTP SSL connections with the application server. If no value is specified and the IP address of the server does not match an IP address contained in the certificate mapping file specified on the protocol_https_cert_mapping_file variable, the default server certificate specified in the RACF SSL Keyring owned by the application server will be used to establish the HTTP SSL connection.

To set this variable at the cell level, in the administrative console click **Environment > WebSphere Variables > New** and enter protocol_https_default_cert_label in the Name field and the appropriate certificate label in the Value field.

To set this variable at the node level, in the administrative console click **System administration > Nodes >** *node_name* **> Custom properties > New** and enter protocol_https_default_cert_label in the Name field and the appropriate certificate label in the Value field.

To set this variable at the server level, in the administrative console click **Servers > Application servers >** *server_name* **> Web container settings > Web container > Custom properties > New** and enter protocol_https_default_cert_label in the Name field and the appropriate certificate label in the Value field.

Data type String

wlm_classification_file:

This variable can be set at the cell, node, or server level. It specifies the location of your workload classification document.

Use this variable when classifying z/OS workload using the common XML file for classifying inbound HTTP, IIOP, and message-driven bean (MDB) work as

described in "Classifying z/OS workload" on page 317. Any rules that are defined in the common XML file override the old format HTTP classification. See the section "Transaction class mapping file entries" in the *Administering applications and their environment* PDF for more information. The common XML file also overrides any rules in the MDB classification file that is defined by the endpoint_config_file WebSphere variable.

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.

To set this variable at the cell level, in the administrative console click **Environment > WebSphere Variables > New** and enter `wlm_classification_file` in the Name field and the location of your workload classification document in the Value field.

To set this variable at the node level, in the administrative console click **System administration > Nodes >** *node_name* **> Custom properties > New** and enter `wlm_classification_file` in the Name field and the location of your workload classification document in the Value field.

To set this variable at the server level, in the administrative console click **Servers > Application servers >** *server_name* **> Web container settings > Web container > Custom properties > New** and enter `wlm_classification_file` in the Name field and the location of your workload classification document in the Value field.

Data type String

Certificate mapping file entries

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
```

Repository service custom properties on z/OS

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.

Support for the following custom property is provided with the WebSphere Application Server for z/OS product.

recoveryNode:

Specifies that a node is a recovery node. 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 z/OS custom properties

Use this page to configure WebSphere Application Server custom properties on a z/OS platform.

Application server custom properties can be specified in the administrative console by clicking **Application Servers** > *server1* > **Server infrastructure** > **Administration** > **Custom Properties**. The support for the following server custom properties is provided with the WebSphere Application Server for z/OS product.

Note: Setting server properties as server-level WebSphere Application Server for z/OS variables is deprecated. However, you can set a WebSphere Application Server for z/OS variable with a scope of `cell` or `node` to establish custom defaults for server properties.

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.

Set this value to `0` (zero) to disable the function.

Data Type	Integer
Units	Seconds
Default	120

protocol_accept_http_work_after_min_srs:

Specifies whether or not the application server waits for a minimum number of servants (specified on the wlm_minimumSRCount variable) to be up before starting the HTTP transports. If this property is set to `true`, when the minimum number of servants is ready for work, the HTTP transport starts accepting work. If this property is set to `false`, the HTTP transports are started when the controller starts.

Data Type	Boolean
Default	false

Used by Daemon	no

protocol_bboc_log_response_failure:

Specifies that if the BBOO0168W message is issued, the failure 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 routing information identifying the client.

Data Type	Boolean
Default	false
Used by Daemon	yes

protocol_bboc_log_return_exception:

Used to indicate 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 routing information identifying the client.

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) exported from this server use the GIOP level indicated.

You might need to change this property from the default if you use a non-WebSphere Application Server client ORB that supports a lower 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 an older, non-WebSphere Application Server client ORB.

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 used 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_response_buffer:

Specifies, in bytes, the maximum length of the response buffer used for HTTP requests. Responses larger than this value are rejected. A value of 0 indicates not to allocate a large response buffer. An HTTP large response buffer is not required if all the HTTP responses are less than 10 MB.

Data Type	Integer
Default	104857600
Used by Daemon	No

protocol_http_large_data_inbound_buffer:

Specifies, in bytes, the length of a serially reusable inbound buffer used for HTTP requests larger than 10 megabytes. This value limits the size of incoming requests. For example, if you set the property to 15 megabytes, any requests over 15 megabytes are rejected. Specify `0` (zero) to indicate that no buffer is needed. Requests that are larger than 10 megabytes are rejected.

Data Type	Integer
Default	0
Used by Daemon	No

protocol_http_timeout_output_recovery:

Specifies the action for the timer expiration. Set the value to SESSION to send the client a message stating that the server timed out and let the server continue running.

Data Type	String
Default	SERVER
Used by Daemon	No

protocol_https_backlog:

Specifies the maximum length for the queue of pending connections using HTTPS. The value used 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_iiop_backlog:

Used to specify the maximum length for the queue of pending connections using the CORBA Internet Inter-ORB protocol (IIOP). The value used may 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:

Used to specify the maximum length for the queue of pending connections using IIOP Secure Sockets Layer (SSL). The value used can be limited by the specification of the SOMAXCONN statement in the TCP/IP Profile.

Data Type	Integer
Default	10
Used by Daemon	Yes

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 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 hardcopy. These messages are redirected to the location identified through the DD card on the server's job control language (JCL) start procedure. These write to operator (WTO) messages are primarily messages that WebSphere Application Server for z/OS issues 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 by IBM service 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 dump options use with a CEE3DMP dump. If you want more than one option, separate each option with a blank. Do not change this property unless directed by IBM service personnel.

Data Type	String
Default	THREAD(ALL) BLOCKS

Used by Daemon	Yes

ras_hardcopy_msg_dd:

Specifies redirect write to operator (WTO) messages that WebSphere Application Server for z/OS routes 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 issued from Java code during initialization.

Data Type	String
Default	empty string
Used by Daemon	Yes

ras_log_logstreamName:

Specifies the log stream for WebSphere Application Server for z/OS to use 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, WebSphere Application Server for z/OS uses the STDERR stream.

Data Type	String
Default	empty string
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:

- `CEEDUMP` - Captures callback and offsets. It takes time for the system to take CEEDUMPs. Transaction time-outs can occur.
- `TRACEBACK` - Captures Language Environment and z/OS UNIX traceback data.
- `SVCDUMP` - Captures an MVS dump (but does not produce a dump in the client).

ras_time_local:

Specifies whether timestamps in the error log display is in local time or Greenwich Mean Time (GMT). The timestamp 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 WebSphere Application Server for z/OS 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. Contact IBM service for the subcomponent numbers and their meanings. Do not change this property unless directed by IBM service personnel.

Data Type	String
Default	null (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 the size of a single trace buffer in bytes. 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_ctraceParms:

Specifies the identify 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 this property is specified and the PARMLIB member is not found, the default PARMLIB member, `CTIBBO00`, is used. If neither the specified nor the default PARMLIB member is found, tracing is defined to CTRACE, but no connection is available to a CTRACE external writer.

Data Type	String
Default	null (empty string)
Used by Daemon	Yes

ras_trace_defaultTracingLevel:

Specifies the default tracing level for WebSphere Application Server for z/OS. Use this variable with the `ras_trace_basic` and `ras_trace_detail` variables to set

tracing levels for Application Server for z/OS subcomponents. Do not change this property unless directed by IBM service 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 WebSphere Application Server for z/OS subcomponents. Subcomponents, specified by numbers, receive detailed traces. If you specify more than one subcomponent, use parentheses and separate the numbers with commas. Contact IBM service for the subcomponent numbers and their meanings. Do not change this property unless directed by IBM service personnel.

Data Type	String
Default	null (empty string)
Used by Daemon	Yes

ras_trace_exclude_specific:

Specifies WebSphere Application Server for z/OS trace points to exclude from tracing activity.

Trace points are specified by 8-digit, hexadecimal numbers. Do not use this property unless directed 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 WebSphere Application Server for z/OS variable name by enclosing the name in single quotes.

Data Type	String
Default	empty string
Used by Daemon	Yes

Note: Sometimes results depend on the ras_trace_minorCodeDefault environment variable. If you code `ras_trace_minorCodeTraceBacks=ALL` and `ras_minorcode_action=NODIAGNOSTICDATA`, you get a traceback. But, if you code `ras_trace_minorCodeTraceBacks=(null value)` and `ras_minorcode_action=TRACEBACK`, you also get a traceback. So, specifying `ras_trace_minorCodeTraceBacks=(null value)` does not cancel the traceback; it does not cause TRACEBACK data to be collected.

ras_trace_outputLocation:

Specifies where to send trace records.

- To SYSPRINT

- To a memory buffer (BUFFER), the contents of which are later written to a CTRACE data set
- To a trace data set (TRCFILE) that is specified on the TRCFILE DD statement in the server start procedure.

For servers, you might 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 WebSphere Application Server for z/OS 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 quotes. Do not use this property unless directed by IBM service personnel.

Data Type	String
Default	null (empty string)
Used by Daemon	Yes

server_region_jvm_localrefs:

Do not use this property unless directed 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, all of the servers write to the same file, so you might have difficulty using the file for diagnostic purposes.

Data Type	String (file name)
Default	empty string (no file name)
Used by Daemon	No

server_region_recycle_count:

Specifies the number of transactions processed by a servant process after which the servant process is recycled. z/OS Workload management (WLM) ends the servant after all affinity requirements have been met. Specify a nonzero value to enable recycling.

You might want to enable recycling if, after running for an extended period of time, 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_start_wait_for_initialization_Timeout:

Specifies how long the startServer.sh command processing waits for WebSphere Application Server initialization to complete. By default, it waits indefinitely until initialization is complete.

You might want to use this property if you want to:

- 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 continue waiting if auto-started Web applications unexpectedly enter a long wait.)

Data Type	Integer
Default	0
Used by Daemon	No

server_use_wlm_to_queue_work:

Specifies whether or not WLM for z/OS is used for workload queuing.

This property should be set to 1 if you are using a stateless application models. With these models, application objects, such as EJBs and HTTPSessions, are only resident in memory for the life of an individual request. Therefore, the WebSphere Application Server should be configured to enable WLM for z/OS to dynamically balance individual requests. This configuration allows the WebSphere Application Server to provide linear scalability and consistent, repeatable response times.

This property should be set to 0 (zero) 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 Application Server JVMs for a period of 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. These application models prevent the Application Server from enabling the use of WLM for z/OS for dynamic workload routing of individual requests in a clustered environment because of the client's affinity to a specific Java Virtual Machine (JVM). In this situation, a round robin algorithm should be used to handle the client's initial request. This algorithm evenly distributes creation of long term affinities and is the best technique for achieving a balanced utilization of system resources under these conditions.

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 want to exploit WLM for z/OS's round robin capability instead of the previously described WebSphere Application Server capability, seeWorkload

management (WLM) for z/OS . The differences between WLM for z/OS's round robin capability and the WebSphere Application Server 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 2nd client to go to the other servant. The WebSphere Application Server provided round robin, initiated by setting server_use_wlm_to_queue_work=0 and server_work_distribution_algorithm=1, acts this way. However, the round robin WLM for z/OS provides goes through all of the threads in the first servant before moving to the threads in the next servant.

When the server_use_wlm_to_queue_work property is set to 0 (zero), the wlm_minimumSRCount and wlm_maximumSRCount properties should be set to the same value. Because the work is not going thru WLM for z/OS, WLM or z/OS only starts the number of servants specified for the wlm_minimumSRCount property.

0 (zero)	WLM for z/OS is not used.
1	WLM for z/OS is used.
Default	1
Used by Daemon	No

server_work_distribution_algorithm:

Specifies the type of work distribution algorithm the Application Server will use 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, the value specified for this property is ignored.

0 (zero)	The hot thread algorithm is used. This option is not recommended.
1	The round robin algorithm is used. This value must be specified if the server_use_wlm_to_queue_work custom property is set to 0 for conversational application models.
Default	0
Used by Daemon	No

transaction_recoveryTimeout:

Specifies the time, in minutes, that this controller (region) uses to attempt to complete all restarted transactions before issuing a write-to-operator-with-reply (WTOR) message to the console, requesting whether to:

- 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, the controller (region) 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 in peer restart and recovery mode.

Data Type	Integer
Default	15
Used by Daemon	No

Shared library files

Shared library files in WebSphere Application Server consist of a symbolic name, a Java class path, and a native path for loading Java Native Interface (JNI) libraries.

You can define a shared library at the cell, node, or server level. Defining a library at one of the three levels does not cause the library to be placed into the application server's class loader. You must associate the library to an application or server in order for the classes represented by the shared library to be loaded in either a server-wide or application-specific class loader.

A separate class loader is used for shared libraries that are associated with an application server. This class loader is the parent of the application class loader, and the WebSphere Application Server extensions class loader is its parent. Shared libraries that are associated with an application are loaded by the application class loader.

Managing shared libraries

Shared libraries are files used by multiple applications. Using the administrative console, you can define a shared library at the cell, node, or server level. You can then associate the library to an application or server to load the classes represented by the shared library in either a server-wide or application-specific class loader. Using an installed optional package, you can 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.

If your deployed applications use shared library files, define shared libraries for the library files and associate the libraries with specific applications or with an application server. Associating a shared library file with a server associates the file with all applications on the server. Use the Shared Libraries page to define new shared library files to the system and remove them.

- Use the administrative console to define a shared library.
  1. Create a shared library for each library file that your applications need.
  2. Associate each shared library with an application or a server.
     - Associate a shared library with an application that uses the shared library file.
     - 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.

The first step for making a library file available to multiple applications deployed on a server is to create a shared library for each library file that your applications need. When you create the shared libraries, set variables for the library files.

Use the Shared Libraries page to create and configure shared libraries.

1. Go to the Shared Libraries page. Click **Environment > Shared Libraries** in the console navigation tree.
2. Change the scope of the collection table to see what shared libraries are in a cell, node, or server.
   a. Select the cell, a node, or a server.
   b. Click **Apply**.
3. Click **New**.
4. Configure the shared library.
   a. On the settings page for a shared library, specify the name, class path, and any other variables for the library file that are needed.
   b. Click **Apply**.
5. Repeat steps 1 through 4 until you define a shared library instance for each library file that your applications need.

Using the administrative console, associate your shared libraries with specific applications 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.

Alternatively, you can use an installed optional package to associate your shared libraries with an application.

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**.

By default, a shared library is accessible to applications deployed (or installed) on the same node as the shared library file. Use the **Scope** field to change the scope to a different node or to a specific server.

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*.

Name:

Specifies a name for the shared library.

Data type	String

Description:

Describes the shared library file.

Data type	String

Classpath:

Specifies the class path used to locate the JAR files for the shared library support.

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.

Data type	String
Units	Class path

Associating shared libraries with applications

You can associate a shared library with an application. Classes represented by the shared library are then loaded in the application's class loader, making the classes available to the application.

This article assumes that you have defined a shared library at the cell, node, or server level. The shared library represents a library file used by multiple deployed applications.

This article also assumes that you want to use the administrative console, and not an installed optional package, to associate a shared library with an application.

To associate a shared library with an application, 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.

1. Click **Applications > Enterprise Applications >***application_name* **> Libraries** in the console navigation tree to access the Library Ref page.
2. Click **Add**.
3. On the settings page for a library reference, specify variables for the library reference as needed. The variables identify the shared library file that your application uses.
4. Click **Apply**.

The name of the library reference is shown in the list on the Library Ref page.

Repeat steps 2 through 4 until you define a library reference instance for each shared library that your application requires.

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.

This article assumes that you have defined a shared library at the cell, node, or server level. The shared library represents a library file used by multiple deployed applications.

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.

1. Configure class loaders for applications deployed on the server.
   a. Click **Servers > Application Servers >** *server_name* to access the settings page for the application server.
   b. Set values for the application **Class loader policy** and **Class loading mode** of the server. For information on these settings, see "Application server settings" on page 205 and Class loaders.
2. Create a library reference for each shared library file that your application needs.
   a. Go to the settings page for a class loader.
   b. Click **Libraries** to access the Library Ref page.
   c. Click **Add**.
   d. On the settings page for a library reference, specify variables for the library reference as needed. The variables identify 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 Ref page.

   Repeat the previous steps until you define a library reference for each shared library that your application needs.

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 2 Platform, Enterprise Edition (J2EE) application is installed on a server or cluster, dependency information is specified in its manifest file. WebSphere Application Server reads the dependency information of the application (`.ear` file) to automatically associate the application with an installed optional

package .jar file. WebSphere Application Server 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 WebSphere Application Server are described in section 8.2 of the J2EE specification, Version 1.4 at http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf.

WebSphere Application Server supports using the manifest file (manifest.mf) in shared library .jar files and application .ear files. 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.

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 J2EE application or a module within a J2EE 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/util1) 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.

Read about installed optional packages in "Installed optional packages" on page 185 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.

Installed optional packages expand the existing shared library capabilities of an application server. Prior to Version 6, 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.

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 WebSphere Application Server 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 serveror cluster.

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 **Applications > Enterprise Applications >***application_name* **> Libraries**.

If no shared libraries are defined, 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 can be used by deployed applications. 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 **Applications > Enterprise Applications >***application_name* **> Libraries >***library_reference_name*. A shared library must be defined to view this page.

A shared library is a container-wide library file that can be used by deployed applications. 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

Environment: Resources for learning

Use the following links to find relevant supplemental information about configuring the WebSphere Application Server cell-wide environment. 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 Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

Programming instructions and examples

- WebSphere Application Server education
-
- Best Practice: Configuring Web Applications on WebSphere Application Server for z/OS and OS/390

Administration

- Listing of all IBM WebSphere Application Server Redbooks

Working with server configuration files

Application server configuration files define the available application servers, their configurations, and their contents.

You should periodically save changes to your administrative configuration. You can change the default locations of configuration files, as needed.

1. Edit configuration files. The master repository is comprised of .xml configuration files. You can edit configuration files using the administrative console, scripting, wsadmin commands, programming, or 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.
2. Save changes made to configuration files. Using the console, you can save changes as follows:
   a. Click **Save** on the taskbar of the administrative console.
   b. Put a check mark in the **Synchronize changes with Nodes** check box.
   c. On the Save page, click **Save**.
3. Handle temporary configuration files resulting from a session timing out.
4. Change the location of temporary configuration files.
5. Change the location of backed-up configuration files.
6. Change the location of temporary workspace files.
7. Back up and restore configurations.

Configuration documents

WebSphere Application Server stores configuration data for servers in several documents in a cascading hierarchy of directories. The configuration documents describe the available application servers, their configurations, and their contents. Most configuration documents have XML content.

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 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*.

On the 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 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 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”″ states whether you can edit a document using the administrative tools or must edit it directly.

Configuration document descriptions

Most configuration documents have XML content. The table below 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

(Locations split for publishing)

Configuration file	Locations	Purpose	Manual editing required
admin-authz.xml	config/cells/*cell_name*/	Define a role for administrative operation authorization.	X
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

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
|	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

Configuration repositories

A configuration repository stores configuration data. By default, configuration repositories reside in the *config* subdirectory of the product installation root 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 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.

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.

When a session times out, the configuration file in use is saved under the userid/timeout directory under the ServletContext's temp area. This is the value of the javax.servlet.context.tempdir attribute of the ServletContext. By default, it is:
install_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 next time you log on to the console, you are prompted to load the saved configuration file. If you decide to load the saved file:
1. If a file with the same name exists in the *install_root*/config directory, that file is moved to the userid/backup directory in the temp area.
2. The saved file is moved to the *install_root*/config directory.
3. The file is then loaded.

If you decide not to load the saved file, it is deleted from the userid/timeout directory in the temp area.

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.

Changing the location of temporary configuration files

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 using system variables or the administrative console.

The default location for the configuration temporary directory is CONFIG_ROOT/temp. Change the location by doing either of the following:
- Set the system variable *was.repository.temp* to the location you want for the repository temporary directory. Set the system variable when launching a Java process using the -D option. For example, to set the default location of the repository temporary directory, use the following option:

  ```
  -Dwas.repository.temp=%CONFIG_ROOT%/temp
  ```

- Use the administrative console to change the location of the temporary repository file location for each server configuration. For example, on the Network Deployment product, to change the setting for a deployment manager, do the following:
  1. Click **System Administration > Deployment Manager** in the navigation tree of the administrative console. Then, click **Administration Services**, **Repository Service**, and **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 can include WebSphere Application Server variables such as `${WAS_TEMP_DIR}/config`. Then, click **OK**.

The system property set using the first option takes precedence over the configuration property set using the second option.

Changing the location of backed-up configuration files

During administrative processes like adding a node to a cell or updating a file, configuration files are backed up to a backup location. The default location for the backup configuration directory is CONFIG_ROOT/backup. Change the location by doing either of the following:

- Set the system variable *was.repository.backup* to the location you want as the repository backup directory. Set the system variable when launching a Java process using the -D option. For example, to set the default location of the repository backup directory, use the following option:

  ```
  -Dwas.repository.backup=%CONFIG_ROOT%/backup
  ```
- Use the administrative console to change the location of the repository backup directory for each server configuration. For example, on the Network Deployment product, do the following to change the setting for a deployment manager:
  1. Click **System Administration > Deployment Manager** in the navigation tree of the administrative console. Then, click **Administration Services**, **Repository Service**, and **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 can include WebSphere Application Server variables such as `${WAS_TEMP_DIR}/backup`. Then, click **OK**.

The system property set using the first option takes precedence over the configuration property set using the second option.

Changing the location of temporary workspace files

The administrative console workspace allows client applications to navigate the configuration. Each workspace has its own repository location defined either in the system property or the property passed to a workspace manager when creating the workspace: workspace.user.root or workspace.root, which is calculated as %workspace.root%/*user_ID*/workspace/wstemp.

The default workspace root is calculated based on the user installation root: %user.install.root%/wstemp. You can change the default location of temporary workspace files by doing the following:

- z/OS platform: Change the setting for the system variable *workspace.user.root* or *workspace.root* so its value is no longer set to the default location. Set the system variable in the servant process when launching a Java process using the -D option. For example, to set the default location the full path of the root of all users' directories, use the following option:

  `-Dworkspace.user.root=`*`full_path_for_root_of_all_user_directories`*

Backing up and restoring administrative configurations

WebSphere Application Server represents its administrative configurations as XML files. You should back up configuration files on a regular basis.

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.
3. Run the restoreConfig command to restore configuration files. Specify backup files that do not contain invalid or inconsistent configurations.

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

Backing up the WebSphere Application Server for z/OS system

Use the following guidelines to back up parts of your WebSphere Application Server for z/OS system.

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 executables, 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 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 developerWorks WebSphere

  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-3.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.

Administering application servers

An application server configuration provides settings that control how an application server provides services for running enterprise applications and their components.

This section describes how to create and configure application servers in an existing application server environment.

A WebSphere Application Server administrator can configure one or more application servers and perform tasks such as the following:

1. Create application servers.
2. Create clusters.
3. Manage application servers.
4. Configure transport chains.
5. Set up peer restart and recovery
6. Develop custom services.
7. Define processes for the application server. As part of defining processes, you can define:
   - process execution statements for starting or initializing a UNIX process
   - monitoring policies to track the performance of a process
   - process logs to which standard out and standard error streams write
   - name-value pairs for properties
   - monitoring policies to track the performance of a process
   - name-value pairs for properties
8. Use the Java virtual machine.

After preparing a server, deploy an application or component on the server. See "Preparing to host applications" on page 297 for a sample procedure that you might follow in configuring the application server runtime and resources.

Application servers

Application servers extend a Web server's capabilities to handle Web application requests, typically using Java technology. An application server makes it possible for a server to generate a dynamic, customized response to a client request.

For example, suppose--

1. A user at a Web browser on the public Internet visits a company Web site. The user requests to use an application that provides access to data in a database.
2. The user request flows to the Web server.
3. The Web server determines that the request involves an application containing resources not handled directly by the Web server (such as servlets). It forwards the request to a WebSphere Application Server product.
4. The WebSphere Application Server product forwards the request to one of its application servers on which the application is running.
5. The invoked application then processes the user request. For example:
   - An application servlet prepares the user request for processing by an enterprise bean that performs the database access.
   - The application produces a dynamic Web page containing the results of the user query.
6. The application server collaborates with the Web server to return the results to the user at the Web browser.

The WebSphere Application Server product provides multiple application servers that can be either separately configured processes or nearly identical clones.

Understanding the need for application server naming conventions

Before you install a new WebSphere Application Server for z/OS environment, it is important to plan your naming convention carefully. Your naming convention

should be able to grow with your system when you increase the number of cells, nodes, servers, and clusters later on. Also, your naming convention should accommodate Sysplex and LPAR names, as well as instances such as test, integration, and production stages in your environment.

WebSphere Application Server for z/OS 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.

A WebSphere Application Server for z/OS 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.

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 application servers

For the Network Deployment and z/OS products, you can create a new application server using either the **createApplicationServer**, **createWebServer**, or the **createGenericServer** wsadmin commands (see the *Administering applications and their environment* PDF), or the **Create New Application Server** wizard in the administrative console. You can also create a new application server when you add a cluster member to a server cluster.

With WebSphere Application Server Version 6.0, you can now upgrade a portion of the nodes in a cell, while leaving others at the older release level. This means that, for a period of time, you may be managing servers that are at the current release and servers that are running the newer release in the same cell. In this mixed environment, there are restrictions on what you can do with servers at the older release level. They are:

- You can only create new server definitions on nodes that are running WebSphere Application Server Version 6.0.
- When you create a new server definition, you must use a server configuration template, and that template must be created from a WebSphere Application Server Version 6.0 server instance. You cannot create (or use) a template from a WebSphere Application Server Version 5.x server instance.

There are no restrictions on what you can do with the servers running on the newer release level.

The steps below describe how to use the Create New Application Server page.

1. Go to the Application Servers page and click **New**. This brings you to the Create New Application Server page. You can also create the new application server using the wsadmin **createApplicationServer** command. For information, see Commands for the AdminTask object.
2. Follow the instructions on the Create New Application Server page and define your application server.
   a. Select a node for the application server.
   b. Type in a name for the application server. The name must be unique within the node.
   c. Select a template to be used in creating the new server. You can use a default application server template for your new server or use an existing application server as a template. The new application server will inherit all properties of the template server.
   d. Select whether the new server will have unique ports for each HTTP transport. By default, this option is enabled. If you select this option, 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, ensure that the default port values do not conflict with other servers on the same physical machine.
   e. If you create the new server using an existing application server as a model, select whether to map applications from the existing server to the new server. By default, this option is disabled.
3. To use multiple language encoding support in the administrative console, configure an application server with UTF-8 encoding enabled.

The new application server appears in the list of servers on the Application Servers page.

Note that the application server created has many default values specified for it. An application server has many properties that can be set and creating an application server on the Create New Application Server page specifies values for only a few of the important properties. To view all of the properties of your application server and to customize your application server further, click on the name of your application server on the Application Servers page and change the settings for your application server as needed.

Configuring application servers for UTF-8 encoding

To use multiple language encoding support in the administrative console, you must configure an application server with UTF-8 encoding enabled.

1. Create an application server or use an existing application server.
2. On the Application Server page, click on the name of the server you want enabled for UTF-8.
3. On the settings page for the selected application server, under Server Infrastructure, click **Java and Process Management > Process Definition**.
4. On the Process Definition page, click **Java Virtual Machine**.
5. On the Java Virtual Machine page, specify `-Dclient.encoding.override=UTF-8` for **Generic JVM Arguments** and click **OK**.
6. Click **Save** on the console task bar.

7. Restart the application server.

Note that the autoRequestEncoding option does not work with UTF-8 encoding enabled. The default behavior for WebSphere Application Server is, first, to check if charset is set on content type header. If it is, then the product uses content type header for character encoding; if it is not, then the product uses character encoding set on server using the system property default.client.encoding. If charset is not present and the system property is not set, then the product uses ISO-8859-1. Enabling autoRequestEncoding on a Web module changes the default behavior: if charset it not present on an incoming request header, the product checks the Accept-Language header of the incoming request and does encoding using the first language found in that header. If there is no charset on content type header and no Accept language header, then the product uses character encoding set on server using the system property default.client.encoding. As with the default behavior, if charset is not present and the system property is not set, then the product uses ISO-8859-1.

Default server values for WebSphere Application Server for z/OS Version 5

Purpose

The following table lists the default server values for WebSphere Application Server for z/OS Version 5.

Parameters

Table 3. Default server values for WebSphere Application Server for z/OS V5

Server	Server short name (long name)	Cluster transition name / Application environment name	Run-time controller start procedure name	Run-time servant start procedure name	Subsystem type	Start parameter	Limit on starting server address space for a subsystem instance
Application Server	BBOS001	BBOC001	BBO5ACR	BBO5ASR	CB	JOBNAME=&IWMSSNM.S, ENV=*cellname.nodename*.&IWMSSNM	No limit
Deployment manager (ND only)	BBODMGR	BBODMGR	BBO5DCR	BBO5DSR	CB	JOBNAME=&IWMSSNM.S, ENV=*cellname.nodename*.&IWMSSNM	No limit
Location service daemon	BBODMNB (base) or BBODMNC (ND)	-	-	-	-	-	-
JMS server	BBOJ001	-	-	-	-	-	-
Node agent (ND only)	BBON001	-	-	-	-	-	-

Managing application servers

To view information about an application server, use the Application Servers panel on the administrative console.

You can use the **Application Servers** panel of the administrative console, the Getting started with scripting, or command line tools such as `startServer` and `stopServer` to manage your application servers.

1. Access the Application Servers page. Click **Servers > Application Servers** in the console navigation tree.
2. View information about application servers.

   The Application Servers page lists application servers in the cell and the nodes holding the application servers. The Network Deployment product also shows the status of the application servers. The status indicates whether a server is started, stopped, or unavailable.

   To view additional information about a particular application server or to further configure an application server, click on the application server name under **Name**. This accesses the settings page for an application server.

   To view product information for an application server:

   a. Verify that the application server is running.
   b. Display the **Runtime** tab on the settings page for an application server.
   c. Click **Product Information**.

   The Product Information page displayed lists the WebSphere Application Server products installed for the application server, the version and build levels for the products, the build dates, and any interim fixes applied to the application server.
3. Create an application server.
   a. Click **Servers > Application Servers** in the console navigation tree to access the Application Servers page.
   b. Click **New**, and follow the instructions on the Create New Application Server page.
4. Start your application server.
5. Monitor the running of application servers.
6. Stop your application server.
7. Delete an application server.
   a. Click **Servers > Application Servers** in the console navigation tree to access the Application Servers page.
   b. Place a checkmark in the check box beside an application server to delete it.
   c. Click **Delete**.
   d. Click **OK** to confirm the deletion.

Server collection

Use this page to view information about and manage application servers, generic servers, Java Message Service (JMS) servers, and Web servers.

Application Servers

The Application Servers page lists the application servers in the cell. You can use this page to create new application servers, create application server templates, or delete existing application servers. You can also use this page to start and stop these application servers.

To view this administrative console page, click **Application Servers**.

Generic Servers

The Generic Servers page lists the generic servers in the cell. You can use this page to create new generic servers, create generic server templates, or delete existing generic servers. You can also use this page to start and stop these generic servers.

The Network Deployment product also shows the status of the generic servers. The status indicates whether a server is running, stopped, or encountering problems.

You can use this page to add or delete application servers.

To view this administrative console page, click **Generic Servers**.

Java Message Service (JMS) Servers

The JMS Servers page lists the JMS servers in the cell. You can use this page to start and stop these JMS servers.

Each JMS server provides the functions of the JMS provider for a node in your administrative domain. There can be at most one JMS server on each node in the administration domain, and any application server within the domain can access JMS resources served by any JMS server on any node in the domain.

Note: JMS servers apply only to WebSphere Application Server Version 5.x nodes. You cannot create a JMS server on a node that is running WebSphere Application Server 6.0, but the existing Version 5.x JMS servers will continue to be displayed, and you can modify their properties. You can also delete Version 5.x JMS servers.

To view this administrative console page, click **JMS Servers**.

Web Servers

The Web Servers page lists the Web servers in your administrative domain. You can use this page to generate and propagate a Web server plug-in configuration file, create new Web servers, create new Web server templates, or delete existing Web servers. You can also use this page to start and stop these Web servers.

To view this administrative console page, click **Web Servers**.

Name:

Specifies a logical name for the server. For WebSphere Application Server, server names must be unique within a node.

For WebSphere Application Server for z/OS, 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 name of the node holding the server.

Version:

Specifies the version of the WebSphere Application Server product on which the server runs.

Status:

Indicates whether the server is started or stopped. (Network Deployment only)

Note that 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.

Application server settings

An application server is a server which provides services required to run enterprise applications. Use this page to view or change the settings of an application server instance.

To view this administrative console page, click **Servers > Application 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 server. Server names must be unique within a node. However, for multiple nodes within a cluster, you may have different 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. WebSphere Application Server uses the server name for administrative actions, such as referencing the server in scripting.

If you are running on a z/OS platform, this property is sometimes called the long name.

Data type	String
Default	server1

Short name:

This setting applies to the z/OS platform. The short name specifies the short name of the server and must be unique within a cell. 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), started task control, and others.

The name is 1-8 characters, alpha-numeric or national language. It cannot start with a number.

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.

Converting a 7-character server short name to 8 characters:

By default, WebSphere Application Server for z/OS assumes you will be 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 by performing the tasks below. **Before doing this, it is important to consider the following:**

- 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 WebSphere Application Server for z/OS 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.

  Note that this consideration does not apply to you 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 your applications and their environment* PDF for more information.

Steps for this task:

1. Change the 7-character server short name to the 8-character name you wish to use:
   a. Navigate to **Servers > Application Servers >***server_name*.

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 servant's start command arguments to use the new 8-character name. Note that if you are reconfiguring a node agent you can skip this step because neither has an associated servant process.
   a. Navigate to **Servers > Application Servers >** *server_name* **> Server Infrastructure > Java and process management > Process Definition >** ***Process type***.
   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**.

Run in development mode:

Enabling this option may reduce the startup time of an application server. This may include JVM settings such as disabling bytecode verification and reducing JIT compilation costs. Do not enable this setting on production servers. This setting is only available on application servers running WebSphere Application Server Version 6.0 and later.

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 will not be started in development mode. Setting this option to `true` specifies that the server will be started in development mode (with settings that will speed server startup 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 the server be started using multiple threads. Setting this option to `false` specifies that the server will not be started in using multiple threads (which may lengthen startup time).

Note that the order in which the applications start depends on the weights you assigned to each them. Applications that have the same weight are started in parallel. You set an application's weight with the *Starting weight* option on the **Applications > Enterprise Applications >** *application_name* page of the Administrative Console. For more information about the *Starting weight* option, see the *Developing and deploying applications* PDF.

Data type	Boolean
Default	true

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 should search in the parent class loader or in the application class loader first to load a class. The standard for Developer Kit class loaders and WebSphere Application Server class loaders is `Parent first`.

If you select `Parent last`, 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 native operating system's process ID for this server.

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

Cell name:

The name of the cell in which this server is running.

The Cell name property is read only.

Node name:

The name of the node in which this server is running.

The Node name property is read only.

State:

The run-time execution state for this server.

The State property is read only.

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 > Application Servers >***server_name* **> Communications > Ports**.

Note that 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.

For the z/OS platform, you can view this administrative console page by clicking one of the following paths:

- **Servers > Application Servers >***server_name* **> Ports >***end_point_name*
- **Servers > 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 a radio button to:

Well-known Port
: select a previously defined port from the drop down list

User-defined Port
: create a port with a new name by entering the name in the text box

Data type String

The following ports apply to the z/OS platform only:

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.
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.

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.

Data type Integer
Default None

Range	1-65536

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 a **Custom Properties** link.

Name:

Specifies the name (or key) for the property.

Do not start your property names with `was.` because this prefix is reserved for properties that are predefined in WebSphere 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.

For Network Deployment and the z/OS platform, you can view this administrative console page by clicking one of the following paths:

- **Servers > Application Servers >***server_name*. Then, under Server Infrastructure, click **Administration** > **Custom Properties** *custom_property*
- **Servers >JMS Servers** > *server_name*. Then, under Server Infrastructure, click **Administration** > **Custom Properties** *custom_property*

Name:

Specifies the name (or key) for the property.

Do not start your property names with `was.` because this prefix is reserved for properties 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

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** > **Application Servers** > *server name*. Then, under Server Infrastructure, click **Administration** > **Server Components** > **JMS Servers**.

Short name:

Specifies the short name of the JMS queue manager.

The name is 1-4 characters, alpha-numeric 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 through the WebSphere ISPF Customization Dialog.

Data type String

Server component collection

Use this page to view information about and manage server component types such as application servers, messaging servers, or name servers.

To view this administrative console page, click **Servers > Application Servers >***server_name*. Then, under Server Infrastructure, click **Administration > Server Components**.

Type:

Specifies the type of internal server.

Server component settings

Use this page to view or configure a server component instance.

To view this administrative console, click **Servers > Application Servers >***server_name*. Then, under Server Infrastructure, 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

Thread pool collection

Use this page to select or create a group of threads that an application server uses. Requests are sent to the server through any of the HTTP transports. A thread pool enables components of the server to reuse threads to eliminate the need to create new threads at run time. Creating new threads expends time and resources.

To view this administrative console page, click **Servers > Application Servers >***server_name* **> Thread Pools**. (You can reach this page through more than one navigational route.)

Thread pool settings

Use this page to configure a group of threads that an application server uses. Requests are sent to the server through any of the HTTP transport channels or HTTP transports. A thread pool enables components of the server to reuse threads to eliminate the need to create new threads at run time. Creating new threads expends time and resources.

To view this administrative console page, click **Servers > Application Servers >***server_name* **> Thread Pools**, then select the thread pool. (You can reach this page through more than one navigational route.)

Name:

The name of the thread pool to create. The name must be unique within the server.

Data type String

Description:

A text description of the thread pool.

Data type String

Minimum size:

Specifies the minimum number of threads to allow in the pool.

Data type	Integer
Default	10

Maximum size:

Specifies the maximum number of threads to allow in the pool.

If your Tivoli Performance Viewer shows the Percent Maxed metric to remain consistently in the double digits, consider increasing the Maximum size. The Percent Maxed metric indicates the amount of time that the configured threads are used. If there are several simultaneous clients connecting to the server-side ORB, increase the size to support up to 1000 clients.

Data type	Integer
Default	50
Recommended	50 (25 on Linux systems)

Thread inactivity timeout:

Specifies the number of milliseconds of inactivity that should elapse before a thread is reclaimed. A value of 0 indicates not to wait and a negative value (less than 0) means to wait forever.

Note: The administrative console does not allow you to set the inactivity timeout to a negative number. To do this you must modify the value directly in the *server.xml* file.

Data type	Integer
Units	Milliseconds
Default	3500

Allow thread allocation beyond maximum thread size:

Specifies whether the number of threads can increase beyond the maximum size configured for the thread pool.

Data type	Boolean
Default	Not enabled (false)

Server Instance Settings

Use this page to support Servant instance settings. This support controls the number of servant processes.

To view this administrative console page, click **Servers** > **Application Servers** > *server name*. Then, under Server Infrastructure, 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 server has the following:

- A minimum number of servant process instances as specified on the Minimum Number of Instances setting
- A maximum number of servant process instances as specified on the Maximum Number of Instances setting

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.

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 WebSphere Application Server administrative domain, although it is not a server that is supplied by the WebSphere Application Server 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 > 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. WebSphere Application Server 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 WebSphere 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

Changing the values of variables referenced in BBOM0001I messages

BBOM0001I messages are issued during server startup, and indicate the WebSphere Application Server configuration settings. Unless otherwise indicated, these variables apply to all servers (deployment managers, node agents, JMS servers, and application servers).

Some of these setting can not be changed. The ones you can change, must be changed using the administrative console. Changes made directly to the server's was.env file will be overridden the next time the server is started.

Use the following table to determine which WebSphere Application Server variable, custom property or administrative console field must be updated to change the value of a specific internal variable.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields

Internal variable name	How to change indicated value	Comments
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 _password	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **client_protocol_password** property and specify a different value.	See Application server z/OS custom properties for additional information.
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_protocol_user	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **client_protocol_user** property and specify a different value.	See Application server z/OS custom properties for additional information.
client_ras_ logstreamname	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **client_ras_logstreamname** property and specify a different value.	See Application server z/OS custom properties for additional information.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

clustered_server	User cannot change. A non-clustered server can be promoted to a clustered server, but the reverse action cannot be performed.	Value derived from the cluster name setting specified using the administrative console.
com_ibm_security_ SAF_unauthenticated	In the administrative console, click **Security** > **Global Security**. Under Additional Properties. click**Custom Properties** > **New**. Add the **com_ibm_security_SAF_unauthenticated** property and specify a different value.	If a value is not specified or is not a valid SAF userid, the ID specified by security.remote.identity is used. (The default is WSGUEST.)
com_ibm_security_ SAF_EJBROLE_ Audit_Messages_ Suppress	In the administrative console,click **Security** > **Global Security**. Under Additional Properties. click **Custom Properties** > **New**. Add the **com_ibm_security_SAF_EJBROLE_Audit_Messages_Suppress** property and specify a different value.	
com_ibm_CSI_claim _ssl_sys_v2_timeout	User cannot change.	This variable has been deprecated.
com_ibm_CSI_claim _ssl_sys_v3_timeout	User cannot change.	
com_ibm_CSI_ claimClient Authenticationtype	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound authentication**.	
com_ibm_CSI_ claimClient Authentication Required	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound authentication**.	
com_ibm_CSI_ claimClient Authentication Supported	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound authentication**.	
com_ibm_CSI_ claimIdentity AssertionSupported	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound authentication**.	
com_ibm_CSI_ claimIdentity AssertionTypeCert	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound authentication**.	
com_ibm_CSI_ claimIdentity AssertionTypeDN	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound authentication**.	
com_ibm_CSI_ claimIdentity AssertionTypeSAF	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound authentication**.	
com_ibm_CSI_ claimKeyringName	In the administrative console, click **Security** > **SSL** > **SSL Configuration Repertoires** > *Alias* > **Key File Name**.	Repertoire is pointed to by the **Security** > **Authentication ProtocolCSIv2 Inbound Transport** navigation in the administrative console.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

com_ibm_CSI_ claimMessage IntegrityRequired	User cannot change.	
com_ibm_CSI_ claimMessage IntegritySupported	User cannot change.	
com_ibm_CSI_ claimSecurity CipherSuiteList	In the administrative console, click **Security** > **SSL** > **SSL Configuration Repertoires** > *Alias* > **Cipher Suites** .	Repertoire is pointed to by the **Security** > **Authentication ProtocolCSIv2 Inbound Transport** navigation in the administrative console.
com_ibm_CSI_ claimSecurityLevel	In the administrative console, click **Security** > **SSL** > **SSL Configuration Repertoires** > *Alias* > **Security Level**.	Repertoire is pointed to by the **Security** > **Authentication ProtocolCSIv2 Inbound Transport** navigation in the administrative console.
com_ibm_CSI_ claimStateful	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol**. Then select either **CSIv2 inbound authentication** or **CSIv2 outbound authentication** .	
com_ibm_CSI_ claimTransport AssocSSLTLSRequired	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound transport** .	
com_ibm_CSI_ claimTransport AssocSSLTLS Supported	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound transport** .	
com_ibm_CSI_ claimTLClient Authentication Required	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound authentication**.	
com_ibm_CSI_ claimTLClient Authentication Supported	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 inbound authentication**..	
com_ibm_CSI_ perform_ssl_ sys_v2_timeout	User cannot change.	This variable has been deprecated.
com_ibm_CSI_ perform_ssl_ sys_v3_timeout	In the administrative console, click **Security** > **SSL** > **SSL Configuration Repertoires** .	Repertoire is pointed to by the **Security** > **Authentication ProtocolCSIv2 Inbound Transport** navigation in the administrative console.
com_ibm_CSI_ performClient Authenticationtype	User cannot change.	

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

com_ibm_CSI_ performIdentity AssertionRequired	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 outbound authentication**.	
com_ibm_CSI_ performIdentity AssertionSupported	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 outbound authentication**.	
com_ibm_CSI_ performKeyring Name	In the administrative console, click **Security** > **SSL** > **SSL Configuration Repertoires** > *Alias* > **Key File Name**.	Repertoire is pointed to by the **Security** > **Authentication Protocol** > **CSIv2 Outbound Transport** navigation in the administrative console.
com_ibm_CSI_ performMessage Confidentiality Required	User cannot change.	
com_ibm_CSI_ performMessage Confidentiality Supported	User cannot change.	
com_ibm_CSI_ performMessage IntegrityRequired	User cannot change.	
com_ibm_CSI_ performMessage IntegritySupported	User cannot change.	
com_ibm_CSI_ performSecurity CipherSuiteList	In the administrative console, click **Security** > **SSL** > **SSL Configuration Repertoires** > *Alias* > **Cipher Suites** .	Repertoire is pointed to by the **Security** > **Authentication Protocol** > **CSIv2 Outbound Transport** navigation in the administrative console.
com_ibm_CSI_ performSecurity Level	In the administrative console, click **Security** > **SSL** > **SSL Configuration Repertoires** > *Alias* > **Security Level**.	Repertoire is pointed to by the **Security** > **Authentication Protocol** > **CSIv2 Outbound Transport** navigation in the administrative console.
com_ibm_CSI_ performStateful	User cannot change.	
com_ibm_CSI_ performTransport AssocSSLTLS Required	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 outbound transport**.	
com_ibm_CSI_ performTransport AssocSSLTLS Supported	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 outbound transport**.	

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

com_ibm_CSI_ performTLClient Authentication Required	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 outbound transport**.	
com_ibm_CSI_ performTLClient Authentication Supported	In the administrative console, click **Security > Global security**. Under Authentication, click **Authentication protocol > CSIv2 outbound transport**.	
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 _claimClient Authentication	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 _claimSecurityCipher SuiteList	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.	This variable has been deprecated.
com_ibm_HTTP_ claim_ssl_sys_ v3_timeout	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports** > *ssl_transport*.	Transport option only appears if you have an transport defined for your system.
com_ibm_HTTP_ claim_sslEnabled	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports** > *ssl_transport*.	Transport option only appears if you have an transport defined for your system.
com_ibm_HTTP_ claimClient Authentication	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports**.	Transport option only appears if you have an transport defined for your system.
com_ibm_HTTP_ claimKeyringName	In the administrative console, click **Server** > **Application Server** > *server* > **Web Container Settings > Web container** > **HTTP transports**.	Transport option only appears if you have an transport defined for your system.
com_ibm_HTTP_ claimSecurityCipher SuiteList	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports**	Transport option only appears if you have an transport defined for your system.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

com_ibm_HTTP_ claimSecurity Level	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports**.	Transport option only appears if you have an transport defined for your system.
com_ibm_Server _Security_Enabled	In the administrative console, click **Servers** > **Application Servers** > *server* > **Security** > **Sever 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	User cannot change. However an additional path can be appended to the specified path. To do this, in the administrative console click **Servers** > **Application Servers** > *server* > **Java and Process Management > Process Definitions** > **Java Virtual Machine** and specify the classpath to be appended.	Specifies the classpath used by the conroller's JVM.
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 **Server** > **Application Server** > *server* > **Java and Process Management > Process Definitions**, and specify the libpath.	
control_region_ mdb_request_ timeout (application server only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **control_region_mdb_request_timeout** property and specify a different value.	See Application server z/OS custom properties for additional information.
control_region_ security_enable_ trusted_applications (application server only)	In the administrative console, click **Security** > **Global Security** > **Custom Properties**. Select the **EnableTrustedApplications** property and specify a new value.	
control_region_ssl _thread_pool_size		Should only be changed under the direction of IBM Support personnel.
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.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

control_region_use_ java_g	In the administrative console, click **Servers** > **Application Servers** > *server* > **Java and Process Management > Process Definitions 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** > **Application Servers** > *server*. Under Container Settings, click **Container Services > ORB Service**. Specify a different value in the **WLM Timeout** field.	
daemon_group _name	User cannot change.	
daemon_protocol_ iiop_listenIPAddress	In the administrative console, click **Environment** > **WebSphere Variables** > **New**. Add the **DAEMON_protocol_iiop_listenIPAddress**. In the Value field, type * to specify 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.	
nls_language	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **nls_language** property and specify a different value.	See Application server z/OS custom properties for additional information.
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**.	
protocol_accept_http _work_after_min_srs	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_accept_http_work_after_min_srs** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_bboc_log_ response_failure	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_bboc_log_response_failure** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_bboc_log_ return_exception	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_bboc_log_return_exception** property and specify a different value.	See Application server z/OS custom properties for additional information.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

protocol_giop_level_ highest	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_giop_level_highest** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_backlog	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_backlog** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_cloneid	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container** > **HTTP transports** > **New**. Add the **HttpSessionCloneId** property and specify a different value.	See HTTP transport custom properties for additional information.
protocol_http_ defaultIdentity	User cannot change.	No longer used.
protocol_http_ enableSSLTracking	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Session Management**. Check/uncheck **Enable SSL ID tracking** for the Session tracking mechanism property and specify a different value.	Indicates that the SSL Session ID is to be used as the HTTP Session ID.
protocol_http_large_ data_inbound_buffer	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_large_data_inbound_buffer** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_large_ data_response_buffer	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_large_data_response_buffer** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_ listenIPAddress	In the administrative console, click **Server** > **Application Servers** > *server* > **Web container settings** > **HTTP transports** > *non_ssl_transport* and specify a new IP address.	See HTTP transport custom properties for additional information.
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_max_ persist_requests	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *non_ssl_transport* > **Custom Properties**. Select the **MaxKeepAliveRequests** property and specify a new value.	See HTTP transport custom properties.
protocol_http_port	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *non_ssl_transport* and specify a new port.	
protocol_http_session _cookie _name	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Session Management** > **Enable Cookies** > **Cookie Name** and specify a new name.	
protocol_http_session _id_length	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > Custom Properties** > **New**. Add the **HttpSessionIdLength** property and specify a different value.	See Session management custom properties.
protocol_http_session _no_additional_info	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container** > **Custom Properties** > **New**. Add the **NoAdditionalSessionInfo** property and specify a different value.	See Session management custom properties.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

protocol_http_session _persistent	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Session Management** > **Additional Properties** > **Distributed Environment Settings** and select **Database**.	Indicates if persistent sessions are being used. See Configuring for database session persistence.
protocol_http_session _rewrite_identifier	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Session Management**. Check/uncheck **Enable URL Rewriting** for the **Session tracking mechanism** property.	
protocol_http_tcp_ no_delay	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_tcp_no_delay** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_timeout _input (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **HTTP transports** > *non_ssl_transport* > **Custom Properties**. Select the **ConnectionIOTimeOut** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_http_timeout _output (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container > HTTP transport >** *non_ssl_transport* > **Custom Properties**. Select the **ConnectionResponseTimeout** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_http_timeout _output_recovery (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_timeout_output_recovery** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_timeout _persistentSession (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container > HTTP transport >** *non_ssl_transport* > **Custom Properties**. Select the **ConnectionKeepAliveTimeout** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_http_ transactionClass	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_http_transactionClass** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_ transport_class_ mapping_file	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **z/OS additional Settings**.	
protocol_http_ transport_ network_qos	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **z/OS additional Settings**.	
protocol_http_ trusted_proxy	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *non_ssl_transport* > **Custom Properties**. Select the **TrustedProxy** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_https_ backlog	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_https_backlog** property and specify a different value.	See Application server z/OS custom properties for additional information.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

protocol_https_ default_identity	User cannot change.	No longer used.
protocol_https_ listenIPAddress	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *ssl_transport* and specify a new IP address.	This variable is the IP address over which the J2EE server is to receive requests. It causes the server to bind to this specific IP address rather than to the default, which is to bind to all addresses.
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_max_ persist_requests	In the administrative console, click **Servers** > **Application Servers** > *server* >**Web Container Settings** > **Web container > HTTP transports** > *ssl_transport* > **Custom Properties**. Select the **MaxKeepAliveRequests** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_https_mutua l_auth_cbind_check	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *ssl_transport* > **Custom Properties**. Select the **MutualAuthCBindCheck** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_https_port	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *ssl_transport* and specify a new port	
protocol_https_tcp _no_delay	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_https_tcp_no_delay** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_https_ timeout_input (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *ssl_transport* > **Custom Properties**. Select the **ConnectionIOTimeOut** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_https_ timeout_output (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *ssl_transport* > **Custom Properties**. Select the **ConnectionResponseTimeout** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_https_ timeout_output_ recovery (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_https_timeout_ output_recovery** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_https_ timeout_ persistentSession (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *ssl_transport* > **Custom Properties**. Select the **ConnectionKeepAliveTimeout** property and specify a new value.	See HTTP transport custom properties for additional information.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

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_https_ transport_class_ mapping_file	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **z/OS additional settings**.	
protocol_https_ transport_network _qos	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings > Web container** > **z/OS additional settings**.	
protocol_https_ trusted_proxy	In the administrative console, click **Servers** > **Application Servers** > *server* > **Web Container Settings** > **Web container > HTTP transports** > *ssl_transport* > **Custom Properties**. Select the **TrustedProxy** property and specify a new value.	See HTTP transport custom properties for additional information.
protocol_iiop_ backlog_ssl	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_iiop_backlog_ssl** property and specify a different value	See Application server z/OS custom properties for additional information.
protocol_iiop_backlog	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_iiop_backlog** property and specify a different value	See Application server z/OS custom properties for additional information.
protocol_iiop_ daemon_listenIP Address	In the administrative console, click **System Administration > Node groups > sysplex node group > z/OS Location Service** . Specify the new IP address.	
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_ listenIPAddress	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Communications, click **Ports**.	
protocol_iiop_ local_timeout	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Container Settings, click **Container Services > ORB Service**.	
protocol_iiop_no _local_copies	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Container Settings, click **Container Services > ORB Service**.	
protocol_iiop_port	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Communications, click **Ports** > **ORB_LISTENER_ADDRESS**.	
protocol_iiop_port _ssl	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Communications, click **Ports** > **ORB_SSL_LISTENER_ADDRESS**.	
protocol_iiop_ propagate_unknown_ service_ctxs	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **protocol_iiop_propagate_unknown_service_ctxs** property and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_iiop_ server_session_ keepalive	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Container Settings, click **Container Services > ORB Service**.	

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

protocol_iiop_ server_session_ keepalive_ssl	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Container Settings, click **Container Services** > **ORB Service**.	
ras_debugEnabled		Should only be changed under the direction of IBM Support personnel.
ras_default_ msg_dd	In the administrative console, click **Environment** > **Manage WebSphere Variables** > **New**. Add the **ras_default_msg_dd** variable.	
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** > **New**. Add the **ras_hardcopy_msg_dd** variable.	
ras_log_ logstreamName	In the administrative console, click **Environment** > **WebSphere Variables** > **New**. Add the **ras_log_logstreamName** variable.	
ras_minorcode_action		Should only be changed under the direction of IBM Support personnel.
ras_time_local	In the administrative console, click **Environment** > **WebSphere Variables** > **New**. Add the **ras_time_local** variable.	
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_minor CodeTraceBacks		Should only be changed under the direction of IBM Support personnel.
ras_trace_output Location	In the administrative console, click **Environment** > **WebSphere Variables** > **New**. Add the **ras_trace_outputLocation** variable.	
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** > **New**. Add the **ras_trace_BufferCount** variable.	
ras_trace_ BufferSize	In the administrative console, click **Environment** > **Manage ebSphere Variables** > **New**. Add the **ras_trace_BufferSize** variable.	

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

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_local_ identity	In the administrative console, click **Security** > **Global Security** > **Custom Properties** > **New**. Add the **security_local_identity** property and specify a different value.	
security_remote_ identity	In the administrative console, click **Security** > **Global Security** > **Custom Properties** > **New**. Add the **security_remote_identity** property and specify a different value.	
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** > **Application Servers** > *server* > **Server Security** > **zSAS Transport** > **SSL Setttings**. Select a different repertoire.	
security_Enable RunAsIdentity	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **security_EnableRunAsIdentity** property and specify a different value.	See Application server z/OS custom properties for additional information.
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)	User cannot change. However an additional path can be appended to the specified path. To do this, in the administrative console click **Servers** > **Application Servers** > *server*. Under Server Infrastructure click **Java and Process Management > Process Definitions** > **Servant** > **Java Virtual Machine** > **Classpath**, and specify the classpath to be appended.	Specifies the classpath used by the servant region's JVM.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

server_region_ dynapplenv_jclparms (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure click **Java and Process Management > Process Definitions** > **servant** > **Start Command Args**. 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** > **Application Servers** > *server*. Under Server Infrastructure click **Java and Process Management > Process Definitions** > **servant** > **Start Command**. Specify the new JCL procedure.	When dynamic applenv is being 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 **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **server_region_jvm_localrefs** property and specify a different value.	See Application server z/OS custom properties for additional information.
server_region_ jvm_logfile (application server and deployment manager only)	User cannot change	
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)	**Servers** > **Application Servers** > *server*. Under Server Infrastructure click **Java and Process Management > Process Definitions** > **Servant** > **Environment Entries** > **LIBPATH**	Specifies the libpath for the servant region's JVM
server_region_ recycle_count (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **server_region_recycle_count** property and specify a different value.	See Application server z/OS custom properties for additional information.
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** > **Application Servers** > *server*. Under Server Infrastructure click **Java and Process Management > Process Definitions** > **Servant** > **Java Virtual Machine** and check the **Debug Mode** checkbox.	Indicates if the servant region's JVM should use the debug JVM (java_g). Should only be used under the direction of IBM support.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

server_region_ workload_profile (application server and deployment manager only)	n the administrative console, click **Servers** > **Application Servers** > *server*. Under Container Settings, click **Container Services > ORB Service**.	
server_specific_name	User cannot change.	
server_specific_ short_name	In the administrative console, click **Servers** > **Application Servers** > *server* > **Short Name**.	
server_specific_uuid	User cannot change.	Specifies the unique identifier for this server
server_start_wait_ for_initialization _Timeout	In the administrative console, click **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **server_start_wait_for_initialization_Timeout** property and specify a different value. For a node agent, in the administrative console, click **System Administration > Node Agents >** *nodeagent* > **Administration Services > Custom Properties**. Add the **server_start_wait_for_initialization_Timeout** property and specify a different value.	See Application server z/OS custom properties for additional information.
server_SMF_ container_ activity_enabled (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Custom Properties** > **New**. Add the **server_SMF_container_activity_enabled** property and specify a different value.	See 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** > **Application Servers** > *server*. Under Server Infrastructure, click **Administration > Custom Properties** > **New**. Add the **server_SMF_container_interval_enabled** property and specify a different value.	See 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** > **Application Servers** > *server*. Under Server Infrastructure, click **Administration > Custom Properties** > **New**. Add the **server_SMF_interval_length** property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_ server_activity _enabled (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server* . Under Server Infrastructure, click **Administration > Custom Properties** > **New**. Add the **server_SMF_server_activity_enabled** property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

server_SMF_server _interval_enabled (application server and deployment manager only)	In the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure, click **Administration > Custom Properties** > **New**. Add the **server_SMF_server_interval_enabled** property and specify a different value.	See 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.
transaction_ defaultTimeout (application server only)	In the administrative console, click **Servers** > **Application Servers** > *server* > **Container Services > Transaction Service** > **Transaction lifetime timeout**.	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** > **Application Servers** > *server* > **Container Services > Transaction Service** > **Maximum Transaction Timeout** and specify a different value.	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 **Servers** > **Application Servers** > *server* > **Server infrastructure > Administration > Custom Properties > New**. Add the **transaction_recoveryTimeout** property and specify a different value.	See Application server z/OS custom properties for additional information.
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** > **Application Servers** > *server*. Under Server Infrastructure, click **Java and Process Management > Server Instance**. For a deployment manager, in the administrative console, click **System administration** > **deployment manager**. Under Server Infrastructure, click **Java and Process Management > Server Instance**.	

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields (continued)

wlm_maximum SRCount (application server and deployment manager only)	For an application server, in the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure, click **Java and Process Management > Server Instance**. For a deployment manager, in the administrative console, click **System administration** > **deployment manager**. Under Server Infrastructure, click **Java and Process Management > Server Instance**.	
wlm_minimum SRCount (application server and deployment manager only)	For an application server, in the administrative console, click **Servers** > **Application Servers** > *server*. Under Server Infrastructure, click **Java and Process Management > Server Instance**. For a deployment manager, in the administrative console, click **System administration** > **deployment manager**. Under Server Infrastructure, click **Java and Process Management > Server Instance**.	

Starting servers

Starting a server starts a new server process based on the process definition settings of the current server configuration. The node agent for the node on which the Application Server resides must be running before you can start the Application Server.

If you need to restart a server, follow the directions in this article for starting servers. The procedure that applies to starting servers also applies to restarting servers.

Note: If you created a new server definition using a base WebSphere Application Server, you cannot start, stop, or manage the new server using the original base Application Server.

Note: After you start an Application Server, other processes might not discover the running Application Server immediately. Application Servers are discovered by the node agent. Node agents are discovered by the deployment manager. Node agents usually can discover local Application Servers quickly but it can take a deployment manager from 2 to 60 seconds to discover a node agent.

Note: If you are using clusters, note that the **Initial State** property of the Application Server subcomponent (**Servers** > **Application Server** > *server_name* > **Administration** > **Server Components** > **Application Server**) is not intended to be used to control the state of individual servers in the cluster at the time the cluster is started. It is intended only as a way to control the state of the Application Server subcomponent of a server. It is best to start and stop the individual servers of a cluster using the Server options of the Administrative Console or command line commands (**startServer** and **stopServer**).

There are several options for starting an Application Server:

- If you are using z/OS, use the START *appserver_proc_name* command to start an Application Server from the command line.

  If the node agent for the node on which the Application Server resides is not running, run the START *nodagent_proc_name* command and then run the START *appserver_proc_name* command.

- Start an Application Server using the administrative console.
  1. Click **Servers > Application Servers** from the administrative console navigation tree to go to the Application Server page.
  2. If the node agent for the node on which the Application Server resides is not running, click **Restart** or **Restart all Servers on Node** on the Node Agents page (**System Administration > Node agents**) to start the node agent.

     If the node agent does not start, run the startNode command and then run the startServer command. Once a node agent completely stops running and remains stopped, you cannot remotely start the node agent from the Node Agents page. You must run the startNode command to start the node agent on the node where it runs.
  3. Select the Application Server and click **Start**.
  4. View the **Status** value and any messages or logs to see whether the Application Server starts.
- Start an Application Server for tracing and debugging.

  To start the Application Server with standard Java debugging enabled:
  1. Click **Servers > Application Servers** from the administrative console navigation tree. Then, click the Application Server whose processes you want to trace and debug, then **Java and Process Management > Process Definition > Java Virtual Machine**.
  2. On the Java Virtual Machine page, place a checkmark in the check box for the **Debug Mode** setting to enable the standard Java debugger. If needed, set debug arguments. Then, click **OK**.
  3. Save the changes to a configuration file.
  4. Stop the Application Server.
  5. Start the Application Server again as described previously.
- Start an Application Server for Websphere Application Server for z/OS from the MVS console. A typical WebSphere Application Server for z/OS run time includes two nodes:
  - Deployment Manager node. This includes a location service daemon and a Deployment Manager with a controller and any number of servants
  - Application Server node. This includes a location service daemon, a node agent with a controller, and an Application Server with a controller and any number of servants

  Before you start any of the Application Servers, you must verify that all resource managers that are required by your application (DB2, CICS, and so on) are available. You must also start all prerequisite subsystems.
  1. 1. To start a server, issue the following command: S controlregionprocname, JOBNAME= server_shortname, ENV= cell_shortname.Node_shortname.Server_shortname where:

     controlregionprocname
     Is the JCL procedure name in the proclib that is used to start the server.

     server_shortname
     Is the short name of the server (or the step name used to start the proc). This allows you to identify the address space that is running when you view it in the SDSF panels.

     cell_shortname.Node_shortname.Server_shortname
     The ENV variable is a concatenation of the cell short name, the node short name, and the server short name.

Note: This command should be all upper case and should all be on one line (it is shown above on two lines because of space considerations only). For example,

```
S BBO5ACR.BBOS001,ENV=SY1.SY1.BBOS001
```

The following messages indicate that the controller is up:

```
$HASP100 BBO5ACR  ON STCINRDR
$HASP373 BBO5ACR  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.
```

2. The Controller Region automatically starts the daemon by issuing a command that looks like:

```
S <dmn_proc>,JOBNAME=<dmn_jobname>,
       ENV=<cell_shortname.Node_shortname.daemon_instancename>
```

Note: This command should all be on one line (it is shown above on two lines because of space considerations only).

Following is an example of the messages that are displayed during daemon startup:

```
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 BBO5DMN
        WITH JOBNAME BBO5DMN.
$HASP100 BBO5DMN  ON STCINRDR
$HASP373 BBO5DMN  STARTED

BBOO0007I WEBSPHERE FOR Z/OS DAEMON BBODMNB/SY1/BBODMNB/SY1 IS STARTING.

IEC130I STEPLIB  DD STATEMENT MISSING
ITT102I CTRACE WRITER BBOWTR IS ALREADY ACTIVE.

BBOO0215I PRODUCT 'WAS FOR Z/OS' SUCCESSFULLY REGISTERED WITH IFAED SERVICE.
BBOO0015I INITIALIZATION COMPLETE FOR DAEMON SY1.
```

3. WLM will start servant address spaces with a command that looks like:

```
S <Srv_Reg_Proc>,JOBNAME=<Server_shortname>,
           ENV=<Cell_shortname.Node_shortname.Server_shortname>
```

Note: This command should all be on one line (it is shown above on two lines because of space considerations only).

4. Determine if the Daemon is up. See Determining if the location service daemon is up. If the Daemon is up, go to the next step to start the server.
5. To start a server from the administrative console, click **Servers** > **Application Server** in the console navigation tree to access the Application Server page.

6. Place a checkmark in the check box beside the Application Server that you want to start and click **Start**.
7. View the **Status** value and any messages or logs to verify that the server starts.

Once the server is started, you can install your applications.

Detecting and handling problems with run-time components

You must monitor the status of run-time components to ensure that, once started, they remain operational as needed.

1. Regularly examine the status of run-time components. Browse messages displayed under **Websphere Runtime Messages** in the status area at the bottom of the console. The run-time event messages marked with a red **X** provide detailed information on event processing.
2. If an application stops running when it should be operational, examine the application's status 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 has stopped running, use the Server Cluster page to try restarting 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 run-time components do not restart, reexamine the messages and read information on problem determination to help you to restart the components.

Stopping servers

Stopping an application server stops a server process based on the process definition settings in the current application server configuration.

- Stop the application server from the command line.

  On the z/OS platform, you can use the STOP *appserver_proc_name* command or the STOP *dmgr_proc_name* command.

  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 *WAS_INSTALL_ROOT*`/AppServer/profiles/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 JSPs when you install an application or if you use the JspBatchCompiler function to precompile them before they are invoked.
- Use the administrative console to stop an application server:
  1. Click **Servers > Application Servers** from the administrative console navigation tree to go to the "Server collection" on page 203 page.
  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.

Displaying the status of ARM-registered address spaces including WebSphere Application Server for z/OS servers and server instances

This section describes how to use ARM to display the status of all ARM-registered address spaces (including the address spaces of server instances) in the WebSphere Application Server for z/OS environment.

WebSphere Application Server for z/OS ships with all controllers issuing automatic restart management (ARM) registration commands. If your installation enables ARM, you should read this section.

This section describes how to use ARM to display the status of all ARM-registered address spaces (including the address spaces of server instances) in the WebSphere Application Server for z/OS environment. ARM is used to restart all address spaces that go down, if they are registered with ARM. This does not apply if the address spaces are canceled.

Each WebSphere Application Server for z/OS controller registers with ARM. If a controller terminates abnormally or the system fails, ARM will try to restart the failing address spaces. In doing this, ARM will ensure that dependent address spaces are grouped together and will start in the appropriate order. In general, the default ARM policy will restart WebSphere Application Server for z/OS in place. If using a sysplex, see the *Installing your application serving environment* PDF for setup guidelines to ensure that no cross-system restarts are performed.

Steps for displaying the status of ARM-registered address spaces

Perform the following steps to use ARM to display the status of ARM registered address spaces (including the address spaces of server instances) in the WebSphere Application Server for z/OS environment:

1. Initialize all servers. The bare minimum required for a step is the cmd element. The info element is optional.
2. To display all registered address spaces (including the address spaces of server instances), issue the command: d xcf,armstatus,detail

Converting a 7-character server short name to 8 characters

By default, WebSphere Application Server for z/OS assumes you will be 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 by performing the tasks below. **Before doing this, it is important to consider the following:**

- 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 WebSphere Application Server for z/OS 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.

  Note that this consideration does not apply to you 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 your applications and their environment* PDF

1. Change the 7-character server short name to the 8-character name you wish to use:
   a. Navigate to **Servers > Application Servers >***server_name*.
   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 servant's start command arguments to use the new 8-character name. Note that if you are reconfiguring a node agent, you can skip this step because it does not have an associated servant process.
   a. Navigate to **Servers > Application Servers >** *server_name* **> Server Infrastructure > Java and process management > Process Definition >** ***Process type***.
   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**.

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 > Application servers>** > *server*. Then under **Additional properties**, select **Core group service**.

Click **Save** to save and synchronize your changes with all managed nodes.

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: When selected, 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.

Important: 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

Setting the time zone used by the administrative console for a specified server

You can set the TZ variable to establish the time zone that the administrative console uses for a specific server.

Note: For the TZ variable to function, the JVM property **-DUser.timezone** must not be set.

1. Log onto the administrative console and click **Application Servers** > *server_name* > **Process Definition** > **Control** > **Environment Entities**.
2. On the Custom Properties panel, click **New**.
3. On the Configuration tab, under General Properties, enter **TZ** in the Name field.
4. Enter the appropriate time zone in the Value field. Some examples are *CST*, *BST*, and *ECT*.
5. Optionally, enter a description of the variable and the setting you specified.
6. Click **Save changes to master configuration**.
7. Make sure **Synchronize changes with Nodes** is checked.
8. Click **Save**.

You must restart the server for the change to take effect.

Creating generic servers

There are two types of generic application servers:
- Non-Java applications or processes.
- Java applications or processes

You can use the wsadmin tool or the **Generic servers** panel of the administrative console to create either type.

Note: For the Base WebSphere Application Server product, although you can use the administrative console to create a generic application server definition, you cannot use it 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, use the wsadmin tool.

- **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**. You can then specify the name of the generic server you are creating.
  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 Servers. This will enable you to quickly determine whether to use the **Terminate** or **Stop** button in the administrative console to stop specific application server. You must use the **Terminate** button to stop a generic application server.
  4. Select a template to use in creating the new server. You can use a default application server template for your new server or use an existing application server as a template. The new application server will inherit all properties of the template server. If you create the new server using an existing application server do not enable the option to map applications from the existing server to the new server. This option does not apply for a generic server.
  5. Click **Next**
  6. Click **Finish**. The generic server now appears as an option on the **Generic servers** panel in the administrative console.
  7. On the **Generic servers** panel, 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 name of the non-WebSphere Application Server program that is to be launched when you start this generic server. 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
  10. 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** > **Generic servers**

2. Click **New**. You can then specify the name of the generic server you are creating.
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 Servers. This will enable you to quickly determine whether to use the **Terminate** or **Stop** button in the administrative console to stop specific application server. You must use the **Terminate** button to stop a generic application server.
4. Click **Next**
5. Click **Finish**. The generic server now appears as an option on the **Applications Server** panel in the administrative console.
6. Click **Finish**. The generic server now appears as an option on the **Generic servers** panel in the administrative console.
7. On the **Generic servers** panel, 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 Websphere Application Server's default JVM (${JAVA_HOME}/bin/java), which will be 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**, will be used as the executable target of this Java process. The default for Websphere Application Server 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 will be either a Java class containing a main() method, or the name of an executable JAR file.) The default for Websphere Application Server is **com.ibm.ws.runtime.WsServer**.
12. Click **OK**.

    Note: If the generic server is to run an application server other than the WebSphere Application 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.

You can now start and terminate the generic server whenever you want to start or terminate the non-WebSphere Application Server server or process associated with this server.

Starting and terminating generic servers

This topic describes how to start and terminate generic servers.

If you created a generic server on a Base WebSphere Application Server, you cannot start, terminate, or monitor this server with the Base Application Server administrative console. You must use the wsadmin tool to manage Base generic servers.

Starting generic servers

There are two ways to start a generic server in a Network Deployment environment:

- Use the administrative console:

1. From the administrative console navigation tree, select **Servers > Application Servers**.
2. Select the check box beside the name of the generic server, and then click **Start**.
3. View the **Status** value and any messages or logs to see whether the generic server starts.

- Use the MBean NodeAgent launchProcess operation of the wsadmin tool.

Terminating generic servers

There are two ways to terminate a generic server in a Network Deployment environment:

- Use the administrative console:
  1. From the administrative console navigation tree, select **Servers > Application Servers**.
  2. Select the check box beside the name of the generic server, and then click **Terminate**.
  3. View the **Status** value and any messages or logs to see whether the generic server terminates.

  Note: The **Stop** and **Stop Immediate** buttons on the administrative console do not work for generic servers.
- Use the MBean terminate launchProcess operation of the wsadmin tool.

Generic servers

A generic server is a server that is managed in the WebSphere administrative domain, although it is not a server that is supplied by the WebSphere Application Server 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
- A CORBA server
- A Remote Method Invocation (RMI) 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 server or process.

After you define a generic server, use the Application Server administrative console to start, stop, and monitor the associated non-WebSphere server or process when stopping or starting the applications that rely on them.

Setting up peer restart and recovery

Important: New recovery support was added to WebSphere Application Server Version 6 to specifically support high availability. This recovery support is called *transaction peer recovery*. The *Peer restart and recovery* support, documented in this article, continues to be available for WebSphere Application Server Version 5 and WebSphere Application Server Version 4 workloads. However, Version 5 and Version 4 workloads cannot use new Version 6 features such as JCA 1.5 connectors, WS-AT transactions, or JMS using Jetstream. The support for peer restart and recovery will be removed in a future version of

WebSphere Application Server in favor of peer recovery with the HA manager. For information about transaction peer recovery for high availability, see the *Developing and deploying applications* PDF.

To allow WebSphere Application Server for z/OS 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. 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 for z/OS 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 WebSphere Application Server for z/OS peer restart and recovery. Consult your JTA XAResource Manager's documentation to determine if it supports restarting on an alternate system.

Note: 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 for z/OS Version 5.0 to perform peer restart and recovery for an application server that is running WebSphere Application Server for z/OS Version 5.1.

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.

Note: 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.

Once you have the prerequisites installed, starting a server on a system to which it was not configured will implicitly place it 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 Manger, you need to perform the following steps before starting a server:

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. Under **Additional Properties**, select **File Synchronization Service**.
   d. Under **Additional Properties**, 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 WebSphere Application Server for z/OS 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 WebSphere Application Server for z/OS variable TRANLOG_ROOT to a shared HFS. The TRANLOG_ROOT variable must point to a shared HFS, to which all systems in the WebSphere Application Server for z/OS cell can write. The XA partner log is stored here, and the alternate system must be able to read and update this log.

      Use the administrative console to set the WebSphere Application Server for z/OS variable, TRANLOG_ROOT, to the directory of a shared HFS, to which all systems in the WebSphere Application Server for z/OS cell can write.

      In the administrative console, click **Environment** > **Manage WebSphere Variables**. Then click on the **TRANLOG_ROOT** variable to bring up an new window in which you can specify the directory of the shared HFS.
   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 WebSphere Application Server for z/OS 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 (i.e., the power unexpectedly goes out in your main system). When this happens, a course of restart action you can take is to restart on a peer system in the sysplex, a function called peer restart and recovery. Starting a server on a system to which it was not configured will implicitly place it into peer restart and recovery mode.

The support for peer restart and recovery is in addition to the peer recovery of transactions with the HA manager that was introduced in WebSphere Application Server version 6. The support for peer restart and recovery will be removed in a future version of WebSphere Application Server in favor of peer recovery with the HA manager. See the the *Developing and deploying applications* PDF for more information.

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 WebSphere Application Server for z/OS and related servers on the same system, or
- Can use the WebSphere Application Server for z/OS peer restart and recovery function to restart related servers on an alternate system in the cell. (The application server itself is not a recoverable *resource* manager. It is a recoverable *communication* manager. It has no recoverable locks of its own and it doesn't 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.

InFlight work and presumed abort mode

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, WebSphere Application Server for z/OS and the resource managers go into presumed abort mode. In this mode, the resource managers abort (rollback) 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

Once the timeout occurs, since we were InFlight at the time of the failure, we will rollback the transaction branch.

When local resource managers are involved, RRS will ensure that they are called to perform presumed abort processing. When doing recovery, RRS will work with the resource managers to ensure that the recovery is done properly. When a failure occurs while work is InFlight, RRS will direct the resource managers involved in the local UR to rollback.

The WebSphere Application Server for z/OS runtime always assumes that there is recovery to do. Every time a server comes up, it does something different depending on what mode it is in:

- If the server is running in restart/recovery mode, it checks to see whether there is any recovery required. If so, it 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 it takes on new work. Once it knows what the restart work is, it can begin to take in new work.

Handling new work during recovery

The procedures for the recovery of InFlight and InDoubt work have been described in some detail, but how is new work handled on a recovered server? Once the InDoubt and InFlight work has been completed, the WebSphere Application Server for z/OS server shuts down. A new application server configured for that system may now be started up to accept new work.

Special considerations must be taken to begin new work on a WebSphere Application Server for z/OS using IMS Connect after recovering to an alternate system. Once the recovery has been 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,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, which can be seen in the example below:

```
*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
```

Once this has been completed IMS Connect is ready for new work to be completed on the server.

When might PRR fail to recover servers

The major reason for recovery 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.

When WebSphere Application Server for z/OS 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 WebSphere Application Server for z/OS 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:

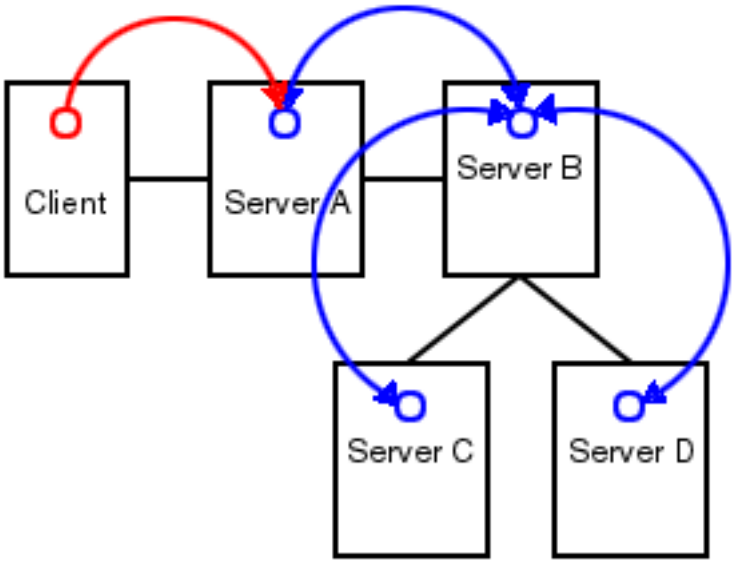
```
RESOURCE MANAGER=BSS00.SY2.BBOASR4A.IBM
RESOURCE MANAGER MUST RESTART ON SYSTEM SY2
```

Recoverable communication manager

WebSphere Application Server for z/OS itself is not a recoverable *resource* manager. It is a recoverable *communication* manager. This means that the Application Server has no recoverable locks of its own and it doesn't need to manage locks nor

manage lock states in a log. Instead, the Application Server needs to make sure that both callers and callees are connected in each of the communications sessions of a distributed transaction.

This example outlines the organization of servers in a recoverable communication manager setup. Use the following diagram to understand the sysplex layout:



Suppose there is a client that talks to server A. Server A then talks to server B, which in turn talks to servers C and D. In this example, server A is the superior, server B is the superior subordinate, and servers C and D are subordinates. With respect to server B, there is a communication session **from** server A and one each **to** servers C and D. OTS logs each of these communication sessions as *recoverable resource references* and prepares them for recovery so that, in the event of a failure, server B can reestablish the communication sessions to these servers.

When server B fails, server A and servers C and D cannot communicate to each other since server B is the intermediary between them. So, when server B recovers, it reads the log and reestablishes the communication sessions to the other servers. Further, once connectivity is reestablished, servers B, C, and D can determine the outcome of the transaction.

Using RRS panels to resolve indoubt units of recovery

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*.

If you receive the console message:

```
BBOT0015D OTS UNABLE TO RESOLVE ALL INCOMPLETE TRANSACTIONS FOR SERVER
string.  REPLY CONTINUE OR TERMINATE.
```

1. Note the server named in *string*..
2. Go to SYSPRINT (the status queue for that server) and search for messages BBOT0019 - BBOT0022 that refer to that server.
3. Follow the resulting messages.
4. RRS does not allow an operator to resolve an indoubt UR if the DSRM for that UR is active at the time, so you need to stop the server. To do this, reply "TERMINATE" to the CONTINUE/TERMINATE WTOR.

The following non-console messages, which can be used to trigger automation, provide details about daemon activities when attempting restart and recovery:

- **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 transaction(s) that could not 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.

Pay particular attention to the URID, XID FormatId, XID Gtrid, and XIDBqual attributes. These pieces of information will be used to manually resolve the relevant units of work via the RRS panels.

Resolving InDoubt units if you receive message BBOT0019W or BBOT0020W:

These two messages indicate that this server could not determine the outcome from its superior. **BBOT0020W**, which describes the resource to which we could not provide an outcome, will always be accompanied by **BBOT0019W** (if you look in the server that is indicated by **BBOT0020W**, you should see a **BBOT0019W** message as well. 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.

Use RRS panels to perform the following steps to view the outcome of other branches in the transaction and set the outcomes of InDoubt branches to match.

1. Select option 3, ″Display/Update RRS Unit of Recovery information″ on the main RRS panel.
2. Attempt to resolve InDoubt URs with an outcome identical to the outcome of other branches of the same transaction (unless those other branches show an outcome of ″InPrepare,″ in which case you should resolve to ″InBackout.″. Complete these steps to view the outcome of other known transaction branches:
   a. Specify the XID FormatId (in decimal) in the ″Format ID″ field. The XID FormatId must be converted from hex to decimal.
   b. Specify the XID Gtrid (in hex) in the ″GTRID Pattern″ fields on the query panel. The XID Gtrid must be entered in 16 bytes to a line.
   c. Press **enter** to execute the query.
3. Match up the ″InDoubt″ branches. The query results obtained in step 2, above, will show all URIDs involved with the specified transaction. In the query results, take note of the outcomes in the *state* column. If any other branches of the transaction are ″InCommit″ or ″InBackout,″ match the corresponding ″InDoubt″ branches to read the same:

   In the left-hand column labeled ″S″, enter ″c″ to commit or ″b″ to backout the UR that was identified in message **BBOT0019W**. Manually resolving the transaction can lead to mixed transaction outcomes across resource managers and servers.
4. Copy to the clipboard the URID on this panel.
5. Remove the interest for this URID

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.″

Resolving InDoubt units if you receive message BBOT0021W:

This message indicates that a server has determined the transaction outcome but has not been able to communicate it to its subordinates. When this 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.

Perform the following steps to remove an expression of interest in this UR.

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 **enter** 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 will be a heading titled ″Expressions of Interest″, followed by one or more rows below it. These rows 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 **enter** to execute the query.

The ″RRS Remove Interest Confirmation″ panel will open. The RM name and UR identifier fields are pre-filled. Press ″enter″ to confirm the removal of this interest.

From this point onward, any subordinate nodes that restart and ask this server about this UR will be unable to obtain this information. If you restart the server containing these nodes, they may be assigned an outcome 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.

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.″

Resolving InDoubt units if you receive message BBOT0022W:

This message indicates that the transaction outcome has been determined and communicated to the subordinate, but the subordinate resource has not yet been ″forgotten.″ When this message is displayed, there has already been a heuristic outcome.

The subordinate will not let the UR move to the ″in-forget″ state until it is told to forget it. In other words, the subordinate will still be involved in a UR, so RRS will not mark the subordinate server as ″restart anywhere.″

See *z/OS MVS Programming: Resource Recovery* for more information on how to use the RRS panels and the administrative access (RACF access to the facility class, for example) that is needed to resolve URs and remove interests.

Use RRS panels to perform the following steps to remove an expression of interest in this UR.

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 the URID in the ″URID Pattern″ field on the query panel. Press **enter** to execute the query. The query results should display the UR. 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 will be a heading titled ″Expressions of Interest″, followed by one or more rows below it. These rows 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 **enter** to execute the query.
4. Press **enter** 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 **enter** to confirm the removal of this interest.

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.″

Resource Recovery Services Operations

Purpose

This topic provides tips for using the z/OS Resource Recovery Services with WebSphere Application Server for z/OS

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.

  Note: 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.

  Note: 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

Purpose

When a JTA XAResource manager is enlisted in a global transaction, it cannot express an interest in the RRS unit of recovery (UR) like an RRS resource manager

can. Instead, the WebSphere Application Server for z/OS transaction service will save information in its RRS interest indicating that a JTA Resource Manager was enlisted in the transaction. 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 WebSphere Application Server for z/OS XA partner log 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 logstream 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.

Setting up WebSphere Application Server for z/OS on multiple systems in a sysplex

A typical WebSphere Application Server for z/OS base runtime includes a cell with a location service daemon (BBODMNB) and one node which includes an Application Server (server1) with a controller and any number of servants. After you have installed the Application Server runtime and associated business application servers on a monoplex, you can migrate the runtime and associated application servers to a sysplex configuration.

The benefits of migrating to a sysplex include:
- You can balance the workload across multiple systems, thus providing better performance management for your applications.
- As your workload grows, you can add new systems to meet demand, thus providing a scalable solution to your processing needs.
- By replicating the runtime and associated business application servers, you provide the necessary system redundancy to assure availability for your users. Thus, in the event of a failure on one system, you have other systems available for work.
- You can upgrade the Application Server from one release or service level to another without interrupting service to your users.

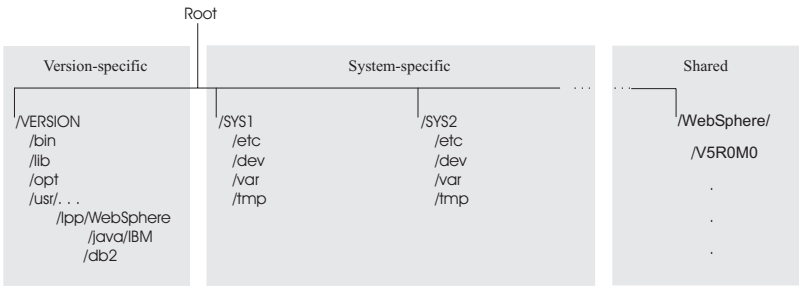
The following table shows the subtasks and associated procedures for enabling WebSphere Application Server for z/OS in a sysplex.

Table 5.

Subtask	**Associated procedure (See . . .)**
Setting up a sysplex	*z/OS MVS Setting Up a Sysplex*
Making decisions about the WebSphere Application Server for z/OS configuration and sysplexes	"Steps for planning WebSphere Application Server for z/OS and cells" on page 254
Preparing your security system	*Securing your applications and their environment* PDF
Setting up data sharing	*DB2 Data Sharing: Planning and Administration*
Customizing base z/OS functions on the other systems in the sysplex	"Steps for customizing base z/OS functions on the other systems in the sysplex" on page 255
Making changes to TCP/IP	"Steps for making changes to TCP/IP" on page 256
Defining new Application Server clusters in the sysplex	"Creating cluster members" on page 340

Overview of a WebSphere Application Server for z/OS sysplex

A typical WebSphere Application Server for z/OS base runtime includes a cell with a location service daemon (BBODMNB) and one node which includes an application server (server1) with a controller and any number of servants. Once you have installed the Application Server runtime and associated business application servers on a monoplex, you can migrate the runtime and associated application servers to a sysplex configuration. The benefits of migrating to a sysplex include:
- You can balance the workload across multiple systems, thus providing better performance management for your applications.
- As your workload grows, you can add new systems to meet demand, thus providing a scalable solution to your processing needs.
- By replicating the runtime and associated business application servers, you provide the necessary system redundancy to assure availability for your users. Therefore, in the event of a failure on one system, you have other systems available for work.
- You can upgrade the Application Server from one release or service level to another without interrupting service to your users.



Steps for planning WebSphere Application Server for z/OS and cells

You should have completed the WebSphere Application Server for z/OS installation and customization on a monoplex or on a single system in a sysplex. Also, you must have enabled a z/OS sysplex. For more information on sysplexes, see *z/OS MVS Setting Up a Sysplex*.

Follow these steps to plan for using the Application Server and cells:

1. 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.
2. Decide how you will share application executables in the cell.
3. 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).
4. Decide whether you will run all the WebSphere Application Server for z/OS run-time servers on every system in the cell.

   Recommendations: The following table provides recommendations and requirements for running servers in a cell.

Table 6. 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 these servers on each system in the cell in which you want Application Server work to run. Thus, you may have some systems in your cell that do not run the Application Server or Application Server applications at all. But, for those systems on which you want Application Server applications to run, you must have the location service daemon and node agent. • 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.

5. **Optional:** Follow these steps to build WebSphere Application Server for z/OS Deployment Manager cells:
   a. Install the base server on each node in your sysplex.
   b. Install a deployment manager cell on one node in your sysplex.

c. Add base server nodes to the deployment manager cell.

Steps for customizing base z/OS functions on the other systems in the sysplex

You must have the WebSphere Application Server for z/OS product code installed through SMP/E, and you must have created copies of the product sample files.

Repeat the same customization to base z/OS functions that you did for your initial installation and customization of the Application Server. The steps are repeated here for convenience.

Note: The following steps assume that your default Application Server data set high-level qualifier (hlq) is ″BBO″. If it is not, modify the examples to use your specified hlq.

Perform the following steps to change the base system:

1. Change SCHEDxx to include the statements from the BBOSCHED sample file you ran in the customization dialog.
2. APF-authorize the BBO.SBBOLOAD, BBO.SBBOLD2, and BBO.SBBOLPA data sets.

   Example: Your PROGxx PARMLIB member could include:

```
APF FORMAT(DYNAMIC)
/*********************************************************************/
/* BOSS LOCAL DATASETS                                               */
/*********************************************************************/
APF ADD
    DSNAME(BBO.SBBOLOAD)
    VOLUME(vvvvvv)
APF ADD
    DSNAME(BBO.SBBOLD2)


    VOLUME(vvvvvv)
APF ADD
    DSNAME(BBO.SBBOLPA)
    VOLUME(vvvvvv)
```

   where vvvvvv is your volume identifier.
3. Ensure that the Language Environment data sets, SCEERUN and SCEERUN2, and the DB2 data set, SDSNLOAD, are authorized.
4. Do **not** APF-authorize BBO.SBBOULIB or BBO. SBBOMIG, because they should run under the authority of the client user.
5. Place Application Server modules. Use the following table to place Application Server modules:

Table 7. Placing modules in LPA or link list

Modules	Notes
BBO.SBBOLPA	Load all members into the LPA.
BBO.SBBOLOAD	We recommend you dynamically load all members into the LPA. If your virtual storage is constrained, place the members in the link list.
BBO.SBBOMIG	You can put members into the link list or LPA.
BBO.SBBOLD2	Do **not** put members from SBBOLD2 in the LPA. Place these members in the link list.
BBO.SBBOULIB	Do **not** place these members in **either** the LPA or link list.

Table 7. Placing modules in LPA or link list (continued)

Modules	Notes

	Rule: These data sets are PDSEs and cannot be added to members in LPALSTxx or IEALPAxx.
	
	Recommendation: For automation, if you want to ensure the Application Server modules are loaded into dynamic LPA and available after an IPL, create a new PROGxx member with the SETPROG LPA commands and invoke the PROGxx member from PARMLIB COMMNDxx.
	
	Example:
	
	`SETPROG LPA,ADD,MASK=*,DSNAME=`
	`BBO.`
	`SBBOLOAD`
	`SETPROG LPA,ADD,MASK=*,DSNAME=`
	`BBO.`
	`SBBOLPA`
	• Change "BBO" if it is not the high-level qualifier for your Application Server data sets.
	• If using SETPROG on a running system, be sure to purge modules with the same name as those from BBO.SBBOLPA, BBO.SBBOLOAD, or BBO.SBBOMIG that are already in the LPA.

6. **Optional:** If you used a PROGxx file for APF authorizations or the LPA, be sure to issue:

   ```
   SET PROG=xx
   ```

7. **Optional:** Make sure all the BBO.* data sets are cataloged. While not required, this is highly recommended.
8. Update your SYS1.PARMLIB(BLSCUSER) member with the IPCS models supplied by member BBOIPCSP. For details in BLSCUSER, see *z/OS MVS IPCS User's Guide*.
9. **Optional:** Start SMF recording. If you want to start SMF recording to collect system and job-related information on the WebSphere Application Server for z/OS system:
   a. Edit the SMFPRMxx parmlib member.
      1) Insert an 'ACTIVE' statement to indicate SMF recording.
      2) Insert a SYS statement to indicate the types of SMF records you want the system to create.

         Example: Use SYS(TYPE(120:120)) to select type 120 records only. Keep the number of selected record types small, to minimize the performance impact.

   b. To start writing records to DASD, issue the following command:

      ```
      t smf=xx
      ```

      Where xx is the suffix of the SMF parmlib member (SMFPRMxx). For more information about the SMF parmlib member, see *z/OS MVS System Management Facilities (SMF)*.

      When you activate writing to DASD, the data is recorded in a data set (specified in SMFPRMxx).

Steps for making changes to TCP/IP

You must have TCP/IP installed and configured.

1. Change DNS entries. Assuming you use an implementation of the DNS that allows use of generic IP names that dynamically resolve to like-configured servers, you must adjust the IP names in your DNS. 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. This is important not only for workload balancing, but in the event of a server failure: the DNS can direct work to other servers.
2. In the TCP/IP profile for each additional system in the cell, add a port for the location service daemon and associate it with a new location service daemon server name.

   By default, WebSphere Application Server for z/OS uses port 5655 for the location service daemon. Also, WebSphere Application Server for z/OS names the first location service daemon server DAEMON01 and increments the suffix on that name for each new location service daemon server (DAEMON02, DAEMON03, and so forth). Therefore, on your second system in the cell, add a port and associate it with DAEMON02.

   Example:

```
5655
   TCP     DAEMON02
```

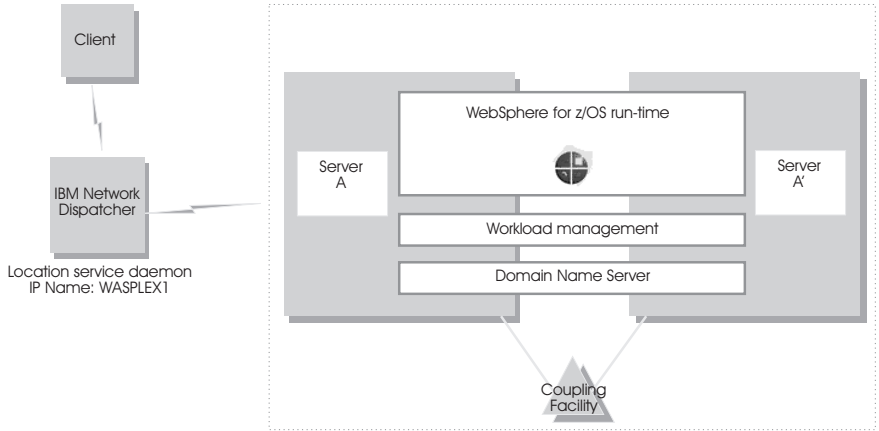
   Follow the same pattern for your third and subsequent systems in the cell.

Load Balancer

Load Balancer (known as Network Dispatcher in earlier releases) is a router that handles network requests for the cell.

Characteristics of such a configuration are:

- The location service daemon IP Name is associated with the IP address of the router.
- Load Balancer cooperates with workload management to route requests through the cell. The client never sees a change in IP addresses.
- The implication for clients is that they can cache the IP addresses, because this configuration does not change them dynamically.



Configuring transport chains

You need to configure transport chains or HTTP transports to provide networking services to such functions as 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 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. To define these channels:

1. Create a transport chain: You can either use the administrative console or wsadmin commands to create a transport chain. If you want to use the administrative console:
   a. Ensure that a port is available for the new transport chain.
   b. In the administrative console, click **Servers > Application servers** >*server_name*, and then click on one of the following:
      - Under **Web container settings**, click **Web container transport chains**.
      - Under **Server messaging**, click either **Messaging engine inbound transports** or **WebSphere MQ link inbound transports**.
   c. 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 will be 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.

      When you click **Finish**, the new transport chain is added to the list of defined transport chains on the **Transport chain** panel.
2. Click on the transport chain's name to view the configuration settings that are in effect for the transport channels contained in this chain. To change any of these settings:
   a. Click on the channel that requires changes to its settings.
   b. Make your changes to 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 have been defined for your system.
3. When you have made all of your changes, click **OK**.
4. Stop the application server and start it again. You must stop the application server and start it again before the configuration changes you made take affect.

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.

Important: On the z/OS platform, transport chains are not as streamlined as the native HTTP transports. Therefore, you should continue to use HTTP transports instead of transport chains unless you want to take advantage of IPv6 or Web Services Atomic Transaction (WS-AT) support , or if you have multiple ports configured for the Application Server.

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.

TCP channel
Used to provide client applications with persistent connections within a Local Area Network (LAN). When configuring a TCP channel, you can specify a list of IP addresses that are allowed to make inbound connections and a list of IP addresses that are not allowed to make inbound connections. You can also specify the thread pool that this channel uses, which allows you to segregate work by the port that the application server is listening on.

HTTP 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 server HTTP requests and to send HTTP specific information to servlets expecting this type of information.

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.

Web container channel
Used to create a bridge in the transport chain between an HTTP inbound channel and a servlet and JavaServer Pages (JSP) engine.

DCS channel
Used by the core group bridge service, the data replication service (DRS), and the high availability manger to transfer data, objects, or events among application servers.

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 WebSphere System Integration Bus and a WebSphere MQ client or queue manager.

JFAP channel
Used by the Java Message Service (JMS) server to create connections to JMS resources on a service integration bus.

SSL channel

Used to associate an SSL configuration repertoire with the transport chain. This channel is only available when Secure Sockets Layer (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.

HTTP transport channel custom property

If you are using an HTTP transport channel, you can add the following custom property to the configuration settings for that HTTP transport channel.

To add a custom property:

1. In the administrative console, click **Application servers >** *server_name* **Web container settings > Web container transport chains >***chain_name* **> 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 is a list of custom properties provided with the application server. These properties are not shown on the settings page for an HTTP transport channel.

inProcessLogFilenamePrefix

Use 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`.

Data type String

HTTP Tunnel transport channel custom property

If you are using an HTTP Tunnel transport channel, you can add the following custom property to the configuration settings for that HTTP Tunnel transport channel.

To add a custom property:

1. In the administrative console, click **Servers > 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.

Following is a description of the custom property that is provided with the application server. 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

Troubleshooting transport chain problems

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 a WebSphere 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 window to display the state of the port to which you are trying to bind.

Error message CHFW0030E indicates there is "No such file or directory,"

If you receive an Error message CHFW0030E that indicates there is "No such file or directory," and you are running on an HP-UX operating system, make sure you have the most current patches for that operating system installed.

Configuring HTTP transports

Important: On the z/OS platform, the administrative console page used to configure an HTTP transport will always be available. The HTTP transport that is created by the WebSphere Application Server for z/OS ISPF Customization Dialog is a required element of the WebSphere Application Server for z/OS runtime. Optionally, you can configure an HTTPS transport using the HTTP Transport panel of the administrative console, or new Version 6 HTTP transport channels using the Channel/Chain Configuration panel.

Note that 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.

An HTTP transport is the request queue between a WebSphere Application Server plug-in for Web servers and a Web container in which the Web modules of an application reside. To define the characteristics of the connections between that plug-in and the Web container, you must specify:
- How the transport is to handle a set of connections.
- Whether to secure the connections with SSL.
- The Host and IP information for the transport participants.

Note: Only one HTTP transport and one HTTPS transport can be defined per Application Server.

1. Change the configuration for an existing HTTP transport.
   a. Ensure that virtual host aliases include port values for the transport your are changing.
   b. Go to the HTTP Transports page in the administrative console and click on the transport under **Host** whose configuration you want to change.
   c. On the settings page for an HTTP transport, which might have the page title DefaultSSLSettings, change the specified values as needed, then click **OK**.
   d. Custom properties page, add and set any custom properties you want to use.
2. Stop the WebSphere Application Server and start it again. You must stop the WebSphere Application Server and start it again before the configuration changes you made take affect.

If the Web server is located on a machine remote from the Application Server but is defined on a managed node, the updated plugin-cfg.xml is automatically propagated to the Web server.

HTTP transport collection

Use this page to view or manage HTTP transports. Transports provide request queues between WebSphere Application Server plug-ins for Web servers 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.

On the z/OS platform, the HTTP transport will continue to be used as the default transport for HTTP requests. The WebSphere Application Server for z/OS ISPF Customization Dialog will continue to be used to configure a default HTTP transport and, at most, one additional transport for HTTPS. If you are a WebSphere Application Server for z/OS Version 5.x user who has migrated to Version 6, note that the Version 5.x HTTP (and, optionally HTTPS) transport definitions will remain unchanged in Version 6. You can use the HTTP Transport panel of the administrative console to configure these transports. In Version 6, you can define additional HTTP listeners as HTTP transport channels.

Note that 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 > Application Servers >***server_name* **> Web Container Settings > Web Container > HTTP Transports**.

the HTTP Transport panel on the administrative console

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 the z/OS platform, 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 connections between the WebSphere plug-in and application server 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.

On the z/OS platform, the HTTP transport will continue to be used as the default transport for HTTP requests. The WebSphere Application Server for z/OS ISPF Customization Dialog will continue to be used to configure a default HTTP transport and, at most, one additional transport for HTTPS. If you are a WebSphere Application Server for z/OS Version 5.x user who has migrated to Version 6, note that the Version 5.x HTTP (and, optionally HTTPS) transport definitions will remain unchanged in Version 6. You can use the HTTP Transport panel of the administrative console to configure these transports. In Version 6, you can define additional HTTP listeners as HTTP transport channels.

Note that 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 panel on the administrative console, click **Servers > Application Servers >***server_name* **> Web Container Settings > Web Container > HTTP Transports >***host_name*.

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 connections between the WebSphere Application Server plug-in and application server 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 connections between the WebSphere Application Server plug-in and application server. The options include one or more SSL settings defined in the Security Center; for example, DefaultSSLSettings, ORBSSLSettings, or LDAPSSLSettings.

Data type	String
Default	An SSL setting defined in the Security Center

Transports

A transport is the request queue between a WebSphere Application Server plug-in for Web servers and a Web container in which the Web modules of an application reside. When a user at a Web browser requests an application, the request is passed to the Web server, then along the transport to the Web container.

Transports define the characteristics of the connections between a Web server and an application server, across which requests for applications are routed. Specifically, they define the connection between the Web server plug-in and the Web container of the application server.

Administering transports is closely related to administering WebSphere Application Server plug-ins for Web servers. Indeed, without a plug-in configuration, a transport configuration is of little use.

On the z/OS platform, the HTTP transport will continue to be used as the default transport for HTTP requests. The WebSphere Application Server for z/OS ISPF Customization Dialog will configure a default HTTP transport and a maximum of one additional transport for HTTPS. If you are a WebSphere Application Server for z/OS Version 5.x user who has migrated to Version 6, note that the Version 5.x HTTP (and, optionally HTTPS) transport definitions will remain unchanged in Version 6. You can use the HTTP Transport panel of the administrative console to configure these transports. In Version 6, you can define additional HTTP listeners as HTTP transport channels.

Note that 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.

The internal transport

The internal HTTP transport allows HTTP requests to be routed to the application server indirectly through a Web server plug-in. Logging is provided for debug purposes.

The HTTP transport functionality provides the means of accepting HTTP requests forwarded by an HTTP plug-in that is connected to a Web server. By default, the internal HTTP transport listens for HTTP requests on port 9080 and for HTTPS requests on port 9443.

The transport configuration is a part of the Web container configuration. You can configure the internal transport to use ports other than 9080 and 9443. However, you must also adjust your virtual host alias and what you type into the Web browser.

Remember: You can only configure one HTTP port and one HTTPS port.

HTTP transport custom properties

Use this page to set custom properties for an HTTP transport.

On the z/OS platform, the initial HTTP transport is created by the WebSphere Application Server for z/OS ISPF Customization Dialog. You can use the HTTP Transport panel on the administrative console to update the new HTTP transport's configuration settings or to create an additional, and optional, HTTPS transport. If you are a WebSphere Application Server Version 5.x user and have migrated to Version 6, you can use the HTTP Transport panel on the administrative console to update the configuration settings for the required HTTP transport and the optional HTTPS transport. You can define additional HTTP listeners as HTTP transport channels.

Note that 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** panel on 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:

1. In the administrative console click **Servers > Application Servers >***server_name* **> Web Container settings > Web Container >HTTP Transport**
2. Select a host.
3. Under **Additional Properties** select **Custom Properties**.
4. On the Custom Properties page, click **New**.
5. 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.
6. Click **Apply** or **OK**.
7. Click **Save** on the console task bar to save your configuration changes.
8. 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 an HTTP transport.

ConnectionIOTimeOut:

Use the `ConnectionIOTimeOut` property to specify the maximum number of seconds to wait when trying to read or process data during a request.

Data type	Integer
Default	For the z/OS platform: 120 seconds

ConnectionKeepAliveTimeout:

Use the `ConnectionKeepAliveTimeout` property to specify the maximum number of seconds to wait for the next request on a keep alive connection.

For WebSphere Application Server for z/OS, this value is input to the TCP/IP SOCK_TCP_KEEPALIVE option. The Application Server does not set the TCP/IP SOCK_TCP_KEEPALIVE option by default, but will do so if a value is specified for this property.

Data type	Integer
Default	30 seconds

ConnectionResponseTimeout:

This property is only valid in a z/OS environment. Use the `ConnectionResponseTimeout` property to specify the maximum number of seconds to wait when trying to read data during a response. For WebSphere Application Server and WebSphere Application Server for Network Deployment, this also applies to writing data during a response.

Data type	Integer
Default	300 seconds

MaxConnectBacklog:

This property is only valid in a WebSphere Application Server for Distributed Platforms environment. Use the `MaxConnectBacklog` property to specify the maximum number of outstanding connect requests that the operating system will 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 will be rejected.

Set this value to the number of concurrent connections that you would like to allow. Keep in mind that a single client browser might need to open multiple concurrent connections (perhaps 4 or 5); however, also keep in mind that increasing this value consumes more kernel resources. The value of this property is specific to each transport.

Data type	Integer
Default	511

MaxKeepAliveConnections:

This property is only valid in a WebSphere Application Server for Distributed Platforms environment. It is ignored in a z/OS environment because asynchronous I/O sockets are used to maintain connections in that environment.

Use the `MaxKeepAliveConnections` property to specify the maximum number of concurrent keep alive (persistent) connections across all HTTP transports. To make a particular transport close connections after a request, you can set MaxKeepAliveConnections to `0` (zero) or you can set KeepAliveEnabled to `false` on that transport.

The Web server plug-in keeps connections open to the application server as long as it can. However, if the value of this property is too small, performance is negatively impacted because the plug-in has to open a new connection for each request instead of sending multiple requests through one connection. The application server might not accept a new connection under a heavy load if there are too many sockets in TIME_WAIT state. If all client requests are going through the Web server plug-in and there are many TIME_WAIT state sockets for port 9080, the application server is closing connections prematurely, which decreases performance. The application server closes the connection from the plug-in, or from any client, for any of the following reasons:

- The client request was an HTTP 1.0 request when the Web server plug-in always sends HTTP 1.1 requests.
- The maximum number of concurrent keep-alives was reached. A keep-alive must be obtained only once for the life of a connection, that is, after the first request is completed, but before the second request can be read.
- The maximum number of requests for a connection was reached, preventing denial of service attacks in which a client tries to hold on to a keep-alive connection forever.
- A time out occurred while waiting to read the next request or to read the remainder of the current request.

Data type	Integer
Default	90% of the maximum number of threads in the Web container thread pool. This prevents all of the threads from being held by keep alive connections so that there are threads available to handle new incoming connect requests.

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.

If this property is set to 0 (zero), the connection stays open as long as the application server is running.

Data type	Integer
Default	50 requests

MutualAuthCBindCheck:

This property is only valid in a z/OS environment. 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.servername CLASS(CBIND) ID(clientCertUserid) ACCESS(CONTROL)
```

Data type	String
Default	false

TrustedProxy:

This property is only valid in a z/OS environment. Use the `TrustedProxy` property to indicate that the application server can use the private headers that the Web server plug-in adds to requests.

Data type	String
Default	false

ServerHeader:

This property is only valid in a z/OS environment. 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
Default	true

Transport chains collection

Use this page to view or manage transport chains. Transport chains enable communication through transports, 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.

Important: On the z/OS platform, transport chains are not as streamlined as the native HTTP transports. Therefore, you should continue to use HTTP transports instead of transport chains unless you want to take advantage of IPv6 or Web Services Atomic Transaction (WS-AT) support , or if you have multiple ports configured for the Application Server.

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 > 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. For WebSphere Application Server, transport name must be unique within a WebSphere Application Server configuration. Click on the name of a transport chain to change its configuration settings.

Enabled: When set to true, the transport chain is activated at application server startup.

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, might be localhost or the wildcard character * (an asterisk). The port number must be unique for each application server instance on a given machine

SSL Enabled: When set to true, users are notified if there is a channel that enables Secure Sockets Layer (SSL) in the listed chain. When SSL is enabled, all traffic going through this transport is encrypted and digitally secured.

Transport chain settings

This page lists 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 > Application servers >** *server_name* **> Ports**. Click on **View associated transports** for the port whose transport chains you want view and then click on 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 a WebSphere Application Server configuration.

Enabled: When checked, this transport chain is activated at application server startup.

Transport channels: Lists the transport channels configured for this transport chain and their configuration settings. To change a transport channel's configuration settings, click on the name of that transport 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 > 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.

This name must be unique across all channels in a WebSphere Application Server environment. For example DCS and TCP transport channels cannot have the same name if they reside within the same 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

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.

On the z/OS platform, the HTTP transport will continue to be used as the default transport for HTTP requests. The WebSphere Application Server for z/OS ISPF Customization Dialog will continue to be used to configure a default HTTP transport and, at most, one additional transport for HTTPS. If you are a WebSphere Application Server for z/OS Version 5.x user who has migrated to Version 6, note that the Version 5.x HTTP (and, optionally HTTPS) transport definitions will remain unchanged in Version 6. You can use the HTTP Transport panel of the administrative console to configure these transports. In Version 6, you can define additional HTTP listeners as HTTP transport channels.

To view this administrative console page, click **Servers > Application servers >***server_name* **> Ports** . Click on **View associated transports** for the port associated with the HTTP transport channel whose settings you want to look at.

Transport channel name:

Specifies the name of the HTTP transport channel.

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.

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

Maximum persistent requests:

Specifies the maximum number of persistent (keep-alive) requests that are allowed on a single HTTP connection. If a value of 0 (zero) is specified, only one request is allowed per connection. If a value of -1 is specified, an unlimited number of requests is allowed per connection.

Data type	Integer
Default	100

Use Keep-Alive: When selected, the HTTP transport channel, when sending an outgoing HTTP message, uses a persistent connection (keep-alive connection) instead of a connection that closes after one request or response exchange occurs.

Note: If a value other than 0 is specified for the `maximum persistent requests` property, the `Use Keep-Alive property` setting is ignored.

The default for this property is selected.

Read timeout:

Specifies the amount of time, in seconds, the HTTP transport channel waits for a read request to complete on a socket after the first read request occurs. The read being waited for could be an HTTP body (such as a POST) or part of the headers if they were not all read as part of the first read request on the socket.

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 response data to be transmitted. This timeout usually only occurs in situations where the writes are lagging behind new requests. This can occur when a client has a low data rate or the server's network interface card (NIC) is saturated with I/O.

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.

Data type	Integer
Default	30 seconds

Enable NCSA access logging:

When selected, the HTTP transport channel performs NCSA access and error logging. Enabling NCSA access and error logging slows server performance.

To configure NCSA access and error logging, click **HTTP error and NCSA access logging** under **Related Items**. Even if HTTP error and NCSA access logging is configured, it is not enabled unless the Enable NCSA access logging property is selected.

The default value for the Enable NCSA access logging property is not selected.

TCP transport channel settings

Use this page to view and configure an 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 > 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.

This name must be unique across all channels in a WebSphere Application Server environment. For example, a TCP transport channel and an HTTP transport channel cannot have the same name if they reside within the same system.

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.

Thread pool: Select from the drop-down list of available thread pools the thread pool you want the TCP transport channel to use when dispatching work.

Maximum open connections:

Specifies the maximum number of connections that can be open at one time.

Data type	Integer between 1 and 20,000 inclusive
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.

Note: 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
```

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 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
```

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 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 can not 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
```

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.

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 can not 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 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 chains. 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 > Application servers >***server_name* **> Ports >** . Click on **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.

This name must be unique across all channels in a WebSphere Application Server environment. For example DCS and TCP transport channels cannot have the same name if they reside within the same 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.

The discrimination weight of the DCS channel in a DCS-Secure transport chain should always be less than the discrimination weight of the SSL channel that is in that chain. Other SSL channels in other chains might have different discrimination values.

Data type Positive integer

Default	1 for the DCS channel	2 for the SSL channel

SSL inbound channel

Use this page to determine which SSL inbound channel options to specify for WebSphere Application Server.

To view this administrative console page:

1. Click **Servers** > **Application Servers** > *server_name*.
2. Under Container settings, click **Web container transport chains** > *secure_transport_chain*.
3. Under Transport channels, click **SSL Inbound Channel (SSL_1)**.

Transport Channel Name:

Unique name for a given layer in a network protocol stack.

Specifies transport channel name.

Discrimination weight:

Discrimination weight is used to determine the discrimination order in situations where transport channels are shared amongst several chains.

Specifies discrimination weight.

SSL repertoire:

SSL repertoire is used to define any number of SSL settings which can be used to make HTTPS, IIOPS or LDAPS connections.

Specifies SSL repertoire

Web container 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** > **Application servers** > *server_instance* > **Container Settings** > **Web Container Settings**> **Web container** > **Web container transport chains** > *transport_chain* > **Web Container Inbound Channel** .

Transport Channel Name:

This name must be unique across all channels in a WebSphere Application Server environment. This means that TCP transport channels and HTTP transport channels cannot have the same name if they reside within the same system.

Discrimination weight:

Specifies the priority that this transport chain has in relation to other transport chains if this transport channel is shared amongst several transport chains.

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	9192 bytes

Custom services

A custom service provides the ability to plug into a WebSphere Application Server application server to define a hook point that runs when the server starts and shuts down.

A developer implements a custom service containing a class that implements a particular interface. The administrator configures the custom service in the administrative console, identifying the class created by the developer. When an application server starts, any custom services defined for the application server are loaded and the server runtime calls their initialize methods.

For z/OS, 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.

Developing custom services

To define a hook point to be run when a server starts and shuts down, you develop 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.

The following restrictions apply to the WebSphere Application Server custom services implementation:

- The init 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 init and shutdown methods are called only once on each service, and once for each operating system process that makes up the server instance. File I/O is supported.
- 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 J2EE code (client code, servlets, enterprise beans) is not supported.
- The JTA interface is not available. This feature is available in J2EE 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.

Note that these restrictions apply to the shutdown and init methods equally. Some JNDI operations are available.

1. Develop a custom service class that implements the com.ibm.websphere.runtime.CustomService interface. The properties passed by the application server runtime to the initialize method can include one for an external file containing configuration information for the service (retrieved with externalConfigURLKey). In addition, the properties can contain any name-value pairs that are stored for the service, along with the other system administration configuration data for the service. The properties are passed to the initialize method of the service as a Properties object.

   There is a shutdown method for the interface as well. Both methods of the interface declare that they may create an exception, although no specific exception subclass is defined. If an exception is created, the runtime logs it, disables the custom service, and proceeds with starting the server.
2. On the Custom Service page of the administrative console, click **New**. Then, on the settings page for a custom service instance, create a custom service configuration for an existing application server, 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. Doing this adds the path name to the WebSphere Application Server extension classloader, which allows the custom service to locate and correctly load the native library.
3. Stop the application server and then restart the server.
4. Check the application server to ensure that the initialize method of the custom service ran as intended. Also ensure that the shutdown method performs as intended when the server or node agent stops.

As mentioned above, your custom services class must implement the CustomService interface. In addition, your class must implement the initialize and shutdown methods. Suppose the name of the class that implements your custom service is *ServerInit*, your code would declare this class as shown below. The code below assumes that your custom services class needs a configuration file. It shows how to process the input parameter in order to get the configuration file. If your class does not require a configuration file, the code that processes configProperties is not needed.

```
public class ServerInit implements CustomService
{
/**
* The initialize method is called by the application server run-time when the
* server starts. The Properties object passed to this method must contain all
* configuration information necessary for this service 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 run-time when the
* server begins its shutdown processing.
*
* @param configProperties java.util.Properties
*/
    public void shutdown() throws Exception
    {
        // Implement shutdown method
    }
```

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 a WebSphere application server and define code that runs when the server starts or shuts down.

To view this administrative console page, click **Servers > Application servers >***server_name*. Then, under Server Infrastructure, 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 > Application servers >***server_name*. Then, under Server Infrastructure, 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

Process definition

A process definition specifies the run-time characteristics of an application server process.

A process definition can include characteristics such as JVM settings, standard in, error and output paths, and the user ID and password under which a server runs.

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 run-time properties such as the program to run, arguments to run the program, and the working directory.

1. In the administrative console, click **Servers > Application Servers** in the console navigation tree, click on an application server name and then **Java and Process Management > Process Definition**. Note that you can also define application server processes using the wsadmin tool. For more information, see the *Administrating applications and their environment* PDF.
2. 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**.
3. Specify process execution statements for starting or initializing a UNIX process.
4. Specify monitoring policies to track the performance of a process.
5. 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.
6. Specify name-value pairs for properties needed by the process definition.
7. Stop the application server and then restart the server.
8. Check the application server to ensure that the process definition runs and operates as intended.

Process definition settings

Use this page to view or change settings for a process definition. For WebSphere Application Server, this page provides command-line information for starting or initializing a process.

To view this administrative console page, click **Servers > Application Servers >***server_name*. Then under Server Infrastructure click **Java Process Management > Process Definition**.

For the z/OS platform, this page provides command-line information for starting, initializing, or stopping a process. Each of the commands that follows can be used for the control process. The servant process uses only the Start command and Start Command Args. 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 put the commands for the two different processes on the same panel.

Start Command (startCommand):

This command applies to the z/OS platform only. It specifies the platform-specific command to launch the server process.
Control process

Data type	String
Format	START *control JCL procedure name*
Example	START BBO5ACR

Servant process

For the 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.

Data type	String
Format	*servant JCL procedure name*
Example	BBO5ASR

Start Command Args (startCommandArgs):

This command applies to the z/OS platform only. It specifies any additional arguments required by the start command.
Control process

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

Servant process

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

Note: For the z/OS platform, the server short name (JOBNAME) contains 7 characters by default, but you can lengthen the short name to 8 characters. For more information, see Converting a 7-character jobname to 8 characters.

Stop Command (stopCommand):

This command applies to the z/OS platform only. It specifies the platform-specific command to stop the server process

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*
Example	STOP BBOS001;CANCEL BBOS001

Stop Command Args (stopCommandArgs):

This command applies to the z/OS platform only. It specifies any additional arguments required by the stop command.

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*

Example	;ARMRESTART **Note:** In this example, Stop has no arguments. Immediate Stop has the argument ARMRESTART. A semicolon precedes ARMRESTART.

Terminate Command (terminateCommand):

This command only applies to the z/OS platform. It specifies the platform-specific command to terminate the server process.

Data type	String
Format	FORCE *server short name*
Example	FORCE BBOS001

Terminate Command Args (terminateCommandArgs):

This command only applies to the z/OS platform. It specifies any additional arguments required by the terminate command.

The default is an empty string.

Data type	String
Format	*terminate command arg string*
Example	ARMRESTART

Working Directory:

Specifies the file system directory that the process uses as its current working directory.

The process uses this directory to determine the locations of input and output files with relative path names.

Passivated enterprise beans are placed in the current working directory of the application server on which the beans are running. Make sure the working directory is a known directory under the root directory of the WebSphere Application Server product.

Data type	String

Process execution settings

Use this page to view or change the process execution settings for a server process that applies to either an application server, a node agent or a deployment manager.

To view this administrative console page for an application server, click **Servers > Application Servers >***server_name*. Then, under Server Infrastructure , click **Java and Process Management > Process Execution**.

To view this administrative console page for a node agent, click **System Administration > Node agents >***node_agent_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, under Server Infrastructure , click **Java and Process Management > Process Definition > 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 for WebSphere Application Server on all operating systems.

UMASK:

Specifies the user mask under which the process runs (the file-mode permission mask).

Data type	Integer

Run As User:

Specifies the user that the process runs as.

Data type	String

Run As Group:

Specifies the group that the process is a member of and runs as.

On OS/400, the Run As Group setting is ignored.

Data type	String

Run In Process Group:

Specifies a specific process group for the process. This process group is useful for such things as processor partitioning. A system administrator can assign a process group to run on, for example, 6 of 12 processors. The default (0) is not to assign the process to any specific group.

On OS/400, the Run In Process Group setting is ignored.

Data type	Integer
Default	0

Process logs settings

Use this page to view or change settings for specifying the files to which standard out and standard error streams write.

To view this administrative console page, click **Servers > Application Servers >***server_name*. Then, under Troubleshooting, click **Logging and Tracing > Process Logs**.

Stdout File Name:

Specifies the file to which the standard output stream is directed. The file name can include a symbolic path name defined in the variable entries.

Use the field on the configuration tab to specify the file name. Use the field on the Runtime tab to select a file for viewing. View the file by clicking **View**.

Direct server output to the administrative console or to the process that launched the server, by either deleting the file name or specifying `console` on the configuration tab.

Data type	String
Units	File path name

Stderr File Name:

Specifies the file to which the standard error stream is directed. The file name can include a symbolic path name defined in the variable entries.

Use the field on the configuration tab to specify the file name. Use the field on the runtime tab to select a file for viewing. View the file by clicking **View**.

Data type	String
Units	File path name

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 > Application Servers >***server_name*. Then, under Server Infrastructure, click **Java and Process Management > Process Definition >***process* **> Monitoring Policy**.

Maximum Startup Attempts:

Specifies the maximum number of times to attempt to start the application server before giving up.

Data type	Integer

Ping Interval:

Specifies 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.

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 2147483. If you specify a value greater than 2147483, the application server acts as though you set the value to 0.

Automatic Restart:

Specifies whether the process should restart automatically if it fails. The default is to restart the process automatically.

Data type	Boolean
Default	true

Node Restart State:

Specifies the desired state for the process after the node completely shuts down and restarts.

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

To view this administrative console page, click **Servers > Application Servers >***server_name* **> Process Definition**.

Control:

Specifies the process definitions for the control process

Servant:

Specifies the process definitions for the servant process.

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 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.

Multiple TCP/IP stacks

When configuring WebSphere Application Server for z/OS on a system with multiple stacks, you must first establish the Application Server's stack affinity to the desired stack so that all socket communications are bound to that stack, and then you establish the Application Server's allocation of the proper host name resolution configuration data sets so that host name lookups have the desired results.

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 OSA Features, each one connecting your system to a different network. You can assign a TCP/IP stack to each one. To do so, use the common INET physical file system (C_INET PFS). This physical file system allows you to configure multiple physical file systems (network sockets) and make them active concurrently.

1. Specify common INET through the NETWORK DOMAINNAME parameter of SYS1.PARMLIB(BPXPRMxx).
2. If you plan to configure WebSphere Application Server for z/OS 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.

To configure WebSphere Application Server for z/OS on a system with multiple stacks:

Note: 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.

1. Configure the data set for each Application Server's host name resolution. In the administrative console, click **Environment** > **Manage 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** > **Manage 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.

   See *z/OS UNIX System Services Planning*, for more information on the **_BPXK_SETIBMOPT_TRANSPORT** variable.

Java virtual machines (JVMs)

The Java virtual machine (JVM) is an interpretive computing engine 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 in order to run, and to support the Java applications running on it. JVM settings are part of an application server configuration.

Using the JVM

As part of configuring an application server, you might define settings that enhance your system's use of the Java virtual machine (JVM).

To view and change the JVM configuration for an application server's process, use the Java Virtual Machine page of the administrative console or use the wsadmin tool to change the configuration through scripting.

1. Access the Java Virtual Machine page.
   a. Click **Servers > Application Servers** in the console navigation tree.
   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, click **Java and Process Management > Process Definition**.
   d. On the Process Definition page, click **Java Virtual Machine**.

2. On the Java Virtual Machine page, specify values for the JVM settings as needed and click **OK**.
3. Click **Save** on the console task bar.
4. Restart the application server.

″“Configuring application servers for UTF-8 encoding” on page 201″ 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 293″ 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 for the application server's process.

To view this administrative console page, click **Servers > Application Servers >***server_name* **> Process Definition >** *process* **> Java Virtual Machine**.

Classpath:

Specifies the standard class path in which the Java virtual machine code looks for classes.

Enter each classpath entry into a table row. You do not need to add the colon or semicolon at the end of each entry.

Data type	String
Units	Class path

Boot Classpath:

Specifies bootstrap classes and resources for JVM code. This option is only available for JVM instructions that support bootstrap classes and resources. You can separate multiple paths by a colon (:) or semi-colon (;), depending on operating system of the node.

Data type	String

Verbose Class Loading:

Specifies whether to use verbose debug output for class loading. The default is not to 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

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 the initial heap size available to the JVM code, in megabytes.

Increasing the minimum heap size can improve startup. The number of garbage collection occurrences are reduced and a 10% gain in performance is realized.

Increasing the size of the Java heap improves throughput until the heap no longer resides in physical memory, in general. After the heap begins swapping to disk, Java performance suffers drastically.

Data type	Integer
Default	The default for the Controller is 48, and for the Servant it is 128.

Maximum Heap Size:

Specifies the maximum heap size available to the JVM code, in megabytes.

Increasing the heap size can improve startup. By increasing heap size, you can reduce the number of garbage collection occurrences with a 10% gain in performance.

Increasing the size of the Java heap improves throughput until the heap no longer resides in physical memory, in general. After the heap begins swapping to disk, Java performance suffers drastically. Set the maximum heap size low enough to contain the heap within physical memory.

Data type	Integer
Default	0 for iSeries, 256 for all other platforms. Keep the value low enough to avoid paging or swapping-out-memory-to-disk.

Debug Mode:

Specifies whether to run the JVM in debug mode. The default is not to 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 Debug Mode is enabled.

Debug arguments are only required if the Debug Mode property is set to true. If you enable debugging on multiple application servers on the same node, make sure that the servers are using different `address` arguments, which define the port for debugging. For example, if you enable debugging on two servers and leave the default debug port for each server as `address=7777`, the servers could fail to start properly.

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.

The following are optional command line arguments that you can use by entering them into the **General JVM Arguments** field. If you enter more than one argument, separate each argument by a space.

Note: If the argument says it is for the IBM Developer Kit only, you cannot use the argument with another JVM, such as the Sun JDK or the HP JDK.

- **-Xquickstart:** You can use **-Xquickstart** for initial compilation 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. 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%.

  The **-Xquickstart** option is not supported on OS/400.
- **-Xverify:none:** When using this value, the class verification stage is skipped during class loading . By using **-Xverify:none** with the just in time (JIT) compiler enabled, startup time is improved by 10-15%.

  The **-Xverify:none** option is not supported on OS/400.
- **-Xnoclassgc:** You can use this value to disable class garbage collection, which leads to more class reuse and slightly improved performance. The trade-off is that you won't be collecting the resources owned by these classes. You can monitor garbage collection using the verbose:gc configuration setting, which will output class garbage collection statistics. Examining these statistics will help you understand the trade-off between the reclaimed resources and the amount of garbage collection required to reclaim the resources. However, if the same set of classes are garbage collected repeatedly in your workload, you should disable garbage collection. Class garbage collection is enabled by default.
- **-Xgcthreads:** You can use several garbage collection threads at one time, also known as *parallel garbage collection*. When entering this value in the **Generic JVM Arguments** field, also enter the number of processors that your machine has, for example, **-Xgcthreads***n*, where *n* is the number of processors. On a node with *n*

processors, the default number of threads is *n*. You should use parallel garbage collection if your machine has more than one processor. This argument is valid only for the IBM Developer Kit.

The **-Xgcthreads** option is not supported on OS/400.

- **-Xnocompactgc:** This value disables heap compaction which is the most expensive garbage collection operation. Avoid compaction in the IBM Developer Kit. If you disable heap compaction, you eliminate all associated overhead.
- **-Xinitsh:** You can use this value to set the initial heap size where class objects are stored. The method definitions and static fields are also stored with the class objects. Although the system heap size has no upper bound, set the initial size so that you do not incur the cost of expanding the system heap size, which involves calls to the operating system memory manager. You can compute a good initial system heap size by knowing the number of classes loaded in the WebSphere Application Server product, which is about 8,000 classes, and their average size. Having knowledge of the applications helps you include them in the calculation. You can use this argument only with the IBM Developer Kit.
- **-Xgpolicy:** You can use this value to set the garbage collection policy. If the garbage collection policy (gcpolicy) is set to `optavgpause`, concurrent marking is used to track application threads starting from the stack before the heap becomes full. The garbage collector pauses become uniform and long pauses are not apparent. The trade-off is reduced throughput because threads might have to do extra work. The default, recommended value is `optthruput`. Enter the value as `-Xgcpolicy:[optthruput|optavgpause]`. You can use this argument only with the IBM Developer Kit.
- **-XX:** The Sun-based Java Development Kit (JDK) Version 1.4.2 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 short lived objects so that objects in the pool do not live through more then one garbage collection cycle. The size of new generation pool is determined by the `NewSize` and `MaxNewSize` parameters. Objects that survive the first garbage collection cycle are transferred to another pool. The size of the survivor pool is determined by parameter `SurvivorRatio`. If garbage collection becomes a bottleneck, you can try customizing the generation pool settings. To monitor garbage collection statistics, use the object statistics in Tivoli Performance Viewer or the verbose:gc configuration setting. Enter the following values: `-XX:NewSize (lower bound)` , `-XX:MaxNewSize (upper bound)`, and `-XX:SurvivorRatio=NewRatioSize`. The default values are:`NewSize=2m MaxNewSize=32m SurvivorRatio=2` However, if you have a JVM with more than 1 GB heap size, you should use the values: `-XX:newSize=640m -XX:MaxNewSize=640m -XX:SurvivorRatio=16`, or set 50 to 60% of total heap size to a new generation pool.

  The **-XX** option is not supported on OS/400.
- **-Xminf:** You can use this value to specify the minimum free heap size percentage. The heap grows if the free space is below the specified amount. In reset enabled mode, this option specifies the minimum percentage of free space for the middleware and transient heaps. This is a floating point number, 0 through 1. The default is .3 (30%).
- **-server | -client:** Java HotSpot Technology in the Sun-based Java Development Kit (JDK) Version 1.4.2 introduces an adaptive JVM containing algorithms for optimizing byte code execution over time. The JVM runs in two modes, **-server** and **-client**. If you use the default **-client** mode, there will be a faster startup time and a smaller memory footprint, but lower extended performance. You can enhance performance by using **-server** mode if a sufficient amount of time is

allowed for the HotSpot JVM to warm up by performing continuous execution of byte code. In most cases, use **-server** mode, which produces more efficient run-time execution over extended periods. You can monitor the process size and the server startup time to check the difference between **-client** and**-server**.

The **-server | -client** option is not supported on OS/400.

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 started, the process uses the JVM settings for the operating system of the node.

Data type	String

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*. To instruct the server not to use the Web server's document root and to use instead the Web application's context root for sendRedirect() calls, configure the JVM by setting the com.ibm.websphere.sendredirect.compatibility property to a `true` or `false` value.

1. Access the settings page for a property of the JVM.
   a. Click **Servers > Application Servers** in the console navigation tree.
   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, under Server Infrastructure, click **Java and Process Management > Process Definition**.
   d. On the Process Definition page, click **Java Virtual Machine**.
   e. On the Java Virtual Machine page, click **Custom Properties**.
   f. On the Custom Properties page, click **New**.

2. On the settings page for a property, specify a name of `com.ibm.websphere.sendredirect.compatibility` and either `true` or `false` for the value, then click **OK**.
3. Click **Save** on the console task bar.
4. Stop the application server and then restart the application server.

Setting custom JVM properties

In the WebSphere Application Server administrative console, you can change the values of the following custom JVM properties:

com.ibm.websphere.bean.delete.sleep.time

Specifies 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.

Steps for this task

1. Connect to the administrative console and navigate to the Java Virtual Machine Custom Properties panel. For example, **Application Server** >*server1* **> Process Definition** >*Servant* **> Java Virtual Machine > Custom Properties** to define the property in the Servant.
2. If the **com.ibm.websphere.bean.delete.sleep.time** property is not present in the list, create a new property name.
3. Enter the name and value.

com.ibm.websphere.network.useMultiHome

For the z/OS platform:

In a multihomed environment where WebSphere Application Server 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, WebSphere Application Server will only listen 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 WebSphere Application Server 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.

Steps for this task

1. To set this property, connect to the administrative console and navigate to the indicated page.

Application server	**Servers > Application Servers >***server1* **> Process Definition >***Control* **> Java Virtual Machine > Custom Properties**
Deployment manager	**System Administration > Deployment Manager > Process definition >** *Control* **> Java Virtual Machine > Custom Properties**
Node agent	**System Administration >Node Agent >** *nodeagent* **> Process definition >***Control* **> Java Virtual Machine > Custom Properties**

2. If the **com.ibm.websphere.network.useMultiHome** property is not present in the list, create a new property name and indicate its value.
3. To ensure the correct host is specified for the `End Point` property, navigate to the indicated page.

Application server	**Servers > Application Servers >***server1*. Then, under Communications, click **Ports >***port_name*
Deployment manager	**System Administration > Deployment Manager**. Then, under Additional Properties, click **Ports***port_name*
Node agent	**System Administration >Node Agent >** *nodeagent*. Then, under Additional Properties, click **Ports >***port_name* **Note:** For the node agents, there are MULTICAST ports; NODE_MULTICAST_DISCOVERY_ ADDRESS (IPv4) and NODE_MULTICAST_IPV6_DISCOVERY_ ADDRESS (IPv6). This IP Address must be a CLASS D IP address, first octet is 224-239.

4. Restart the server.

com.ibm.websphere.deletejspclasses

Deletes JavaServer Pages classes for all applications after those applications have been deleted or updated. By default, the value of this property is `true`.

Steps for this task

1. Connect to the administrative console and navigate to the Java Virtual Machine Custom Properties panel.

   For the z/OS platform:

Base configuration	**Servers > Application Servers >***server1* **> Process Definition > Servant > Java Virtual Machine > Custom Properties**
ND configuration	**System Administration > Node Agents >***nodeagent* **> Process definition > Control > Java Virtual Machine > Custom Properties**

2. If the **com.ibm.websphere.deletejspclasses** property is not present in the list, create a new property name.
3. Enter the name and value.

com.ibm.websphere.deletejspclasses.delete

Deletes JavaServer Pages classes for all applications after those applications have been deleted, but not after they have been updated. By default, the value of this property is `true`.

Steps for this task

1. Connect to the administrative console and navigate to the Java Virtual Machine Custom Properties panel.

   For the z/OS platform:

Base configuration	**Servers > Application Servers >***server1* **> Process Definition > Servant > Java Virtual Machine > Custom Properties**
ND configuration	**System Administration > Node Agents >***nodeagent* **> Process definition > Control > Java Virtual Machine > Custom Properties**

2. If the **com.ibm.websphere.deletejspclasses.delete** property is not present in the list, create a new property name.
3. Enter the name and value.

com.ibm.websphere.deletejspclasses.update

Deletes JavaServer Pages classes for all applications after those applications have been updated, but not after they have been deleted. By default, the value of this property is `true`.

Steps for this task

1. Connect to the administrative console and navigate to the Java Virtual Machine Custom Properties panel.

   For the z/OS platform:

Base configuration	**Servers > Application Servers >***server1* **> Process Definition > Servant > Java Virtual Machine > Custom Properties**
ND configuration	**System Administration > Node Agents >***nodeagent* **> Process definition > Control > Java Virtual Machine > Custom Properties**

2. If the **com.ibm.websphere.deletejspclasses.update** property is not present in the list, create a new property name.
3. Enter the name and value.

jmx.async.timeout

The **jmx.async.timeout** custom property is associated with the asynchronous JMX connector. By setting this property, you can override the default timeout of 180 seconds (three minutes), which allows you to more closely control the behavior of the asynchronous JMX connector. If the timeout value is too short, JMX operations may fail to complete. If the timeout value is too long, JMX operations that take a long time to complete (for instance, because of a slow servant, lengthy MBean implementation, or server error) may slow down JMX clients.

Steps for this task

1. Connect to the administrative console and navigate to the Java Virtual Machine Custom Properties page:

   Servers > Application Servers >*server1* **> Custom Properties**

2. If the **jmx.asynch.timeout** property is not present in the list, create a new property name.
3. Enter the name and value (in milliseconds).

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.

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 run-time environment to support applications.

1. Create an application server.
2. Create a virtual host.
3. Configure a Web container. See the *Developing and deploying applications* PDF for a description of how to perform this step.
4. Configure an EJB container. See the *Developing and deploying applications* PDF for a description of how to perform this step.
5. Create resources for data access. See the *Developing and deploying applications* PDF for a description of how to perform this step.
6. Create a JDBC provider and data source. See the *Developing and deploying applications* PDF for a description of how to perform this step.
7. Create a URL and URL provider. See the *Developing and deploying applications* PDF for a description of how to perform this step.
8. Create a JavaMail session.
9. Create resources for session support. See the *Developing and deploying applications* PDF for a description of how to perform this step.
10. Configure a Session Manager. See the *Developing and deploying applications* PDF for a description of how to perform this step.

Java memory tuning tips

Enterprise applications written in the Java language involve complex object relationships and utilize large numbers of objects. Although, the Java language automatically manages memory associated with object life cycles, understanding the application usage patterns for objects is important. In particular, verify the following:

- The application is not over-utilizing objects
- The application is not leaking objects
- The Java heap parameters are set properly to handle a given object usage pattern

Understanding the effect of garbage collection is necessary to apply these management techniques.

The garbage collection bottleneck

Examining Java garbage collection gives insight to how the application is utilizing memory. Garbage collection is a Java strength. By taking the burden of memory management away from the application writer, Java applications are more robust than applications written in languages that do not provide garbage collection. This robustness applies as long as the application is not abusing objects. Garbage collection normally consumes from 5% to 20% of total execution time of a properly functioning application. If not managed, garbage collection is one of the biggest

bottlenecks for an application. The Java virtual machine (JVM) uses a parallel garbage collector to fully exploit an SMP during most garbage collection cycles where the Sun HotSpot 1.3.1 JVM has a single-threaded garbage collector.

The garbage collection gauge

You can use garbage collection to evaluate application performance health. By monitoring garbage collection during the execution of a fixed workload, you gain insight as to whether the application is over-utilizing objects. Garbage collection can even detect the presence of memory leaks.

You can monitor garbage collection statistics using the **verbose:gc**JVM configuration setting. The **verbose:gc** format is not standardized between different JVMs or release levels. For a description of the IBM verbose:gc output and more information about the IBM garbage collector, see the See the *Tuning* PDF for a description of how to perform this task.

For this type of investigation, set the minimum and maximum heap sizes to the same value. Choose a representative, repetitive workload that matches production usage as closely as possible, user errors included.

To ensure meaningful statistics, run the fixed workload until the application state is steady.

Detecting over-utilization of objects

You can check if the application is overusing objects, by observing the counters for the JVM runtime. You have to set the **-XrunpmiJvmpiProfiler** command line option, as well as the JVM module maximum level in order to enable the Java virtual machine profiler interface (JVMPI) counters. The best result for the average time between garbage collections is at least 5-6 times the average duration of a single garbage collection. If you do not achieve this number, the application is spending more than 15% of its time in garbage collection.

If the information indicates a garbage collection bottleneck, there are two ways to clear the bottleneck. The most cost-effective way to optimize the application is to implement object caches and pools. Use a Java profiler to determine which objects to target. If you can not optimize the application, adding memory, processors and clones might help. Additional memory allows each clone to maintain a reasonable heap size. Additional processors allow the clones to run in parallel.

Detecting memory leaks

Memory leaks in the Java language are a dangerous contributor to garbage collection bottlenecks. Memory leaks are more damaging than memory overuse, because a memory leak ultimately leads to system instability. Over time, garbage collection occurs more frequently until the heap is exhausted and the Java code fails with a fatal `Out of Memory` exception. Memory leaks occur when an unused object has references that are never freed. Memory leaks most commonly occur in collection classes, such as Hashtable because the table always has a reference to the object, even after real references are deleted.

High workload often causes applications to crash immediately after deployment in the production environment. This is especially true for leaking applications where the high workload accelerates the magnification of the leakage and a memory allocation failure occurs.

The goal of memory leak testing is to magnify numbers. Memory leaks are measured in terms of the amount of bytes or kilobytes that cannot be garbage collected. The delicate task is to differentiate these amounts between expected sizes of useful and unusable memory. This task is achieved more easily if the numbers are magnified, resulting in larger gaps and easier identification of inconsistencies. The following list contains important conclusions about memory leaks:

- **Long-running test**

  Memory leak problems can manifest only after a period of time, therefore, memory leaks are found easily during long-running tests. Short running tests can lead to false alarms. It is sometimes difficult to know when a memory leak is occurring in the Java language, especially when memory usage has seemingly increased either abruptly or monotonically in a given period of time. The reason it is hard to detect a memory leak is that these kinds of increases can be valid or might be the intention of the developer. You can learn how to differentiate the delayed use of objects from completely unused objects by running applications for a longer period of time. Long-running application testing gives you higher confidence for whether the delayed use of objects is actually occurring.
- **Repetitive test**

  In many cases, memory leak problems occur by successive repetitions of the same test case. The goal of memory leak testing is to establish a big gap between unusable memory and used memory in terms of their relative sizes. By repeating the same scenario over and over again, the gap is multiplied in a very progressive way. This testing helps if the number of leaks caused by the execution of a test case is so minimal that it is hardly noticeable in one run.

  You can use repetitive tests at the system level or module level. The advantage with modular testing is better control. When a module is designed to keep the private module without creating external side effects such as memory usage, testing for memory leaks is easier. First, the memory usage before running the module is recorded. Then, a fixed set of test cases are run repeatedly. At the end of the test run, the current memory usage is recorded and checked for significant changes. Remember, garbage collection must be suggested when recording the actual memory usage by inserting System.gc() in the module where you want garbage collection to occur, or using a profiling tool, to force the event to occur.
- **Concurrency test**

  Some memory leak problems can occur only when there are several threads running in the application. Unfortunately, synchronization points are very susceptible to memory leaks because of the added complication in the program logic. Careless programming can lead to kept or unreleased references. The incident of memory leaks is often facilitated or accelerated by increased concurrency in the system. The most common way to increase concurrency is to increase the number of clients in the test driver.

  Consider the following points when choosing which test cases to use for memory leak testing:
  - A good test case exercises areas of the application where objects are created. Most of the time, knowledge of the application is required. A description of the scenario can suggest creation of data spaces, such as adding a new record, creating an HTTP session, performing a transaction and searching a record.
  - Look at areas where collections of objects are used. Typically, memory leaks are composed of objects within the same class. Also, collection classes such as Vector and Hashtable are common places where references to objects are implicitly stored by calling corresponding insertion methods. For example, the get method of a Hashtable object does not remove its reference to the retrieved object.

Heap consumption indicating a possible leak during a heavy workload (the application server is consistently near 100% CPU utilization), yet appearing to recover during a subsequent lighter or near-idle workload, is an indication of heap fragmentation. Heap fragmentation can occur when the JVM can free sufficient objects to satisfy memory allocation requests during garbage collection cycles, but the JVM does not have the time to compact small free memory areas in the heap to larger contiguous spaces.

Another form of heap fragmentation occurs when small objects (less than 512 bytes) are freed. The objects are freed, but the storage is not recovered, resulting in memory fragmentation until a heap compaction has been run.

Heap fragmentation can be reduced by forcing compactions to occur, but there is a performance penalty for doing this. Use the Java **-X** command to see the list of memory options.

Java heap parameters

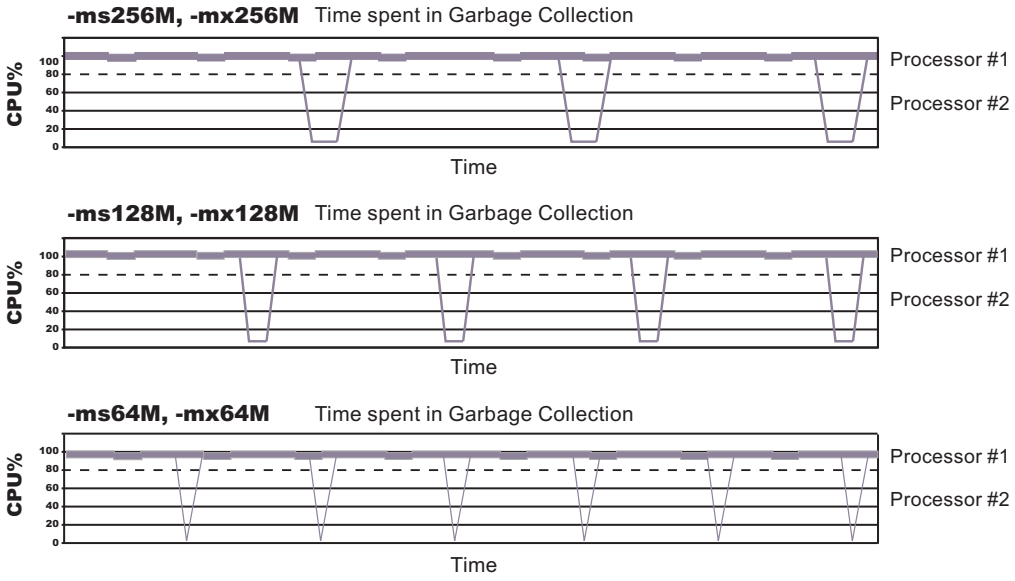
The Java heap parameters also influence the behavior of garbage collection. Increasing the heap size supports more object creation. Because a large heap takes longer to fill, the application runs longer before a garbage collection occurs. However, a larger heap also takes longer to compact and causes garbage collection to take longer.

Note that Java Heap information is contained in SMF records and can be viewed dynamically using the console command DISPLAY,JVMHEAP.

For performance analysis, the initial and maximum heap sizes should be equal.

When tuning a production system where the working set size of the Java application is not understood, a good starting value for the initial heap size is 25% of the maximum heap size. The JVM then tries to adapt the size of the heap to the working set size of the application.

Varying Java Heap Settings



The illustration represents three CPU profiles, each running a fixed workload with varying Java heap settings. In the middle profile, the initial and maximum heap sizes are set to 128MB. Four garbage collections occur. The total time in garbage collection is about 15% of the total run. When the heap parameters are doubled to 256MB, as in the top profile, the length of the work time increases between garbage collections. Only three garbage collections occur, but the length of each garbage collection is also increased. In the third profile, the heap size is reduced to 64MB and exhibits the opposite effect. With a smaller heap size, both the time between garbage collections and the time for each garbage collection are shorter. For all three configurations, the total time in garbage collection is approximately 15%. This example illustrates an important concept about the Java heap and its relationship to object utilization. There is always a cost for garbage collection in Java applications.

Run a series of test experiments that vary the Java heap settings. For example, run experiments with 128MB, 192MB, 256MB, and 320MB. During each experiment, monitor the total memory usage. If you expand the heap too aggressively, paging can occur. If paging occurs, reduce the size of the heap or add more memory to the system. When all the runs are finished, compare the following statistics:

- Number of garbage collection calls
- Average duration of a single garbage collection call
- Ratio between the length of a single garbage collection call and the average time between calls

If the application is not over-utilizing objects and has no memory leaks, the state of steady memory utilization is reached. Garbage collection also occurs less frequently and for short duration.

If the heap free space settles at 85% or more, consider decreasing the maximum heap size values because the application server and the application are under-utilizing the memory allocated for heap.

For current information available from IBM Support on known problems and their resolution, see the IBM Support page.

IBM Support has documents that can save you time gathering information needed to resolve this problem. Before opening a PMR, see the IBM Support page.

Testing and production phases

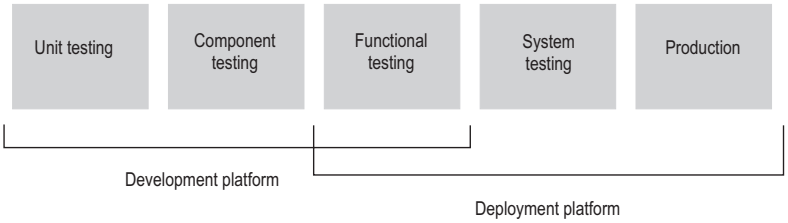
Before explaining the test and production configurations for WebSphere Application Server for z/OS, you must understand which test phase should be done on the z/OS platform and which should be done on other platforms.

Note: Sharing resources between a production workload and a test workload potentially can expose the production workload to a set of error conditions to which it would not be exposed if the production and test workloads ran in different cells. For this reason, you should run production and test workloads in separate cells on your system.

Before explaining the test and production configurations for the Application Server, you must understand which test phase 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

The development platform for WebSphere Application Server for z/OS is a WebSphere Application Server distributed system (for example, Windows or Linux Intel). The development environment includes tools such as WSAD for Web content delivery. The IBM tooling solution assumes that you develop enterprise beans in WSAD and perform unit (basic) testing of the business logic in the WebSphere test environment.

Component test phase

Component testing involves the joining together of several beans into logical components, providing them with access to data, and testing them together. While this can be done on WebSphere Application Server for z/OS, most of our current installations perform this level of testing using a distributed platform such as Windows 2000 running WebSphere Application Server Advanced Edition. This allows a small team of developers to join the pieces of code that they have developed together and test the interactions. This testing does not test z/OS platform functions and features directly, but focuses on the individual beans and the relationships between them.

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 is dependent on what the function is, and what its data requirements are. If the target deployment platform is z/OS, then it may make sense to do this level of testing there. This is possible by setting up one or more **test servers** into which the application is installed.

When the application is installed into the test server, the installer defines where in the JNDI directory the references to the application will be stored. The test clients will need to be configured with this information that tells the clients the location of the test application. The test clients will then drive requests against the test server to perform the functional testing. You can also 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 deploy the code into a WebSphere Application Server for z/OS server for testing. You may bring up the application and simulate a real load on the application. The important point here is that the code needs to run on z/OS before it goes into production. To do this, you need to define an additional **test server** (on a whole new cell

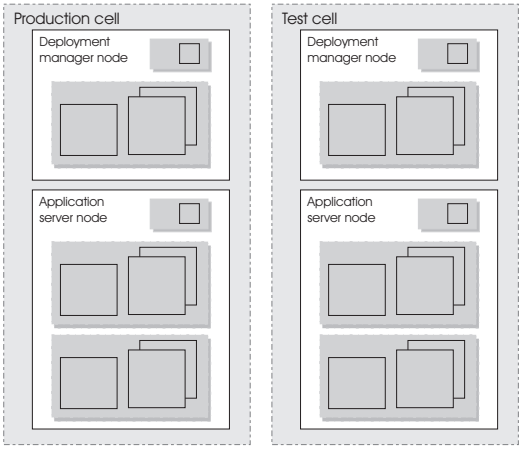
dedicated to the test system) and install the application into it. When installed, beans that are part of the application should be registered in a different subtree of the JNDI directory (this occurs by default). The test clients need to be configured to the version of the application that is being tested and the tests run.

Production phase

You can install the application in a production WebSphere Application Server for z/OS cell after you are satisfied with the functional and system testing. 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 someone flowed a remote debugging request to it.

Test cells and production cells

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. If you require complete availability of your production system, this configuration eliminates the risk of including production and test in the same cell.



Configuring multiple network interface card support

Use this task to first prepare your server for multiple network interface card support and enable multiple network interface card logic, including configuring multiple network interface cards to coexist on the same machine.

1. Configure object request broker (ORB) properties to support multiple network interface cards.
   - Set the `com.ibm.CORBA.LocalHost` property to resolve to a valid host name or IP address. The value should not resolve to a local host or loop back address (for example, 127.0.0.1).

   a. In the administrative console, click **Servers > Application servers >** *server_name* **> Java and process management > Process Definition > Java Virtual Machine**.

   b. Add the following generic JVM argument:

   ```
   -Dcom.ibm.CORBA.LocalHost=xxxx.xx.xx.xx -Dcom.ibm.websphere.network.
       useMultiHome=false -Dcom.ibm.CORBA.ServerSocketQueueDepth=50 -Dcom.
       ibm.ws.orb.transport.useMultiHome=false
   ```

2. Update the host name for all HTTP transport chains. In the administrative console, click **Servers > Application servers >** *server_name* **> Web container settings > Web container transport chains**. Update the host name for all of the transport chains listed on this page. The value should not resolve to a local host or loop back address (for example, 127.0.0.1).

3. Change the initial state to **stopped** for each of the listener ports. In the administrative console, click:
   - **Servers > Application servers >** *server_name* **> Messaging > Message listener service > Listener ports > SamplePtoPListenerPort**
   - **Servers > Application servers >** *server_name* **> Messaging > Message Listener Service > Listener Ports > SamplePubSubListenerPort**

   Update the state to **stopped** for each port.
4. In a Network Deployment environment, you must also add the following generic Java virtual machine (JVM) arguments and make sure all nodes are in sync.
   a. In the administrative console, click **System administration > Deployment manager > Java and process management > Process definition > Java virtual machine**. Add the following generic JVM argument:

      ```
      -Dcom.ibm.CORBA.LocalHost=xxxx.xx.xx.xx -Dcom.ibm.websphere.network.
          useMultiHome=false -Dcom.ibm.CORBA.ServerSocketQueueDepth=50 -Dcom.
          ibm.ws.orb.transport.useMultiHome=false
      ```

   b. In the administrative console, click **System administration > Node agents >***node_agent_server* **> Java and process management > Process definition > Java virtual machine**. Add the following generic JVM argument:

      ```
      -Dcom.ibm.CORBA.LocalHost=xxxx.xx.xx.xx -Dcom.ibm.websphere.network.
          useMultiHome=false -Dcom.ibm.CORBA.ServerSocketQueueDepth=50 -Dcom.
          ibm.ws.orb.transport.useMultiHome=false
      ```
5. Change the initial state of the JMS server to stopped. In the administrative console, click **Servers > JMS Servers >***JMS_server* **> Initial state > stopped**.
6. Apply these changes to all application servers and the deployment manager.

By completing these steps, you enabled multiple network interface card support.

To configure multiple application servers to coexist on a single machine that is using two network interface cards, perform the following steps:

1. Install the WebSphere Application Server base product for each network interface card. To install on distributed platforms, see for more information.
2. Start the server that is on the first network interface card. Follow the preceding steps in this task to prepare this server for multiple network interface card support and enable multiple network interface card logic. After you complete this step, stop the server.
3. Start the server that is on the other network interface card. Follow the preceding steps in this task to prepare this server for multiple network interface card support and enable multiple network interface card logic. After you complete this step, stop the server.
4. Start the servers on both network interface cards.

By completing these steps, you enabled multiple application servers to coexist on a single machine that has two network interface cards.

Tuning application servers

The WebSphere Application Server contains interrelated components that must be harmoniously tuned to support the custom needs of your end-to-end e-business application.

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.

- **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 tune the ORB with the following parameters:
  - Set **Pass by reference (com.ibm.CORBA.iiop.noLocalCopies)** as described in the *Developing and deploying applications* PDF.
  - Set the **com.ibm.CORBA.FragmentSize** as described in the *Developing and deploying applications* PDF.
- **Tune the XML parser definitions.**
  - **Description:** Facilitates server startup by adding XML parser definitions to the `jaxp.properties` and `xerxes.properties` files in the ${*install_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.
              StandardParserConfiguration
    ```
  - **Default value:** None
  - **Recommended value:** None
- **Tune the dynamic cache service.** Using the dynamic cache service can improve performance. See the *Developing and deploying applications* PDF for information about using the dynamic cache service and how it can affect your application server performance.
- **Tune the EJB container.** An 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 *Developing and deploying applications* PDF.
  - **Break CMP enterprise beans into several enterprise bean modules** while assembling EJB modules.

  See also the *Developing and deploying applications* PDF.
- **Tune the session management.**

  The installed default settings for session management are optimal for performance. See the *Tuning* PDF for more information about tuning session management.
- **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.
  - Review information on connection pooling, contained in the *Developing and deploying applications* PDF, to understand how the number of physical connections within a connection pool can change performance.
  - Use the *Tuning* PDF as a reference for the data source and connection pool properties that most affect performance.

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 allowing 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 *Developing and deploying applications* 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 `inProcessLogFilenamePrefix` 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 `inProcesslLogFilenamePrefix` 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`.

Application servers: Resources for learning

Use the following links to find relevant supplemental information about configuring application servers. 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 Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

Programming instructions and examples
- WebSphere Application Server education

Programming specifications
- The Java™ Virtual Machine Specification, Second Edition
- Sun's technology forum for the Java™ Virtual Machine Specification

Administration
- Listing of all IBM WebSphere Application Server Redbooks

Balancing workloads with clusters

Consider your options for configuring application servers. See "Managing application servers" on page 203 for more information.

To monitor application servers and manage the workloads of servers, use server clusters and cluster members.

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.

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. See "Replicating data across application servers in a cluster" on page 345 for more information. You can create a replication domain when creating a cluster.
3. Deploy the application onto the application server.
4. 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. See "Creating clusters" on page 333 for more information.
5. You can create one or more cluster members of the cluster.
6. Start all of the application servers by starting the cluster. Workload management automatically begins when you start the cluster members of the application server.
7. Once you have the cluster running, you can perform the following tasks:
   - Stop the cluster.
   - Upgrade applications on clusters.
   - Detect and handle problems with server clusters and their workloads.
   -

Clusters and workload management

Clusters are sets of servers that are managed together and participate in workload management. The servers that are members of a cluster can be on different host machines, as opposed to the servers that are part of the same node and must be located on the same host machine. A cell can have 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.

With z/OS, you can set up workload management to balance the tasks between different clusters.

WebSphere Application Server can respond to increased use of an enterprise application by automatically replicating the application to additional cluster members as needed. This lets you deploy an application on a cluster instead of on a single node, without considering workload.

Multiple servers that can service the same client request is 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. In fact, several servers could fail, and as long as at least one cluster member is running, client requests can continue to be serviced. For further information backing up failed processes, see "Replicating data across application servers in a cluster" on page 345.

Clusters and node groups

Node groups bound clusters. All cluster members of a given cluster must be members of the same node group.

Any application you install to a cluster must be able to execute on any application server that is a member of that cluster. For the application you deploy to run successfully, all the members of the cluster must be located on nodes that meet the requirements for that application.

In a cell that has many different server configurations, it might be difficult to determine which nodes have the capabilities to host your application. A *node group* can be used to define groups of nodes that have enough in common to host members of a given cluster. All cluster members in a cluster must be in the same node group.

All nodes are members of at least one node group. When you create a cluster, the first application server you add to the cluster defines the node group that bounds the cluster. All other cluster members you add to the cluster can only be on nodes that are members of this same node group. When you create a new cluster member in the administrative console, you are allowed to create the application server on a node that is a member of the node group for that cluster only.

Nodes can be members of multiple node groups. If the first cluster member you add to a cluster has multiple node groups defined, the system automatically chooses the node group that bounds the cluster. You can change the node group by modifying the cluster settings. Use the "Server cluster settings" on page 338 page to change the node group.

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 Web site. You might also find *z/OS MVS Programming: Workload Management Services* helpful.

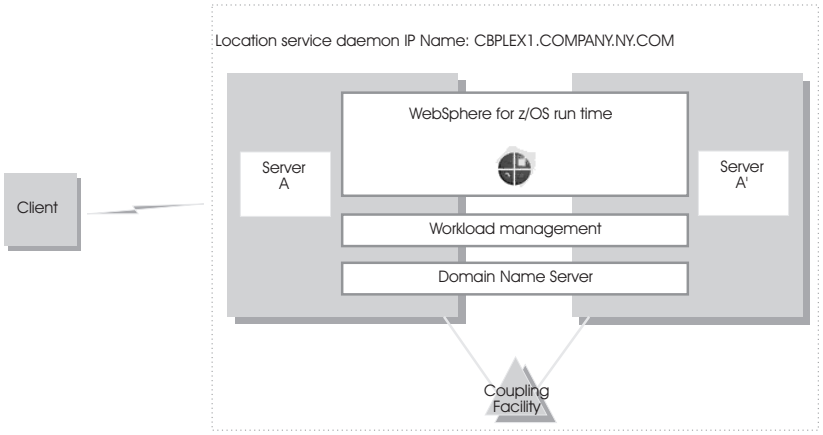
When you are using workload management on z/OS, you can define workload management policies for your application servers. To get started, you do not need to define special classification rules and work qualifiers, but you might want to define them for your production system.

Workload management provides the following benefits to WebSphere Application Server applications:

- 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 WebSphere Application Server 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 more information, see Workload management (WLM) for z/OS. For additional details on setting up servers for connection optimization, see *z/OS Communications Server: IP Configuration Reference*.

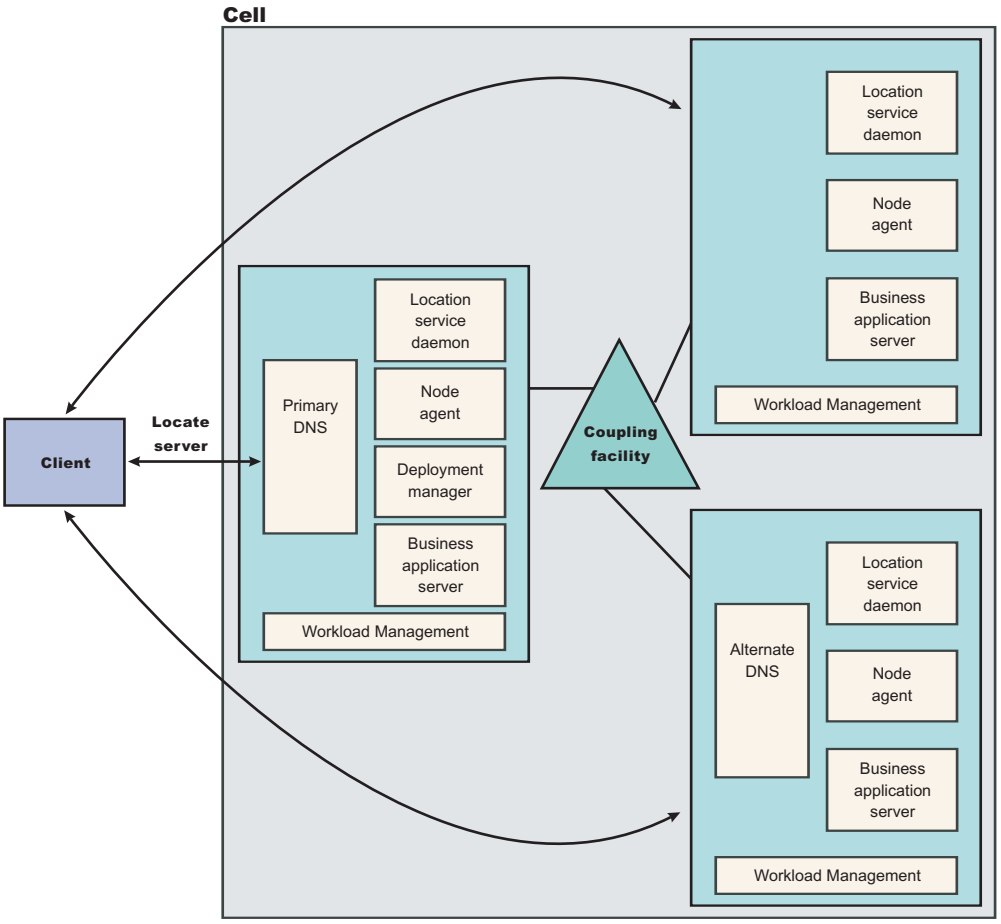
Sysplex routing of work requests

WebSphere Application Server for z/OS routes work requests throughout the cell by using the domain name server (DNS).

Instead using sysplex distributor to distribute workload, you can use a DNS that balances requests for the same host name across multiple IP addresses (one per daemon).

The DNS accepts a generic host name from the client and maps the name to a specific system. In order to select the best available system, the DNS asks workload management (WLM) for a recommendation. Workload management analyzes the current state of the cell and considers a number of factors, such as CPU, memory, and I/O utilization, to determine the best placement of new work. The DNS then routes the client request to the optimal system to run. This use of workload

management and the DNS is optional, however, it eliminates a single point of failure.



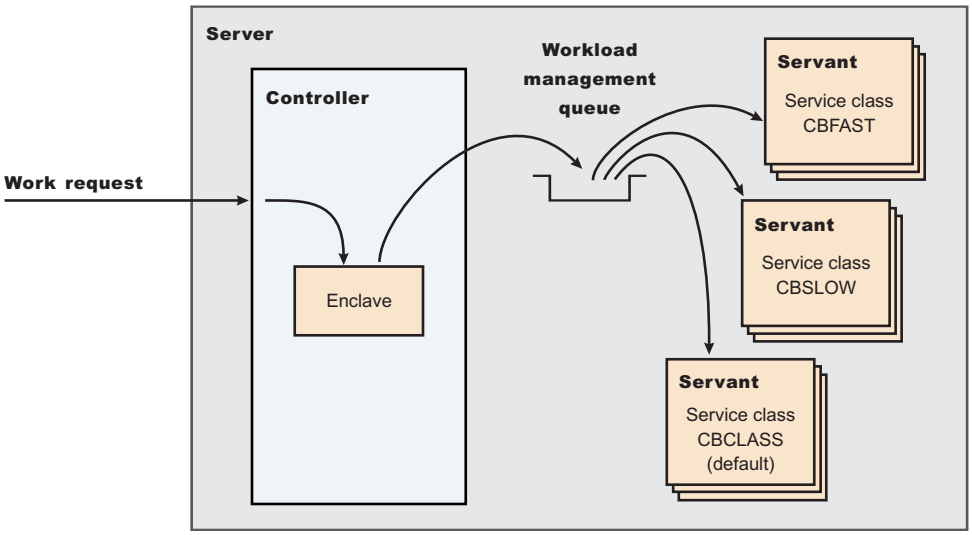
Each system in the cell has the WebSphere Application Server for z/OS runtime. All the systems contain a location service daemon, node agent, and business application servers. One system acts as the deployment manager for the cell. The client uses the CORBA General Inter-ORB Protocol (GIOP) to make requests of WebSphere Application Server for z/OS. 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, 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 WebSphere Application Server for z/OS uses depends on whether the client is local or remote. If the client is remote, that is , not running on the same z/OS system, the transport is TCP/IP. If the client is local, the transport is through a program call. Local transport is fast because it avoids a physical trip over the network, eliminates data transforms, simplifies the marshalling of requests, and uses optimized RACF facilities for security rather than having to invoke Kerberos or SSL.

Address space management for work requests

WebSphere Application Server for z/OS 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:

- WebSphere Application Server for z/OS uses its own set of rules to create an enclave for a client request from the network.
- Some subsystems (such as a Web server) create enclaves and pass them to the application server, which, in turn, passes the enclaves on.
- WebSphere Application Server for z/OS 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 8. WLM work qualifiers and corresponding WebSphere Application Server for z/OS entities

Work qualifier abbreviation	Work qualifier	Corresponding WebSphere Application Server for z/OS 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, see *z/OS MVS Planning: Workload Management* and "Classifying z/OS workload" on page 317.

In addition to client workloads, you must consider the performance of the WebSphere Application Server for z/OS 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 9. Workload management rules

If you are classifying...	... 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.
WebSphere Application Server 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.
WebSphere Application Server servant	A lower velocity and importance STC than the controller	If too high, you spend too much time in Java garbage collection, and you might consume more system resources than you want. If too low, JSP compiles and Java garbage collection could delay processing in your application. Startup of additional servant address spaces might also be delayed.
WebSphere Application Server 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.	

Example of classification rules

Purpose

Let us assume you have three workload management service classes defined for WebSphere Application Server for z/OS (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 10. Classification rules example

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 WebSphere for z/OS
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 WebSphere for z/OS
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 ******************************
```

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 WebSphere Application Server for z/OS run-time servers, gets classified as CBCLASS.

Enabling multiple servants on z/OS

Use this task to enable multiple servants to improve the performance of WebSphere Application Server for z/OS.

Review the limitations of enabling multiple servants. See "Multiple servant regions" for more information.

With Workload Management (WLM), you can manage the performance and number of servants in WebSphere Application Server for z/OS. 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.

1. In the administrative console, click **Servers > Application servers > *server_name* > Server instance**.
2. Select **Multiple instances enabled**.
3. Click **Apply** to finish the Server Instance changes.
4. Click **OK**.

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 servant regions

With Workload Management (WLM), you can manage the performance and number servants in WebSphere Application Server for z/OS. WLM manages the response time and throughput of WebSphere 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.

WebSphere Application Server applications are deployed within a WebSphere Application Server generic *server*. One or more server instances must be defined on one or more systems within the WebSphere Application Server *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.

If you have WLM dynamic application environments enabled on your system (z/OS Release 2 plus APAR OW54622 or later), WLM honors the specifications for the number of servant regions. If you are using static application environments (specified through the WLM ISPF panels), then you must also enable multiple servant regions by indicating **No limit** in the WLM ISPF panels.

Note: Be aware of the following regarding multiple servants:

- Some applications, such as the administrative console and the deployment manager, have serialization requirements that only work in a single Java virtual machine (JVM). These cannot be run in a server with multiple servants. Do not enable multiple instances for any of these servers.
- If you specify a maximum number of instances, WLM is restricted from starting more than this number of servant regions 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**. Remember to account for the *default CB-type service class* and enclaves that may originate outside WebSphere Application Server servers and are classified by other classification rules such as the IBM HTTP Server (IHS).

Controlling the 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 Workload Manager (WLM) to start up additional regions. 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.

1. Start the administrative console.
2. Click **Servers > 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 **OK**.

Classifying z/OS workload

Use this task to classify inbound requests by using a workload classification document, a common XML file that classifies inbound HTTP, IIOP, and message-driven bean (MDB) work for the z/OS workload manager.

You should be using workload management on a z/OS system. See "Workload management (WLM) for z/OS" on page 309 for more information.

By developing a workload classification document, you can classify incoming 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) to classify the work and assign a service class or a report service class.

With WebSphere Application Server for z/OS version 5.1 and later, you can classify inbound HTTP, IIOP, and MDB work in one document. You might have configured classification documents with a previous release using a different format, and specified these files using an advanced setting in the Web container or through the `endpoint_config_file` custom property. If you configure the new document that classifies HTTP, IIOP, and MDB work in one file, the new file overrides the old files. For example, if you have HTTP classification rules in the new workload classification document, these rules are used instead of other rules that you might have in the old style HTTP classification document. However, if your new document specifies only classification rules for EJB 2.0 message-driven beans deployed with listener ports, the old style HTTP classification document is still used.

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.

1. Develop the workload classification document. Use the information in the article,"Workload classification file" on page 324. See "Sample z/OS workload classification document" on page 321 for a sample of the workload classification document and the DTD.
2. If you create the document on z/OS in codepage IBM-1047, the normal codepage for files that exist in the HFS, you must 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 codepage to the ASCII codepage. 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 codepage to a different designated codepage. 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 > Manage 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. You must 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. Restart the application server to implement any changes to the file. 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 and HTTP 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:

   ```
   F X5SR02A,DISPLAY,WORK,CLINFO

   BBOO0281I CLASSIFICATION COUNTERS FOR IIOP WORK
   BBOO0282I CHECKED 14, MATCHED 14, USED 0, COST 0, DESC: IIOP Default
   BBOO0282I CHECKED 14, MATCHED 3, USED 0, COST 0, DESC: sample
   BBOO0282I CHECKED 3, MATCHED 1, USED 1, COST 3, DESC: a1a
   BBOO0282I CHECKED 2, MATCHED 1, USED 1, COST 4, DESC: a1b
   BBOO0282I CHECKED 1, MATCHED 1, USED 1, COST 5, DESC: a1c
   BBOO0282I CHECKED 11, MATCHED 11, USED 0, COST 0, DESC: other
   BBOO0282I CHECKED 11, MATCHED 1, USED 1, COST 4, DESC: a
   BBOO0282I CHECKED 10, MATCHED 1, USED 1, COST 5, DESC: b
   BBOO0282I CHECKED 9, MATCHED 1, USED 1, COST 6, DESC: c
   BBOO0282I CHECKED 8, MATCHED 2, USED 2, COST 7, DESC: d
   BBOO0282I CHECKED 6, MATCHED 1, USED 1, COST 8, DESC: e
   BBOO0282I CHECKED 5, MATCHED 4, USED 4, COST 9, DESC: f
   BBOO0282I CHECKED 1, MATCHED 1, USED 1, COST 10, DESC: g
   BBOO0283I FOR IIOP WORK: TOTAL CLASSIFIED 14, WEIGHTED TOTAL COST 95
   BBOO0281I CLASSIFICATION COUNTERS FOR HTTP WORK
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: HTTP Default
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: n
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: o
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: q
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: r
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: s
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: p
   BBOO0282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: t
   BBOO0283I FOR HTTP WORK: TOTAL CLASSIFIED 0, WEIGHTED
   BBOO0188I END OF OUTPUT FOR COMMAND DISPLAY,WORK,CLINFO
   ```

   An explanation of the command output follows:

   BBOO0281I 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 or IIOP. You cannot display inbound MDB classifications.

   BBOO0282I 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 this times that this rule has been matched by the request.
   - <n3> - The number of times that this rule has actually been used.
   - <n4> - The cost of using the rule, or the number of compares that are required to determine if this 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.

BBOO0283I FOR <type> WORK: TOTAL CLASSIFIED <n1>, WEIGHTED TOTAL COST <n2> This message shows the summary information for the HTTP or IIOP work classification. This message displays the following information:

- <type> - The type of work that is being displayed. The value must be IIOP or HTTP.
- <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 of 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.

By completing this task, you classified inbound requests by using a workload classification document.

See "Balancing workloads with clusters" on page 307 for more information about configuring workload management.

Sample z/OS workload classification document

Sample

```
<?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="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>
     <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>

<!--
     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>

<!--
   Workload Classification Document for P5SR01x servers
   Change History
   ------------------------------------------------------
   Activity                    Date                Author
   Created                   01-28-2005          IPL

--->

</Classification>
```

DTD for the workload classification XML document

```
From SERV1/ws/code/messaging.impl.ws390/src/Classification.dtd
SERV1/ws/code/messaging.impl.ws390/src/Classification.dtd
  WAS601.SERV1      WAS.messaging      1.2        2004/12/10 18:46:00
  1.2                           640    no         2004/12/10 18:46:00
<?xml version='1.0' encoding="UTF-8"?>
<!ELEMENT Classification (InboundClassification|SibClassification)+>
<!ATTLIST Classification schema_version CDATA #REQUIRED>
<!ELEMENT InboundClassification ((iiop_classification_info+|http_
      classification_info+|endpoint+))>
<!ATTLIST InboundClassification type (iiop|mdb|http) #REQUIRED>
<!ATTLIST InboundClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST InboundClassification schema_version CDATA #REQUIRED>
<!ELEMENT iiop_classification_info (iiop_classification_info*)>
<!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>
<!ATTLIST iiop_classification_info transaction_class CDATA #REQUIRED>
<!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>
<!ATTLIST classificationentry classification CDATA #REQUIRED>
<!ATTLIST classificationentry selector CDATA #REQUIRED>
<!ATTLIST classificationentry description CDATA #IMPLIED>
<!ELEMENT http_classification_info (http_classification_info*)>
<!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 #REQUIRED>
<!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>
<!ATTLIST sib_classification_info transaction_class CDATA #REQUIRED>
<!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>
```

Workload classification file

The workload classification document is a common XML file that classifies inbound HTTP, IIOP, message-driven bean (MDB), and mediation work for the z/OS workload manager.

Usage notes

You must perform the task "Classifying z/OS workload" on page 317 when you use the workload classification document. See "Sample z/OS workload classification document" on page 321 for an example of a workload classification document.

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 "Classification.dtd">
Gives the XML parser with the name of the DTD document provided by WebSphere Application Server for z/OS that validates 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.

InboundClassification

```
<InboundClassification type="iiop | http | mdb"
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 iiop, http, or mdb. Only one occurrence of an InboundClassification element can occur in the document for each type. There can be up to three 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.

The rules and XML statements for classifying different types of work are very 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

- "IIOP Classification"
- "HTTP classification" on page 327
- "MDB classification" on page 329

SibClassification

- "JMS RA classification" on page 330
- "Mediation classification" on page 332

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 2 Platform, Enterprise Edition (J2EE) 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 of the artifacts.
- Module name

The name of the 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"] >
```

With the iiop_classification_info element, you can build filters based on the application, module, component, and method names to assign TCLASS values to inbound requests. Use the following rules when using the iiop_classification_info element:

- The transaction_class attribute must be specified. 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.
- 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 trasaction 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 carded value. You can use an asterisk (*) to match strings. The string MAY* matches any string that starts with MAY.
  - Any value. To specify a match to any value, use the asterisk (*) symbol.

  Use any or all of 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 application_name attribute.
- 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 parent element's attribute value, the lower level filter is negated. An example of a child value that conflicts with the parent element's attribute value 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 is true to 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"] >
```

With the http_classification_info element, you can build filters based on the host, port, and URI to assign TCLASS values to inbound requests. Use the following rules when you use the http_classification_info element:

- The transaction_class attribute must be specified. 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.
- The attributes 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.
  - A wild carded value. You can use an asterisk (*) to match strings. The string MAY* matches any string that starts with MAY.
  - Any value. To specify a match to any value, use the asterisk (*) symbol.

  Use any or all of 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.
- 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 parent element's attribute value 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="&apos;East&apos;"
                      classification="MDB1"
                      description="DEF" />
```

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 < and > symbols with the entity references `<` and `>`, respectively. Similarly, if you use an apostrophe or quotation mark, use the `&apos;` and `&quot;` entity references.

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  selector="&apos;East&apos;"
                      transaction_class="sibb"
                      selector="user.Location=&apos;East&apos;"
                      bus="bigrred"
                      destination="abusqueue"
                      description="Some words" />
```

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 must match exactly to the selector clause in the MDB deployment descriptor. It 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 *Developing and deploying applications* PDF.

- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < and > symbols with the entity references `<` and `>`, respectively. Similarly, if you use an apostrophe or quotation mark, use the `&apos;` and `&quot;` 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 depends 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 depends 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.

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 317.

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;"
                   destination="themoon"
                   discriminator="sides/dark"
                   description="n" />
```

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 must match exactly to the selector clause in the mediation deployment descriptor. It 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 *Developing and deploying applications* PDF.
- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < and > symbols with the entity references `<` and `>`, respectively. Similarly, if you use an apostrophe or quotation mark, use the `&apos;` and `&quot;` 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 depends 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 depends 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.

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 317.

Creating clusters

Use this task to create clusters, which are sets of application servers that are managed together and participate in workload management.

You can manage application servers collectively using a cluster. Use the "Server cluster collection" on page 338 to view and manage the cluster.

Note: On the z/OS platform, 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. For more information, see "Considerations for clustered servers and stateful session beans" on page 340.

1. **Enter basic cluster information.**
   a. In the administrative console, click **Servers > Clusters > New**.
   b. Create a name for the cluster.
   c. To enable or disable node-scoped routing optimization, select **Prefer local**. The default is enabled, which indicates that, if possible, Enterprise JavaBeans (EJB) requests are routed to the client node. If you enable this feature, performance is improved because client requests are sent to local enterprise beans.
   d. To create a replication domain for this cluster, select **Create a replication domain for this cluster**. Use replication domains to transfer data, objects, or events for session manager, dynamic cache, or stateful session beans among the application servers in a cluster. Create a separate replication domain to use with each component that acts as a consumer of the replication. For example, you can configure one replication domain to use with a session

manager and another domain to use with dynamic cache. The replication domain name that is created is identical to the cluster name. See "Replicating data across application servers in a cluster" on page 345 for more information.

e. Choose whether to create an empty cluster or to create a cluster based on an existing server.

   To create an empty cluster, select **Do not include an existing server in this cluster**.

   To create a cluster based on an existing server, choose **Select an existing server to add to this cluster** and select the server you want to add. The server you add becomes a template for any additional cluster members that you add to the cluster. Be sure that the template is configured correctly before adding more cluster members that are based on this template. Note that this is the only time you are able to add an existing server to the cluster. After you create the first cluster member, you cannot add other existing application servers to the cluster. Be careful when adding an existing server because the only way to remove an application server from a cluster is to delete the application server. Consider using the existing server as a template for the cluster members but not as a cluster member. Keeping the original application server out of the cluster allows you to reuse the template if you need to rebuild the configuration.

f. If you chose to add an existing server, specify the server weight. 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 assigned to other servers in the cluster, then the 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 the z/OS platform, 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 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.

2. **Create cluster members.**

   Note: With WebSphere Application Server Version 6.0, you can upgrade a portion of the nodes in a cell, while leaving others at the older release level. This means that, for a period of time, you might be managing servers that are at the current release and servers that are running the newer release in the same cell. Note that in this mixed environment, there are some restrictions on what you can do with clusters and cluster members:

   - You cannot create a cluster member using a server running on WebSphere Application Server Version 5.x.
   - You cannot add a member that is running WebSphere Application Server Version 5.x to any cluster.

- You can add a new WebSphere Application Server Version 6.0 cluster member to a mixed cluster only if the cluster already contains a WebSphere Application Server Version 6.0 member that was upgraded from WebSphere Application Server Version 5.x. All new cluster members must be running WebSphere Application Server Version 6.0.

For each new cluster member, perform the following actions:

a. Type the name of a new application server (cluster member) to add to the cluster.
b. Select the node on which the server resides.
c. Make sure that **Generate Unique HTTP Port** is selected.
d. Specify the server template. You can choose the default application server template or an existing application server template. All the application servers you add after the first application server use the same template.
e. Click **Apply** to finish the cluster member. You can add more cluster members. All cluster members you add are based on the same server template.

3. **View the summary.** View a summary of the changes and then click **Finish**. Your cluster is created.
4. Define a virtual host with a unique port number. See "Configuring virtual hosts" on page 160 for more information.
   a. In the administrative console, click **Environment** > **Virtual Hosts > default_host > Host Aliases**.
   b. Click **New** and, on the settings page for a virtual host that displays, specify an administrative name for the virtual host.
   c. Under **Additional Properties**, click **Host Aliases** and define a unique port number for the virtual host.

   Note: Setting up a virtual host is optional on z/OS.
5. Save your configuration. As part of saving the change to the configuration, you can select **Synchronize changes with Nodes** before clicking **Save** on the Save page.
6. Before you can start the cluster, the configuration needs to be synchronized to the nodes. If you selected **Synchronize changes with Nodes** when saving your configuration in the previous step, you can ignore this step. If you are running automatic synchronization, wait until synchronization runs. Or, run manual synchronization to get the configuration files moved to the nodes. Click **System administration > Nodes** and, on the Nodes page, select the node and click **Synchronize** or **Full resynchronize**. The Nodes page displays status indicating whether the node is synchronized.
7. To further configure a cluster, click **Servers > Clusters**. To view the settings for the cluster, click on the cluster **Name**. Note that unless you have clicked **Save** and saved your administrative configuration, you only see the **Configuration** and **Local Topology** tabs; to see the **Runtime** tab you must save your administrative configuration. Also, ensure that changes are synchronized to the nodes (step 7).

You can create cluster members or start the cluster. See "Balancing workloads with clusters" on page 307 for more information about cluster configuration options.

Use scripting to automate the task of creating clusters. See the *Administering applications and their environment* PDF for more information.

Creating a cluster : Basic cluster settings

Use this page to enter the basic cluster settings.

To view this administrative console page, click **Servers > Clusters > New**.

Cluster name:

Specifies the name of the cluster. The cluster name must be unique within the cell.

Prefer local:

Specifies that the node scoped routing optimization is enabled or disabled. The default is enabled, which means that Enterprise JavaBeans (EJB) requests are routed to the client node when possible. Enabling this setting improves performance because client requests are sent to local enterprise beans.

Create replication domain for this cluster:

Specifies that when the cluster is created, a replication domain is also created with the cluster. The replication service transfers both Java 2 Platform, Enterprise Edition (J2EE) application data and any internal data that is used to maintain the application data among WebSphere Application Server run-time processes in a cluster of application servers.

Create a separate replication domain to use with each component that acts as a consumer of the replication. For example, you can configure one replication domain to use with a session manager and another domain to use with dynamic cache. The replication domain name that is created is identical to the cluster name. To manage the replication domain that is created, click **Environment > Replication domains** in the administrative console.

Existing server:

Specifies whether to create an empty cluster or to add an existing application server to the cluster.

To create a cluster without an existing application server as a cluster member, click **Do not include an existing server in this cluster**.

To add an existing server to the cluster, click **Select an existing server to add to this cluster** and choose the application server to add to the cluster. This application server becomes a template for any additional cluster members that you add to the cluster. When adding servers to a cluster, the only way to remove an application server from a cluster is to delete the application server.

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 the z/OS platform, 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 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.

Creating a cluster : Basic cluster member settings

Use this page to enter cluster member settings. You can create a cluster member during or after cluster creation.

You can create a new application server to become a cluster member in two ways:

- Create a cluster member when you create the cluster. In the administrative console, click **Servers > Clusters > New** in the administrative console.
- Create a cluster member after you create the cluster. In the administrative console, click **Servers > Clusters >** *cluster_name* **> Cluster Members > New**.

Member name:

Specifies the name of the application server that is created for the cluster.

The member name must be unique within the cell.

Select node:

Specifies the node on which the application server resides.

Core group: Specifies the core group in which the application server resides. This field is displayed only if you have multiple core groups configured. You can change this value for the first cluster member only.

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 template:

Specifies an application server template for the new application server.

The new application server contains identical configuration settings to the application server template.

Click **Existing application server** to choose from a list of application server templates. The application server that you choose becomes a template for the cluster member.

The template options are available only for the first cluster member that is created. All cluster members that you create after the first member are identical.

Creating a cluster : Summary settings

Use this administrative console page to save settings that you modified when creating a cluster or cluster member.

You can view this administrative console page when creating a new cluster or a new cluster member. This summary page displays your configuration changes before you commit the changes and a new cluster or cluster member is created. To create a cluster or cluster member, click the following paths in the administrative console:

- Click **Servers > Clusters > New** to create a new cluster.
- Click **Servers > Clusters >** *cluster_name* **> Cluster Members > New** to create new application servers for an existing cluster.

Review the changes to your configuration. Click **Finish** to complete and save your work. The bounding node group of the cluster is based on the first application server that is added as a member of the cluster. To select a different bounding node group, click **Servers > Clusters >***cluster_name* in the administrative console to edit the settings for the cluster.

Server cluster collection

Use this page to view information about and manage clusters of application servers.

To view this administrative console page, click **Servers > Clusters**.

Click **New** to access the Create New Cluster page, which you use to define a new cluster.

Name:

Specifies a logical name for the cluster. The name must be unique among clusters within the containing cell.

Status:

Specifies whether cluster members are stopped, starting, or running.

If all cluster members are stopped, the cluster status and state is *stopped*. After you request to start a cluster by clicking **Start** or **Ripplestart**, 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 *partially started*. The state remains *partially started* until all cluster members are running, then the state changes to *running* and the status is *started*. Similarly, when stopping a cluster by clicking **Stop** or **ImmediateStop**, the state changes to *partially stopped* as the first member stops and changes to *stopped* when all members are not running.

Server cluster settings

Use this page to view or change the configuration and local topology of a server cluster instance. Provided you saved your administrative configuration after creating the server cluster instance, you can also view run-time information such as the status of the server cluster instance.

To view this administrative console page, click **Servers > Clusters >***cluster_name*.

Cluster name:

Specifies a logical name for the cluster. The name must be unique among clusters within the containing cell.

Data type String

Cluster short name:

Specifies the cluster short name for this cluster. For clustered servers, the WLM application environment is the cluster short name. The cluster short name must be unique in the cell and cannot equal the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Clustered servers should not have a `ClusterTransitionName` set.

The name is 1-8 characters, alpha-numeric or national language. It cannot start with a number.

Data type String

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 bounds 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 WebSphere Application Server nodes. A node is a logical grouping of managed servers, usually on a computer 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 high availability for persistent services:

Specifies that the recovery logs for persistent services reside on storage devices that have been configured according to high availability requirements.

State:

Specifies whether cluster members are 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.

Data type	String
Range	Valid values are starting, partially started, running, partially stopped, or stopped.

Considerations for clustered servers and stateful session beans

For a cluster of servers that span multiple systems in a sysplex and 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. So, 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 WebSphere administrative console, see the *Developing and deploying applications* PDF.

Creating cluster members

Use this task to create application servers to be members of a configured cluster.

Create a cluster. See "Creating clusters" on page 333 for more information.

You create a cluster member to represent an application server in a cluster. To create a cluster member, view information about cluster members, or manage members of a cluster, use the Cluster Members page.

Note: With WebSphere Application Server Version 6.0, you can now upgrade a portion of the nodes in a cell, while leaving others at the older release level. This means that, for a period of time, you might be managing servers that are at the current release and servers that are running the newer release in the same cell. Note that in this mixed environment, there are some restrictions on what you can do with clusters and cluster members:

- You cannot create a cluster member using a server that is running on WebSphere Application Server Version 5.x.
- You cannot add a member that is running WebSphere Application Server Version 5.x to any cluster.
- You can add a new WebSphere Application Server Version 6.0 cluster member to a mixed cluster only if the cluster already contains a WebSphere Application Server Version 6.0 member that was upgraded from WebSphere Application Server Version 5x. All new cluster members must be running WebSphere Application Server Version 6.0.

1. Click **Servers > Clusters** in the administrative console. Then, click a cluster in the collection of clusters and click **Cluster members**. The Cluster members page lists members of a cluster, states the nodes on which members reside, and states whether members are started, stopped or encountering problems.
2. Click **New** and follow the steps on the Create new cluster members page.
   a. Type a name for the cluster member (application server).
   b. Select the node for the server.
   c. Specify the server weight. 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 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.

      On the z/OS platform, 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 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.

   d. Specify whether to generate a unique HTTP port.
   e. Specify the server template.
   f. Click **Apply** to finish the cluster member. Repeat these steps to define another cluster member.
   g. Click **Next**.
   h. Review the summary of information on new cluster members and click **Finish**.
3. Click **Save** to save your administrative configuration.

4. To examine a cluster member's settings, click on the member's name under **Member name** on the Cluster members page. This displays the settings page for the cluster member instance.

You created application servers that became members of an existing server cluster.

You can automate the task of adding cluster members by using scripting. See the *Administering applications and their environment* PDF for more information.

Cluster member collection

Use this page to view information about and manage members of an application server cluster.

To view this administrative console page, click **Servers > Clusters >***cluster_name* **> Cluster 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.

Version:

Specifies the version of WebSphere Application Server that is on the cluster member.

Status:

Specifies whether a cluster member is running, stopped, or unavailable.

If a cluster member is stopped, its status is `Stopped`. After you request to start a cluster member by clicking **Start**, the status becomes `Started`. After you click **Stop**, its status changes to `Stopped` when it stops running.

Note that 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 cluster member.

Cluster member settings

Use this page to configure a member instance of an application server cluster.

To view this administrative console page, click **Servers > Clusters >***cluster_name* **> Cluster members >***cluster_member_name*.

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.

Data type String

Weight:

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 assigned to other servers in the cluster, then the server receives a larger share of the server workload.

On the z/OS platform, 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 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

Starting clusters

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. See “Java virtual machine settings” on page 289 for more information on how to change the debug port.

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.

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*.

Note: From the z/OS MVS console, you need to start each individual server that you wish to run. With the administrative console, you can start each server individually or start a cluster which will start all defined servers automatically.

1. Click **Servers > Clusters** in the console navigation tree to access the Server Cluster page.
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.

By starting the members of a cluster, you automatically enabled workload management.

See "Balancing workloads with clusters" on page 307 for more information about the tasks that you can complete with clusters.

Stopping clusters

Use this task to stop a cluster and any application servers that are members of that cluster.

You can stop all application servers that are members of a cluster at the same time by stopping the cluster. That is, you can stop all application servers in a server cluster at the same time.

1. Click **Servers > 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*.

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 WebSphere for z/OS environment on a system, stop that system Daemon. Stopping the system Daemon brings down all other server instances on the system. To bring WebSphere for z/OS down on all systems, stop the Daemons on all systems. Note that when you stop the location service daemon on one system, it does not bring down the servers on the other systems.

See "Balancing workloads with clusters" on page 307 for more information about the tasks you can complete with clustering.

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 WebSphere Application Server component that performs replication services, including replicating data, objects, and events among application servers.

If you configured a data replication domain with a previous version of WebSphere Application Server, you might be using a multi-broker replication domain. Any replication domains that you create with the current version of WebSphere Application Server are data replication domains. Migrate any multi-broker replication domains to data replication domains. To learn the differences between the two types of replication domains, see "Comparison of multi-broker versus data replication domains" on page 351 and "Migrating V6 servers from multi-broker replication domains to data replication domains" on page 349.

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. For more information about replication, see "Replication" on page 346.

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. See "Data replication domain settings" on page 348 for more information about the properties that you can configure for your replication domain.
   - **Create a replication domain when creating a cluster.**

     To create a replication domain when you create a cluster, click **Servers > Clusters > New** in the administrative console. Enable **Create replication domain for this cluster**. 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 >** *replication_domain_name* in the administrative console. See "Creating clusters" on page 333 for more information about creating a cluster.

   For more information about the replication domain settings that you can configure in the administrative console, see "Data replication domain settings" on page 348
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 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.

- To configure dynamic cache replication, see the *Developing and deploying applications* PDF..
- To configure memory-to-memory replication for session manager, see the *Developing and deploying applications* PDF..
- To configure replication of stateful session beans, see the *Developing and deploying applications* PDF..

Data is replicating among the application servers in a configured replication domain.

If you use the Data Encryption Standard (DES) or DES3 encryption type for a replication domain, an encryption key is used for the encryption of messages. Click **Regenerate encryption key** on the "Data replication domain settings" on page 348 page at regular intervals to regenerate the key and protect your configuration. After the key has been changed or regenerated, you must restart all of the application servers that are configured as part of the replication domain. Consider performing this step every month to secure your configuration.

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 *Developing and deploying applications* 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 *Developing and deploying applications* 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.

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.

By having a single replica configuration defined, you can 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. Any replication domain that is used by dynamic cache must use a full group replica.

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.

To configure replication, see "Replicating data across application servers in a cluster" on page 345.

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 the information does not exist.

Units	seconds
Default	5 seconds

Encryption type:

Specifies the type of encryption to use when transferring replicated data to another area of the network. The options are NONE, DES, and DES3. The default is NONE. The DES and DES3 options encrypt data sent between application server processes (for example, session manager and dynamic caching) to better secure the network joining the processes.

If you specify DES or DES3, a key for global data replication is generated after you click **Apply** or **OK**. If you use the DES or DES3 encryption type, click **Regenerate encryption key** at regular intervals, for example once each month. Periodically regenerating the key enhances security.

Data type	String
Default	NONE

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	One replica is created. This is the default value.
Full group replica	Each object is replicated to every application server that is configured as a consumer of the replication domain.
Specific number of replicas	A custom number of replicas for any entry that is created in the replication domain.

Migrating V6 servers from multi-broker replication domains to data replication domains

Use this task to migrate multi-broker replication domains to data replication domains. Multi-broker domains were created with a previous version of WebSphere Application Server.

For HTTP session affinity to continue working correctly when migrating V5.x application servers to V6.0 application servers, you must upgrade all of the Web server plug-ins for WebSphere Application Server to the latest version before upgrading the application servers that perform replication.

After you upgrade your deployment manager to the latest version of WebSphere Application Server, you can create data replication domains only. Any multi-broker domains that you created with a previous release of WebSphere Application Server are still functional, however, you cannot create new multi-broker domains or replicators with the administrative console.

The different versions of application servers cannot communicate with each other. When migrating your servers to the current version of WebSphere Application Server, keep at least two application servers running on the previous version so that replication remains functional.

Perform this task on any multi-broker domains in your configuration after all of your servers that are using this multi-broker domain have been migrated to the current version of WebSphere Application Server. For more information about the differences between multi-broker domains and the data replication domains, see the *Developing and deploying applications* PDF.

The following examples illustrate the migration process for common configurations:

Migrating an application server configuration that uses an instance of data replication service in peer-to-peer mode

Use this migration path to migrate a replication domain that uses the default peer-to-peer configuration. Dynamic cache replication domains use the peer-to-peer topology. See Memory-to-memory topology: Peer-to-peer function for more information.

Before you begin, migrate all the Web server plug-ins for your application server cluster to the current version.

1. Migrate one or more of your existing servers to the current version of WebSphere Application Server.

2. In the administrative console, create an empty data replication domain. Click **Environment > Replication domains > New** in the administrative console. See "Replicating data across application servers in a cluster" on page 345 for more information about creating a data replication domain.
3. Add your migrated application servers to the new data replication domain. For example, if you are migrating 4 servers, migrate 2 servers first and add them to the new replication domain. Configure the servers to use the new domain by configuring the consumers of the replication domain.
4. When the new data replication domains are successfully sharing data, migrate the rest of the servers that are using the multi-broker replication domain to data replication domains.
5. Delete the empty multi-broker replication domain.

Migrating an application server configuration that uses an instance of the data replication service in client/server mode

Use this set of steps to migrate a replication domain that uses 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:

1. Migrate one or more of your existing servers to the current version of WebSphere Application Server.
2. In the administrative console, create an empty data replication domain. Click **Environment > Replication domains > New** in the administrative console. See "Replicating data across application servers in a cluster" on page 345 for more information about creating a data replication domain.
3. Add your migrated servers to the new data replication domain. For example, if you are migrating 4 servers, migrate two of these servers and then add them to the new replication domain. Configure the servers to use the new domain by configuring the consumers of the replication domain.
4. Add a part of the clients to the new data replication domain.
5. When the new data replication domains are successfully sharing data, migrate the rest of the clients and servers that are using the multi-broker replication domain to data replication domains.
6. Delete the empty multi-broker replication domain.

Migrating a replication domain that uses HTTP session memory-to-memory replication that is overloaded at the application or web module level

1. Upgrade your deployment manager to the current version of WebSphere Application Server. All the application servers remain configured with the old multi-broker domains on the previous version of WebSphere Application Server.
2. In the administrative console, create an empty data replication domain. Click **Environment > Replication domains > New** in the administrative console. See "Replicating data across application servers in a cluster" on page 345 for more information about creating a data replication domain.
3. Migrate each application server to the current version of WebSphere Application Server, one at a time. The remaining servers on the previous

version of WebSphere Application Server can still communicate with each other, but not with the migrated servers. The migrated servers can also communicate with each other.

4. Continue migrating all of the servers to the current version of WebSphere Application Server. All of the application servers are still using multi-broker replication domains, so the features of data replication domains cannot be used.
5. Configure all of the application servers to use the new data replication domain, adding the application servers to the empty replication domain that you created.
6. Restart all of the application servers in the cluster.
7. Delete the empty multi-broker replication domain.

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 session manager and another replication domain for dynamic cache. See "Replicating data across application servers in a cluster" on page 345 for more information.

Comparison of multi-broker versus data replication domains

Data replication domains replace multi-broker domains for data replication between application servers in a cluster.

Any replication domains that are created with a previous version of WebSphere Application Server might be multi-broker domains. Migrate any multi-broker domains to the new data replication domains. Although you can configure existing multi-broker domains with the current version of WebSphere Application Server, after you upgrade your deployment manager, you can create only data replication domains in the administrative console.

Multi-broker and data replication domains both perform the same function, which is to replicate data across the consumers in a replication domain. Configure all the instances of replication that need to communicate in the same replication domain. You can also configure the session manager with both types of replication domains to use topologies such as peer-to-peer and client/server to isolate the function of creating and storing replicas on separate application servers. You can control the redundancy of replication for each type of replication domain. With a data replication domain, you can specify a specific number of replicas.

If you used multi-broker domains with earlier releases of WebSphere Application Server, use the following comparison chart to learn the differences between how V5.x and V6.0 application servers use the two types of replication domains:

	V5.x application servers using replication domains	V6.0 application servers using replication domains
Replication domain types	Uses only multi-broker replication domains for replication.	Servers that are using the current version of WebSphere Application Server 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 V5.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 WebSphere Application Server. 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 WebSphere Application Server did not support data replication domains. The current version of WebSphere Application Server 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.

	V5.x application servers using replication domains	V6.0 application servers using replication domains
Replication domain configuration	The earlier version of WebSphere Application Server encourages the sharing of replication domains between different consumers, such as session manager and dynamic cache.	The current version of WebSphere Application Server 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.
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.

	V5.x application servers using replication domains	V6.0 application servers using replication domains
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.

To migrate multi-broker domains to data replication domains, see "Migrating V6 servers from multi-broker replication domains to data replication domains" on page 349.

Replicating data with a multi-broker replication domain

Use this task to work with replication domains that were created and used with a previous version of WebSphere Application Server.

Multi-broker domains were created with a previous version of WebSphere Application Server, but remain functional in the current version. Although you can configure existing multi-broker domains with the current version of WebSphere Application Server, 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. See "Migrating V6 servers from multi-broker replication domains to data replication domains" on page 349 for more information about the benefit of migrating your replication domains.

If you do not have any existing replication domains, see "Replicating data across application servers in a cluster" on page 345 for information about creating new data replication domains.

If you are performing this task, it is assumed that you configured replication with a previous version of WebSphere Application Server 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.

1. Manage multi-broker replication domain configuration settings. In the administrative console, click **Environment > Replication domains**.

2. Click on a multi-broker domain. Specify the values for a particular multi-broker replication domain. The default values are generally sufficient, especially for the pooling and timeout values.
   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 WebSphere 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 new 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. If you use the DES or TRIPLE_DES encryption type for a replicator, click **Regenerate encryption key** on the settings for a replication domain instance at regular intervals, such as monthly.

   Periodically changing the key enhances security.

Multi-broker replication domains

A multi-broker replication domain is a collection of replicator entry (or *replicator*) instances used by clusters or individual servers within a cell. Multi-broker replication domains were created with a previous release of WebSphere Application Server.

Note: After you upgrade your deployment manager to the latest version of WebSphere Application Server, you can create data replication domains only. Any multi-broker domains that you created with a previous release of WebSphere Application Server are still functional, however, you cannot create new multi-broker domains or replicators with the administrative console. See "Comparison of multi-broker versus data replication domains" on page 351 for more information.

A replication entry (or *replicator*) is a run-time component that handles the transfer of internal WebSphere Application Server 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 WebSphere Application Server functions (for example, 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 WebSphere Application Server. 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 ame 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 the information does not exist. The default is 5 seconds.

Data type	Integer
Units	Seconds
Default	5

Encryption type:

Specifies the type of encryption used before the object transfers over the network. The options include NONE, DES, TRIPLE_DES. The default is NONE. The DES and TRIPLE_DES options encrypt data sent between WebSphere Application Server processes and secure the network joining the processes.

If you specify DES or TRIPLE_DES, a key for global data replication is generated after you click **Apply** or **OK**. When you use the DES or TRIPLE_DES encryption type, click **Regenerate encryption key** at regular intervals such as monthly because periodically changing the key enhances security.

DRS partition size:

Specifies the number of groups into which a replication domain is partitioned. By default, data sent by a WebSphere Application Server process to a replication domain is transferred to all other WebSphere 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 a WebSphere 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.

Note: 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 2, Enterprise Edition (J2EE) environment where class definitions might reside 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 WebSphere 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 WebSphere Application Server. When you migrate your deployment manager to the current version of WebSphere Application Server, 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 WebSphere Application Server.

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 that has cluster members.

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.

1. Click **Servers > Clusters** in the console navigation tree to access the Server Cluster page.
2. Make sure the cluster you want to remove is **stopped**. If the cluster is **started**, see "Stopping clusters" on page 344.
3. Remove the cluster. Select the cluster and click **Delete**.
4. Save your configuration. Click **Save** on the administrative console task bar. As part of saving the change to the configuration, select **Synchronize changes with Nodes** before clicking **Save** on the Save page.

The cluster is deleted.

Deleting cluster members

Use this task to remove a cluster member from an existing cluster. Removing a cluster member deletes the associated application server. You cannot remove an application server from a cluster without deleting it.

1. Choose the cluster that contains your cluster member. Click **Servers > Clusters** in the console navigation tree to access the Server Cluster page. Select the cluster for your cluster member and click **Cluster members**.
2. Make sure the cluster member you want to remove is **stopped**. If the cluster is **started**, see "Stopping servers" on page 235.
3. Remove the cluster member from the cluster. Select the cluster member you want and click **Delete**.
4. Save your configuration. Click **Save**. As part of saving the change to the configuration, select **Synchronize changes with Nodes** before clicking **Save** on the Save page.

The cluster member is deleted.

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.

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 "Enabling multiple servants on z/OS" on page 316 for more information.

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 WebSphere Application Server 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 "WLM even distribution of HTTP requests" on page 362.

1. Add the **wlm_stateful_session_placement_on** WebSphere variable for your application server configuration settings.
   a. Click **Environment > Manage WebSphere variables**, and select the server for which you want to use the WLM even distribution of HTTP requests function.
   b. Click **New**.
   c. Add a new variable with `wlm_stateful_session_placement_on` for **name** and 1 for **value**.
   d. Click **Apply** and then click **Save and Synchronize** to update your changes.
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 > 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 have set up your classification mapping file to specify more than one transaction class on a mapping rule for WebSphere Application Server managed round robin support, you should remove this section from your classification mapping file. For example, if you have a line in your classification mapping file like the following:

   ```
   TransClassMap *:8080 /Dynacache1Web1/Servlet1 TCLASS1 TCLASS2 TCLASS3
   ```

   Modify this line of the classification mapping file to have only one transaction class, like the following example:

   ```
   TransClassMap *:8080 /Dynacache1Web1/Servlet1 TCLASS1
   ```

   You also must update the z/OS workload manager policy to remove the extra service classes that were only necessary to get WebSphere Application Server managed round robin support. Following is an example of removing 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 `wlm_stateful_session_placement_on` variable after it is restarted.

The application server uses the WLM even distribution of HTTP requests function to handle its workload instead of showing affinity to a certain servant.

See "Detecting and handling problems with run-time components" on page 235 to handle problems with server clusters and workloads.

WLM even distribution of HTTP requests

The z/OS workload management (WLM) component in z/OS 1.4 and later 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 WebSphere Application Server for z/OS 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.

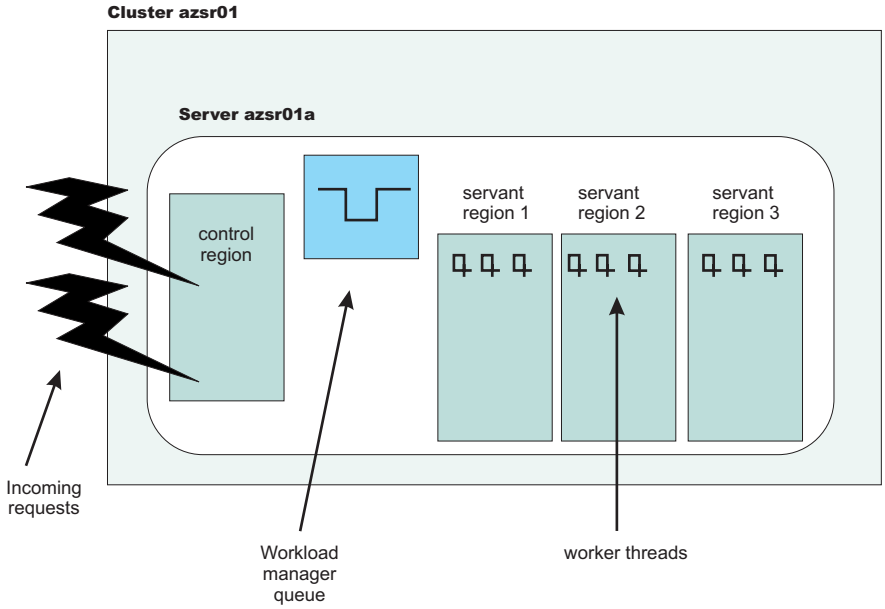


Figure 1. The contents of one clustered server instance

In the preceding diagram, the application server is configured to have the minimum and maximum number of servants set to three. For information about configuring the number of servants, see "Controlling the number of servants" on page 317.

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* server strategy

WebSphere Application Server for z/OS 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.

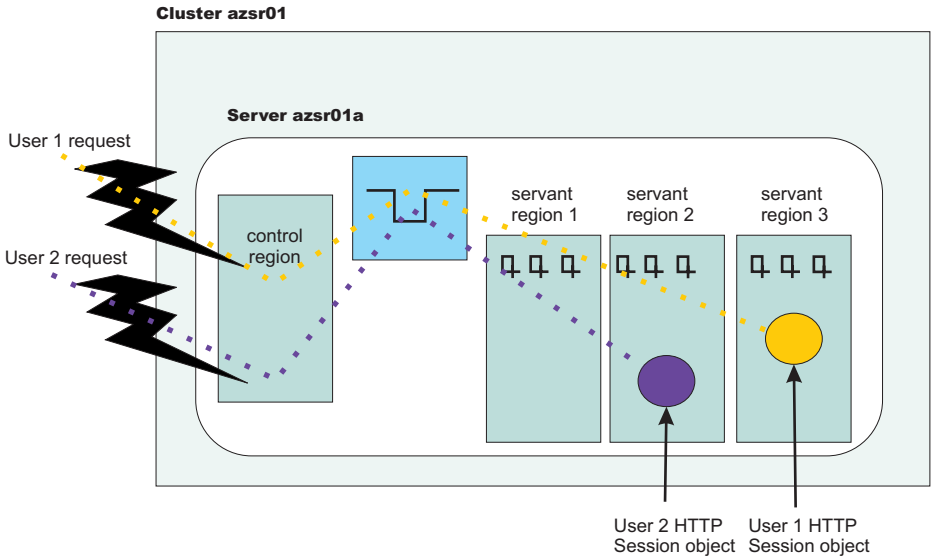


Figure 4. Users establish HTTP session objects

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.

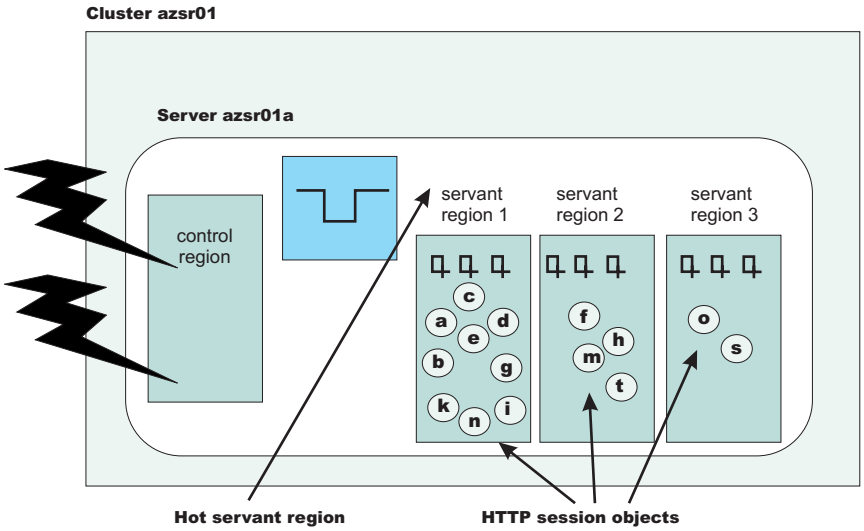


Figure 5. HTTP Session objects assigned to a hot servant

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.

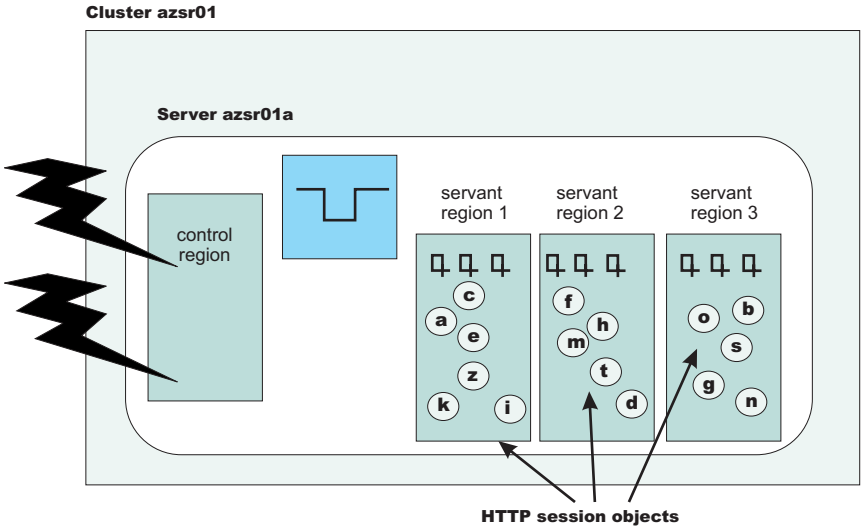


Figure 6. HTTP Session objects assigned to servants without affinity

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 WebSphere Application Server managed round robin support, you should remove this section from your classification mapping file.

For more information about configuring an application server to enable the distribution of incoming HTTP requests across servants without affinity, see "Configuring an application server to use the WLM even distribution of HTTP requests function" on page 361.

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 the operator commands.

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
```

Clustering and workload management: Resources for learning

Use the following links to find relevant supplemental information about clustering and workload management. 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 Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

Programming model and decisions

- IBM WebSphere V5.0 Performance, Scalability, and High Availability: WebSphere Handbook Series

  Note: The information presented in this Redbook is not specific to z/OS. It is written for WebSphere Application Server Network Deployment.
- IBM WebSphere Application Server V5.0 System Management and Configuration: WebSphere Handbook Series
- Workload Manager for z/OS and OS/390
- MVS Planning: Workload Management
- MXI (MVS eXtended Information)

Programming instructions and examples

- WebSphere Application Server education
- Listing of all IBM WebSphere Application Server Redbooks
- Listing of z/OS Redbooks
- Writing Optimized Java Applications for z/OS

Setting up a high availability environment

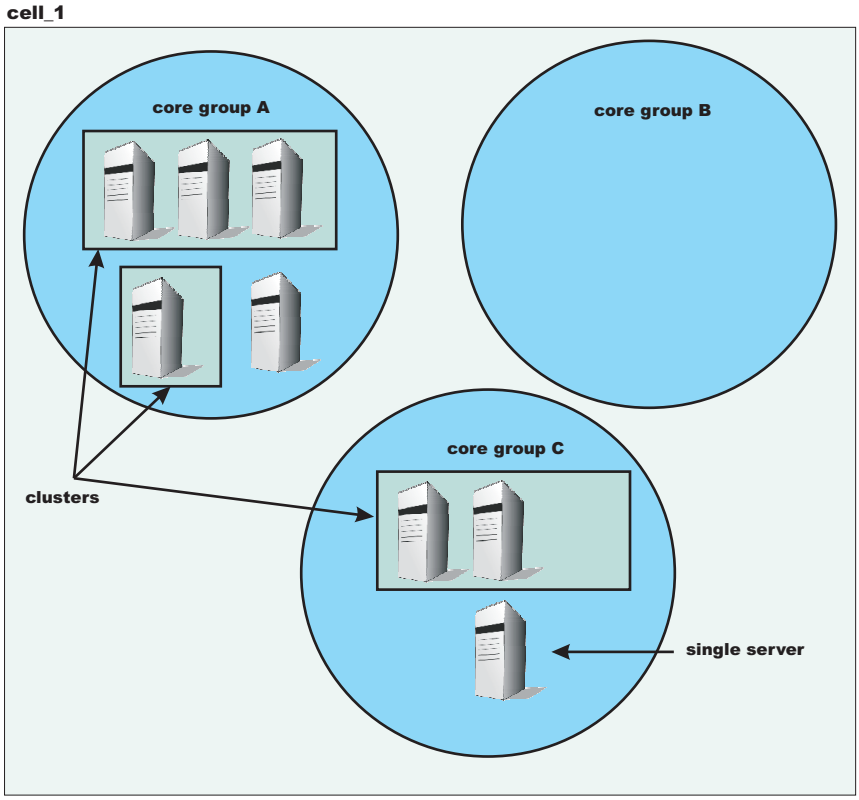
Planning ahead for high availability support is important to avoid the risk of a failure without failover coverage. The application server infrastructure that is managed by a high availability manager includes cells and clusters. These components relate closely to core groups, high availability groups, and the policy that controls the high availability infrastructure.

The high availability manager is designed to function in all of the supported WebSphere Application Server topologies. However, a high availability-managed environment must comply to the following rules:

- A cell in a high availability infrastructure is partitioned into one or more core groups. WebSphere Application Server provides a default core group as part of the high availability manager function. Additional core groups can be created using the administrative console.
- A core group cannot extend beyond the boundaries of a cell, and it cannot overlap with any other core groups.

- A cluster must be a member of only one core group. All of the individual members of that cluster must be members of the same core group. This one-to-one relationship between a cluster and a core group exists for both static and dynamic clusters.
- Individual application servers that are part of the high availability environment must also be part of a core group.
- An application server can only join a core group if its JVM can communicate with all of the other online application servers that are part of that core group. If a single application server can not open a connection to the JVM or send a heartbeat to it, the application server is not joined to the core group.

The following diagram illustrates what a cell might look like in a high availability environment:



Important: After you set up your WebSphere Application Server environment to comply with all of the high availability-managed environment rules, use the default core group to control this environment. DO NOT add additional core groups unless your environment absolutely requires them. Also, do not change the default configurations unless you are doing so to solve a specific problem or situation. When you do make configuration changes, such as changing the policy for a high availability group or moving core group members between core groups in a multi-core group environment, make sure you fully understand the effect such changes will have on your entire environment.

Following are tasks you might perform if you need to change the default configuration:

1. Create additional core groups, if required..
2. Modify the attributes of an existing core group.
3. Modify the attributes of an existing high availability group policy.

4. Create a new policy and associate it with a high availability group.
5. Create core group access points if you create additional core groups.
6. Set up IP addresses for communications within the core group if you are installing the Application Server on a system with multiple NICs.

High availability manager

WebSphere Application Server includes a high availability manager component whose main function is to eliminate single points of failure. This component runs inside every JVM in a WebSphere Application Server cell. A high availability manager is responsible for running key services on available application servers rather than on a dedicated one (such as the deployment manager). It takes advantage of fault tolerant storage technologies such as network attached storage (NAS), which significantly lowers the cost and complexity of high availability configurations. The high availability manager also provides peer-to-peer failover for critical services by always maintaining a backup for these services.

A high availability manager is responsible for managing the availability of singletons within the cell. Examples of singletons include:

- Transaction managers for cluster members.
- Messaging engines.
- Workload manager (WLM) controllers. (These controllers are responsible for gathering the end points for applications deploying in a cluster and aggregating that information to a single route table for that cluster).
- WLM routing information. (When a cluster member is hosting a resource that can be clustered, such as an application, a transaction manager or a messaging engine, this information needs to be shared between all of the controller JVMs in the cell.)
- Application servers.
- WebSphere Application Server partitioning facility instances.

A high availability manager continually monitors the application server environment. If an application server component fails, the high availability manager takes over the in-flight and in-doubt work for the failed server. This action significantly improves application server availability.

In a highly available environment, all single points of failure are eliminated. Because the high availability manager function is dynamic, any configuration changes that you make and save while an application server is running are eventually be picked up and used. You do not have to restart an application server to enable a change. For example, if you change a policy for a messaging engine high availability group while the messaging engine is running, the new policy is dynamically loaded and applied, and the behavior of the messaging engine reflects this change.

To provide this focused failover service, the high availability manager supervises the controller JVMs that are members of the core group. The high availability manager uses one of the following methods to detect failures:

- An application server is marked as failed if the socket fails. This method uses the KEEP_ALIVE function of TCP/IP, and is very tolerant of extreme application server loading, which might occur if the application server is swapping or thrashing heavily. This method is recommended for determining a JVM failure if

you are using multicast emulation and are pushing the application server into extreme CPU starvation or memory starvation.

- A JVM is marked as failed if it stops sending heartbeats for a specified time interval. This method is referred to as *active failure detection*. When it is used, a JVM sends out one heartbeat, or pulse every ten seconds. If the JVM is unresponsive for more than 200 seconds, it is considered down. You can use this method with multicast emulation. However, this method must be used for true multicast addressing.

In either case, if a controller JVM fails, the associated application servers are separated from the core group and any services running on those application servers are failed over to the surviving core group members.

All of the application servers in a cell are defined as members of a core group. Each core group has only one logical high availability manager that services all of the members of that core group. The high availability manager is responsible for making the services within a core group highly available and scalable. It continually polls all of the core group members to verify that they are active and healthy.

A policy matching program is used to localize certain policy-driven components and to place these components into high availability groups. When a core group member fails, the high availability manger assigns the failing member's work to the same type of component from the same high availability group. Using a shared HFS in the position of common logging facilities helps to recover in-doubt and in-flight work if a component fails.

WebSphere Application Server provides a default core group that is created during installation. New server instances are added to the default core group as they are created. The WebSphere Application Server environment can support multiple core groups, but one core group is usually sufficient for most environments.

A high availability manager is manages a variety of components. All of the components in a high availability manager infrastructure work together to ensure peer-to-peer failover effectively protects the application server environment from failures. The following table describes the main high availability components, or areas of components, required for an effective high availability manager environment:

Server component areas	The focus is on the application server run time, which includes such entities as cells and clusters. These areas are necessary for a healthy high availability manager run time because they closely relate to core groups, high availability groups, and the policy that defines the infrastructure.

Core groups	Provide failover support. A default core group is created during startup. This core group should be sufficient for most environments. Additional core groups can be created, but you should only create them if you fully understand the implications to your high availability environment. Core groups are static in nature. The configuration applied to a core group through user-defined policies determines the dynamic relationship within high availability groups.
High availability groups	Closely bound to policy definitions. High availability groups are dynamic in nature and are not configured directly by users. Policy match criteria determines the high availability group to which a core group member belongs.
Network components	Provide the underlying network infrastructure that is crucial to the success of the high availability manager. By default, WebSphere Application Server uses a channel framework protocol. However, a unicast or multicast protocol can also be used. The network components include a technology that enables communication throughout the high availability manager infrastructure.

All of these components must be active and properly configured to achieve a highly available infrastructure.

High availability network components

The foundation for a highly available environment is dependent on the network that is created for this environment. The WebSphere Application Server high availability manager, by default, uses a channel framework network protocol model to establish network connections. This model enables network ports to be shared among all of the channels within a transport chain. A unicast protocol, which supports communication between a specific receiver and a specific sender, or a multicast protocol, which supports open communication to all receivers, can also be used.

The following components are important parts of a high availability network:

- Distributed Consistency Services (DCS). This component implements JVM level group services. It is responsible for detecting when JVMs in a core group start and stop. It also tracks the set of servers that have joined the core group. The set of servers currently in the core group is a called the view. A JVM typically joins the view in 20-30 seconds after it initializes. This delay allows multiple JVMs to start together and to join the view together rather as a sequence of new views, each with a single additional member. High availability resources can only be activated or deactivated in a JVM after the JVM has joined the view.

  The high availability issues message HMGR218I to the system out logs whenever a view changes. This message occurs simultaneously across all current view members, and indicates the number of current view members. A view changes whenever a JVM stops or fails, and whenever a server fails or leaves the

view. A view change enables the high availability manager to quickly react to failures and recover any services running on the failed server.

- Reliable Multicast Messaging (RMM). This component is a very high-speed publish and subscribe message transport that DCS and the high availability manager use for JVM to JVM communication. It can operate using any of the following low-level transports:
  - Reliable Multicast. This is true multicast.
  - TCP unicast. This emulates multicast over unicasts.
  - Channel Framework. This emulates multicast over channel framework point to point streams.

A high availability environment ties all of the network components together. It provides the basis for providing peer-to-peer failover. The moment a component loses its network connection to the rest of the infrastructure, the high availability manager assumes a failure has occurred and assumes the work assigned to that component.

Thousands of high availability groups can exist within a core group. The policies that are associated with a high availability group control how and when members of that group are activated. A group services mechanism is used to communicate high availability group membership information between members of a core group. This information includes the group services that are available and the lightweight component groups that are part of a high availability group.

The group services mechanism also supports the following types of messaging between core group members.

- High speed First In First Out (FIFO) messaging between members of the lightweight component groups.
- High speed view synchronous messaging.
- Java Virtual Machine (JVM) heartbeats or pulses, which are used to indicate that the associated application server is still healthy.

Transport protocol for a high availability manager

The high availability manager network components can be built on either a channel framework, unicast or multicast transport protocol. Channel framework is the default protocol, but there are options and features available with multicast that might be a better match for your application server environment. Multicast emulation can be used with any of these protocols.

The following network protocol configurations can be used with WebSphere Application Server:

Channel framework

Channel framework is the default network protocol configuration for the high availability manager. It provides a common model for connection management, thread usage, channel management, and message access within WebSphere Application Server. It extends the concept of a networking protocol stack, or transport chain, to the WebSphere run time. Each transport chain consists of one or more types of channels, and each channel 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.

The transport chain configuration settings determine which I/O protocols are supported for that chain. Custom channels that support requirements unique to a particular customer or environment can also be added to a transport chain.

Unicast protocol

Unicast protocol is a direct method of sending and receiving messages. JVMs are discovered when a new application server attempts to open connections to application servers in the same core group that are already running. The use of TCP/IP makes this suitable for high speed Wide Area Networks (WANs) and Local Area Networks (LANs). This protocol uses less CPU and memory per connection than multicast protocol, and might be more appropriate for larger core groups.

Multicast protocol

Multicast protocol requires a tuned environment. Heartbeats must be used as the failure detection method with this protocol. In this environment the JVMs cannot be swapped or kept from running. A non responsive JVM is viewed as a failed unit and all work items are taken away from that application server and dispersed among other core group members. The network used by all of the members of the core group must be able to use multicast protocol.

Performance in terms of data rate is about the same for all three protocols. WebSphere Application Server usually performs the best if a channel framework model is used, because this model supports using multiple protocols at the same time. However if you are choosing between a unicast and a multicast protocol, the unicast protocol is usually better because the most common WebSphere Application Server scenario is low fan out or point-to-point messaging. Using the multicast protocol in a low fan out environment forces all the recipients to process the message, even though only a low number of application servers use the message.

Creating a new core group

Before creating a new core group, you must determine which application server processes to add to the core group. For example, if you are creating a new core group because a firewall is used to separate your proxy environment from your application server environment, you might leave the deployment manager, the node agent, and some of the other Application Server processes in the default core group, and move the server processes that exist on the other side of a firewall to the new core group after you create it.

You might want to perform this task if:

- A firewall is used to separate your proxy environment from your application server environment. (Additional core groups are required to provide proper failover support in this topology.)
- The number of open TCP/IP sockets becomes unacceptable. (This could occur if you are using a multicast protocol.)

Every WebSphere process is initially a member of the default core group provided with the WebSphere Application Server product. Using the default core group is sufficient in most configurations.

To create a new core group:

1. In the administrative console, click **Servers > Core groups > Core group settings > New** .
2. In the Name field, specify a unique name for the new core group. This field can only be edited when you create the new core group. Make sure that the name is meaningful and consistent with the names of the other core groups in the cell. It is helpful if the name conveys why the application servers are being moved to this core group. For example, if your company's human resources applications are installed on the application servers that are moved to this core group, you might include HR as part of the core group name.
3. Add a description of this core group that helps other administrators understand the purpose of this core group.
4. In the Number of coordinators field, specify the number of active application servers you want serving as coordinators for this core group. Each core group coordinator can manage up to 10,000 core group components. Specify a value for this field based on the anticipated number of core group components.
5. Select the type of transport that the application servers contained in this core group are using.
   a. If you select **CHANNEL_FRAMEWORK**, specify the name of an already defined transport chain in the Channel chain name field. Any value specified for the Multicast port, Multicast group IP start or Multicast group IP end fields are ignored.
   b. If you select **MULTICAST**, you must enter a port number in the Multicast port field, enter the first IP address in the range of IP addresses that can be used in the Multicast group IP start field, and enter the last IP address in the range of IP addresses that can be used in the field.
   c. If you select **UNICAST**, any value specified for the Channel chain name, Multicast port, Multicast group IP start or Multicast group IP end fields are ignored.
6. Click **Apply**.
7. On the administrative console panel click **Core groups >** *core_group_name* **> Policies > New** , and then select the policy that you want to associate with a high availability group in this core group. The high availability groups that are part of the new core group determine the policies that are required for this core group. New policies do not have to be defined if:
   - The processes contained in the new core group do not contain any high availability groups.
   - The new core group only contains processes, such as the service integration bus, for which default policies are provided as part of the high availability manager function.

   If you need to define a new policy, the policy options are:
   - All active policy: Under this policy, all of the group members are activated.
   - M of N policy: Under this policy, *M* group members are activated. The number represented by *M* is defined as part of the policy details.
   - No operation policy: Under this policy, no group members are activated.
   - One of N policy: Under this policy, only one group member is activated.
   - Static policy: Under this policy, the active members of a group are statically assigned.

   Multiple policies can be defined if different high availability groups require different policies. However, only one policy can be associated with a given high availability group. See for more information about these policies.

Note: If you are setting up a policy for a transaction manager, you must select One of N as your policy type because a transaction manager requires that only one server can have access to a failed server's recovery log at any point in time.

8. Click **Next**.
9. Specify a name for the policy that is unique within the scope of the core group.
10. Change the value specified in the Is alive timer field if the default value is too long or too short a time period. This value, specified in seconds, determines how frequently the high availability manager checks the health of the core group members. Valid values are any integer between -1 and 600, inclusive. If -1 is specified, the timer is disabled. If 0 (zero) is specified, the default value of 120 seconds is used.
11. Select **Quorum** if you want to enable quorum checking for the core group. Quorum is a mechanism that can be used to protect resources shared across members of the group in the event of a failure.

    CAUTION:
    Quorum is an advanced hardware function and should not be enabled unless you thoroughly understand how to properly use this function. If not used properly, this function can cause data corruption.

    The Quorum setting in the policy will only have an effect if the following items are true:

    - The group members are also cluster members.
    - **GroupName.WAS_CLUSTER=***clustername* must be specified as a property in the group name of any high availability group matching this policy.

    When enabled, any group that is using this policy does not achieve quorum until a majority of the members are running. For example, if there are `n` members in the group, (n/2) + 1 servers must be online in order to achieve quorum. No group members are activated until quorum has been achieved.

    The quorum mechanism is designed to work in conjunction with a hardware control facility that allows application servers to be shut down if a failure causes the group to be partitioned.
12. Click **Apply**, and then select **Match criteria** .
13. On the next panel, click **New** and then specify the match criterion for this policy. A match criterion is a set of one or more name=value pairs of data that can be matched to attributes contained in the name of a high availability group.
    a. In the Name field, enter the name of one of the properties contained in a high availability group's name that you want to use to associate this policy with that high availability group.
    b. In the Value field, enter the value of this property that, along with the property name, forms the name=value pair.
    c. Optional: Add a description of this match criteria in the Description field.
    d. Click **Ok**.
    e. If you need to specify additional name=value pairs for your match criterion, repeat this step. Each name=value pair must be entered separately. Repeat this step until you have specified all of the name=value pairs required for this match criterion.
14. Click **Save**, select **Synchronize changes with nodes** and then click **Save** again to save your changes.

You are now ready to add members to this new core group.

Core groups

A core group is a set of application servers that can be divided up into various high availability groups. It is a statically defined component of the WebSphere Application Server high availability manager function that monitors the application server environment and provides peer to peer failover of application server components.

A core group can contain one or more high availability groups. However, a high availability group must be totally contained within a single core group. Any component, such as the service integration bus component of IBM service integration technologies or the WebSphere Application Server transaction manager, can create a high availability group for that component's use. For example, the service integration bus might need a high availability group to support its messaging engines, and the transaction manager component might need a high availability group to support its transaction logs.

A cell must have at least one core group. The WebSphere Application Server creates a default core group, called DefaultCoreGroup, for each cell. All the server processes and Java virtual machines (JVMs), including node agents on all managed nodes, the deployment manager, and application servers residing within a cell are initially members of this default core group.

When properly configured, the default core group is sufficient for establishing a high availability environment. However, certain topologies require the use of multiple core groups. For example, if a firewall is used to separate the proxy environment from the server environment, an additional core group is required in order to provide proper failover support. For this particular type of environment, application servers outside of the firewall are members of a separate core group from the application servers that are inside of the firewall.

The core group contains a bridge service, which supports cluster services that span multiple core groups. Core groups are connected by access point groups. A core group access point defines a set of bridge interfaces that resolve to IP addresses and ports. It is through this set of bridge interfaces that the core group bridge provides access to a core group.

If you create additional core groups, when you move core group members to the new core groups, remember that:

- Each server process and Java virtual machine (JVM), including node agents on all managed nodes, the deployment manager, and application servers within a cell can only be a member of one core group. Core groups cannot overlap each other.
- If a cluster is defined for the cell, all of the cluster members must belong to the same core group.

Network communication between all the members of a core group is essential. The network environment must consist of a fast local area network (LAN) with full Internet Protocol (IP) visibility and bidirectional communication between all core group members. IP visibility means that each member is entirely receptive to the communications of any other core group member.

A core group consists of the following elements:

Coordinator
The coordinator is the elected, or default, high availability manager. The coordinator is responsible for tracking all of the members of a core group when members leave, join, or fail. The coordinator is not a single point of failure. In the event of a failure involving the coordinator, the preferred coordinator, or a default, picks up the high availability manager work, including the management of the core group.

Core group members
Any application server that is created within a cell that is defined as part of a high availability environment is automatically designated as a core group member.

Messaging subsystem
A messaging subsystem is a subsystem, such as the service integration bus component of IBM service integration technologies, that provides high-speed connections between core group members. This subsystem enables all core group members to quickly and effectively communicate with each other.

Core group bridge service
The core group bridge service is only utilized in high availability environments that contain multiple core groups. Use the bridge service to provide quick and effective communication between core groups.

Policy A policy is used to control which members of a high availability group are activated. Even though a policy is defined at the core group level, it does not apply to the core group. The same policy can be used by several different high availability groups but all of the high availability groups to which it applies must be part of the same core group. A policy is established for a high availability group when that group is created. The following policies can be specified for a high availability group:

- All active policy: Under this policy, all of the group members are activated.
- M of N policy: Under this policy, *M* group members are activated. The number represented by *M* is defined as part of the policy details.
- No operation policy: Under this policy, no group members are activated.
- One of N policy: Under this policy, only one group member is activated.
- Static policy: Under this policy, only specified group members are activated.

In a run-time server environment each core group functions as an independent unit. A Java Virtual Machine (JVM) that is contained within a cell can be a member of one core group only. This JVM can be a node agent, an application server or a deployment manager. However, even though the deployment manager can belong to one core group only, it is still responsible for configuring all of the application servers within a cell, even if multiple core groups are defined for that cell.

Every controller within a core group contains an instance of the high availability MBean that can be used for various runtime operations within that core group. Java Management Extensions (JMX) connectors can be used to connect and query for this MBEAN in the following ways:

- JMX can connect and query on the deployment manager controller just like the administrative console and most administrative clients do. This approach is

easier because you only have to specify the deployment manager's host name and port numbers. However, if the deployment manager is down JMX will not be able provide access to the MBean.
- JMX can connect and query for the MBean on any controller or across multiple controllers within the core group. This approach is more reliable because even if the deployment manager is down, the JMX connectors can still access the MBean as long as one of the controllers that contains the MBean is still running, .

Because the high availability MBean instances are scoped to a core group, if you have multiple core groups configured within a cell, you must refine a query for this MBean to the servers that are members of the core group you want to manage.

The coordinator retains and tracks membership and state changes for a core group. To work efficiently, the coordinator must have enough memory to contain the group and state information for the online members of the core group. Configuring multiple coordinators enables this task to be shared equally among a set of online coordinators.

Using the administrative console, you can designate specific servers as preferred coordinator servers. High availability manager coordinators run in these servers. If no preferred coordinator servers are specified, the high availability manager selects them. To minimize churn on the core group, it is recommended that you designate specific servers as preferred coordinator servers and limit how often you restart these servers. The heap sizes for the JVMs and the relative amount of memory that the JVMs use to host coordinator services need to be tuned to ensure that enough memory is available to hold the group and state information for the core group.

While core group members are statically configured, high availability group members are dynamically selected and only exist as run-time entities. The WebSphere Application Server runtime uses a policy matching program to determine the policy that should be used for each high availability group. Each policy includes a match criterion that consists of a set of name=value pairs. The policy matching program matches these name=value pairs with attributes contained in the name of a high availability group. When a match is made, the policy is associated with that high availability group.

High availability groups

High availability groups are dynamically created components of a core group. They cannot be configured directly but are directly affected by static data, such as policy configurations, which are specified at the core group level.

A high availability group cannot extend beyond the boundaries of a core group. However, members of a high availability group can also be members of other high availability groups as long as all of these high availability groups are defined within the same core group.

Any WebSphere Application Server component, such as the transaction manager, can create a high availability group for that component to use. The component code must specify the attributes that are used to create the name of the high availability group for that component. For example, to establish a high availability group for the transaction manager:
- Code included in the transaction manager component code specifies the attribute `type=WAS_TRANSACTIONS` as part of the name of the high availability group associated with this component.

- The high availability manager function includes the default policy Clustered TM Policy that includes `type=WAS_TRANSACTIONS` as part of its match criteria.

Whenever transaction manager code creates a high availability group, the high availability manager matches the match criteria of the Clustered TM Policy to the high availability group member name. In this example, the string `type=WAS_TRANSACTIONS` included in the high availability group name is matched to the same string in the policy match criteria for the Clustered TM Policy. This match associates the Clustered TM Policy with the high availability group the transaction manager component creates.

After a policy is established for a high availability group, you can change some of the policy attributes, such as quorum, fail back, and preferred servers. However, you can not change the policy type. If you need to change the policy type, you must create a new policy and then use the match criteria to associate it with the appropriate group.

If you want to use the same match criteria, you must delete the old policy before defining the new policy. You cannot use the same match criteria for two different policies.

Important: If the old policy is one of the default policies that IBM provides, it is recommended that you do not delete the old policy. Instead, you should use a different match criteria for your new policy. This new match criteria should include more matches with the attributes contained in high availability group's name than the default policy uses for its match criterion. The policy with the greatest number of matches is the one that is used. Not deleting the IBM provided policy enables you to revert back to that policy if a problem occurs when you use your newly created policy.

Before changing the policy type for a high availability group you must fully understand how the application server processes that are contained in that high availability group are configured and how they will be affected by the policy change. You must also verify that the component that uses that particular high availability group supports the new policy type. For example, if the high availability group for the service integration bus uses a One of N policy, because it only wants one server to be active at any given point in time, and you change the policy associated with that group to All Active, the service integration bus high availability support no longer functions properly and data corruption might occur.

Core group collection

A core group is a component of the high availability manager function. A default core group, called **DefaultCoreGroup** is created for each cell in the WebSphere Application Server environment. A core group can contain standalone servers, cluster members, node agents and the deployment manager. A core group must contain at least one node agent or the deployment manager.

To view this administrative console page, click **Servers > Core Groups > Core group settings** .

Name	Specifies the name of the core group. Click the core group name to edit the settings for that core group. This field is read-only.
Description	Describes the core group. This field is read-only.

Connected core groups	Specifies the core groups that are connected to this core group by access points. This field is read-only.

Click **New** to define a new core group. After a core group is defined, several fields become read-only. To change those fields, delete and redefine the core group.

Click **Delete** to delete a core group. A core group must be empty before it can be deleted.

Core group settings

Use this page to create a core group or to edit an existing core group. A core group is a component of the high availability manager function. It can contain standalone servers, cluster members, node agents and the deployment manager. A core group must contain at least one node agent or the deployment manager.

Before you create a core group you must understand the relationship of core groups in a high availability environment and know how you intend to use each core group.

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or *Select an existing core group for editing*.

On the **Configuration** tab, you can edit fields. On the **Runtime** tab, you can look at read-only information.

After you specify your core group settings, click **Apply** before defining additional properties or setting up a core group bridge.

Extended information about the core group fields:

Configuration Tab::

Name: This field can only be edited when you create new core groups.

If you are defining a new core group, specify a name that is unique name among the existing core groups. It is helpful to other WebSphere Application Server administrators if the name helps define the use of this core group and if it is consistent with the names of the other core groups in the cell.

This field can contain alpha and numeric characters. The following characters cannot be used in this field:

```
# \ / , : ; " * ? < > | = + & % '
```

Also, the name cannot begin with a period (.) or a blank space. A blank space does not generate an error. However, leading and trailing blank spaces are automatically deleted.

For example, `DefaultCoreGroup` is the name of the core group that contains the deployment manager server process.

Description: Use this optional field to include a description of the core group. In environments where there are multiple system administrators this field can help these administrators understand the overall organization of the core groups. The

supported length of this field is quite large. However, long descriptions take time to load and can cause a delay when displaying the page.

Example: ″Default Core Group. The default core group cannot be deleted.″ is the description of the DefaultCoreGroup.

Number of coordinators: The coordinator is a component in each server in a core group that provides the service functionality of the high availability manager. The coordinator for a core group determines membership and communicates state and status to the other members of the core group.

The default value is one coordinator, although multiple coordinators are advisable for large core groups. All of the group data must fit in the memory of the allocated coordinators. One coordinator can run out of memory in a system with a large core group, which can cause the system to work improperly.

Transport type: The transport type is a required field that specifies the type of network communication your core group uses to communicate to members and to other core groups. The following transport options are available:

CHANNEL_FRAMEWORK
CHANNEL_FRAMEWORK is the default transport type. It uses the channel framework topology to incorporate port reusability and shared port technology into the communication system.

UNICAST
UNICAST is a targeted network model that focuses on a direct recipient for communication. This type of communication is most suitable when the intended message is sent to a specific set of recipients.

MULTICAST
MULTICAST consists of a broadcast network model. This model broadcasts communication across the defined network, depending upon the values that are provided for the multicast settings. Multicast settings are suitable when there are many recipients for the intended message; otherwise broadcast communication tends to overload the network with traffic, and can impact performance goals.

Channel chain name: Specifies the name of the channel chain if CHANNEL_FRAMEWORK is selected for transport type.

MULTICAST settings: Specifies the following settings if a multicast transport type is used:

- **Multicast port**

  The port setting tells the coordinator where to scan for transmissions. When setting this value, verify that you are specifying a port that is not used by another network communication device. Setting a port value that has conflicts causes problems with your high availability manager infrastructure.
- **Multicast group IP start**

  Specify the starting Internet Protocol (IP) address of the intended communication area.
- **Multicast group IP end**

  Specify the ending IP address of the intended communication area. Plan the network to accommodate scalability.

Additional properties:

Core group servers
Use to display the server processes that belong to the core group. Server processes include the deployment manager, node agents, application servers, and cluster members. You can use the panel that displays to move server processes to a different core group.

Custom properties
Use to define custom properties to be used for configuration purposes.

Policies
Use to define the policies that the coordinator servers use to select the active members of a core group. You can select from existing policies, or create new custom policies.

Preferred Coordinator Servers
Use to designate core group servers as preferred coordinator servers.

Related topics:

Core group bridge
Click on to specify core group bridge communication settings between core groups.

Runtime Tab::

Group name properties:

Specify one or more name=value pairs as the match criterion for a high availability group. If you specify more than one name=value pair, use a comma to separate the pairs. You can specify an asterisk (*) to obtain the selected information for all of the high availability groups within this core group.

When a WebSphere Application Server component creates a new high availability group, it establishes a map of that group's properties as the group name. This map is used to uniquely identify that high availability group.

After you specify a match criterion or an asterisk:

- Select **Calculate** to determine how many high availability groups have names that match this match criterion.
- Select **Show groups** to view a list of the currently running high availability groups that match this match criterion. For each group, this list indicates:
  - Its high availability group name
  - Whether or not quorum has been enabled
  - The policy that is associated with the high availability group. If more than one policy is listed for a high availability group, you must change the match criterion for one or more of your policies so that only one policy is associated with this high availability group.
  - Its status (either the OK icon or the Error icon). If only one policy is listed in the Policy column, the OK icon is displayed in the Status column. If more than one policy is listed Policy column, the Error icon is displayed in the Status column.
- Select **Show severs** to view a list of servers which are hosting active members of the high availability groups that match the specified Group name properties. For each server, this list indicates:
  - The names of the servers which are hosting the active high availability group members.
  - The name of the node on which these servers resides.

– The version of the WebSphere Application Server product on which these servers are running.
– The number of high availability group members that are currently active on these servers.

Example: Suppose the following high availability groups are defined for a core group:

- Component A uses the following properties for its group name: [name=compA, policy=oneofN, owner=smith]
- Component B uses the following properties for its group name: [name=compB, policy=MofN, owner=smith]
- Component C uses the following properties for its group name: [name=compC, policy=oneofN, owner=smith]

If you specify `policy=oneofN` in the **Group name properties** field and then select **Show groups**, the groups for components A and C are listed.

If you specify `owner=smith` in the **Group name properties** field and then select **Show groups**, the groups for components A, B and C are listed.

If you specify all of component C's name properties in the **Group name properties** field:

```
name=compC,policy=oneofN,owner=smith
```

Then select **Show groups**, only the group for component C is listed. Note that the properties are separated by commas. There are no blank spaces.

Changing a core group's configuration

You might want to perform this task if you need to:

- Change the number of coordinators that are to be active.
- Change the transport type for the core group members.
- Set up some custom properties for your specific environment.
- Add or remove application servers from the list of preferred coordinator servers.

To change a core group's configuration:

1. In the administrative console, click **Servers > Core groups > Core group settings >** *core_group_name*.
2. If you want to change the number of coordinators that must be active for this core group, specify a new value in the **Number of coordinators** field.
3. If you want to change the transport specification for this core group, in the **Transport type** pull-down, select the type of transport the application servers in this core group are going to use.
   a. If you select **CHANNEL_FRAMEWORK**, specify the name of the channel chain in the **Channel chain name** field.
   b. If you select **MULTICAST**, enter the multicast port number in the **Multicast port** field, enter the first IP address in the range of IP addresses that can be used in the **Multicast group IP start** field, and enter the last IP address in the range of IP addresses that can be used in the **Multicast group IP end** field.

4. If you need to set up a custom property for your specific environment, click **Custom properties > New** and then specify the name and value pair of your custom property.
5. If you need to change the list of preferred coordinator servers, click **Preferred coordinator servers** and then add or delete the appropriate application servers from the list of preferred coordinator servers.
6. Click **Save**, select **Synchronize changes with Nodes** and then click **Save** again to save your changes.

Core group and core group bridge custom properties

Use these custom properties for advanced configurations with core groups and core groups that communicate with the core group bridge.

FW_PASSIVE_MEMBER:

Use this property in a core group bridge configuration when there is a firewall between the core groups and the secure side of the firewall is configured to listen only.

Set the FW_PASSIVE_MEMBER custom property to make the bridge interfaces that are in the core group access point passive. Set the value on the core group access point that is on the secure side of the firewall so that the core group bridge interfaces listen for connections from the unsecured side of the firewall but do not initiate any connections. The servers on the secure side of the firewall are passive. The custom property should correspond to your defined firewall rules that allow connections from the unsecured region to the secure region only.

To configure this custom property, click **Servers > Core groups > Core group bridge settings > Access point groups >***access_point_group_name* **> Core group access points >** *core_group_access_point_name* **> Show detail > Custom properties > New** in the administrative console.

You also must set this custom property in any peer access points that refer to the core group access points that you configure with this custom property.

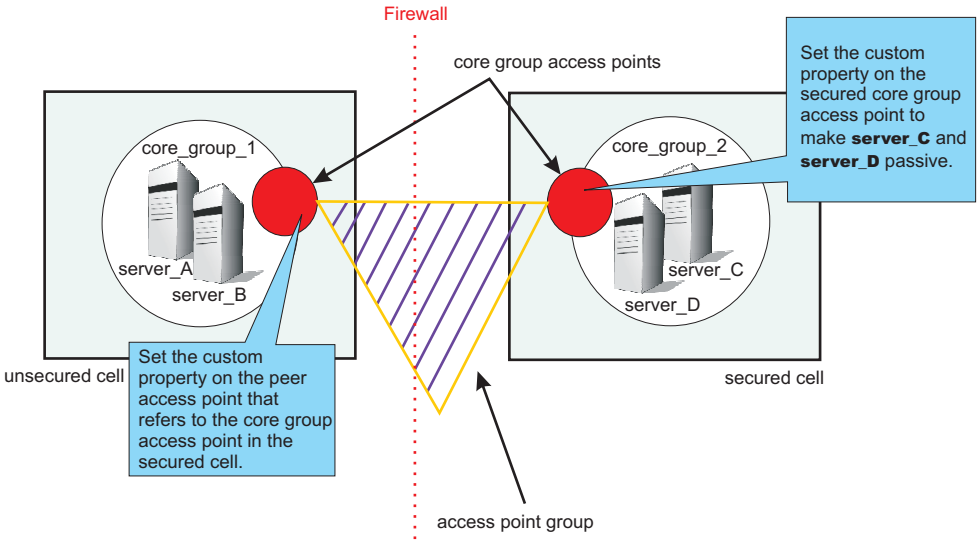


Figure 7. Configuring the FW_PASSIVE_MEMBER custom property

For example, *server_A* and *server_B* are configured in *core_group_1*. *Server_C* and *server_D* are configured in *core_group_2*. *Core_group_2* is behind a firewall that is configured to listen only through the firewall. *Core_group_1* is on the unsecured side of the firewall. *Core_group_1* and *core_group_2* can communicate with each other through an access point group. To configure *server_C* and *server_D* to be passive, perform the following steps:

1. In the administrative console for the cell that contains *core_group_2*, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points >** *core_group_access_point_name* **> Show detail > Custom properties >New** .
2. Add the FW_PASSIVE_MEMBER custom property. Enter any value to enable the property.
3. In the administrative console for the cell that contains *core_group_1*, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points >** *peer_access_point_name* **> Show detail > Custom properties > New**. The peer access point you select should correspond to the core group access point for *core_group_2*.
4. Add the FW_PASSIVE_MEMBER custom property. Enter any value to enable the property.

By configuring the FW_PASSIVE_MEMBER custom property, you configured the servers on the secured side of the firewall, *server_C* and *server_D*, to be passive. These servers listen for connections from the other side of the firewall but do not initiate any connections to the unsecured side of the firewall.

IBM_CS_LS_DATASTACK_MEG:

Use this custom property to eliminate a condition that is reported by a message that is displayed repeatedly in your `SystemOut.log` file.

You might see a message similar to the following message in the SystemOut.log file multiple times:

```
[9/24/04 13:28:19:497 CDT] 00000013 VSync         W
DCSV2005W: DCS Stack DefaultAccessPointGroup.P at Member 172.16.1.86:9353:
The amount of memory available for synchronization is low. The configured memory
size is 418816 bytes. Currently used memory size is 420307 bytes.
```

If the member IP address is in the format of a dotted decimal IP address and port, you can eliminate these messages by increasing the amount of memory that is allocated to the core stack that is used for core group communication. Increase the value of this property until you no longer see the message in your `SystemOut.log` file. Because the memory is dynamically allocated, setting a larger stack size than you need does not cause memory problems.

Set the custom property on the bridge interface that contains the particular member that is in the messages. You can also set the custom property on the access point group or the core group access point. If you set the value on the access point group or core group access point, all the bridge interfaces that are in the particular group are affected. If you set the value on an individual bridge interface and an access point group or core group access point, the value that is set for the bridge interface is used. If the value is set on both an access point group and a core group access point, the value that is set for the core group access point is used.

To configure this custom property, complete the following steps:

1. Set the custom property in the administrative console.

- To set the custom property on a bridge interface, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points >** *core_group_access_point_name* **> Show detail > Bridge interfaces >***bridge_interface_name* **> Custom properties > New**.
- To set the custom property on a core group access point, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points >** *core_group_access_point_name* **> Show detail > Custom properties > New**.
- To set the custom property on an access point group, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Custom properties > New**.

2. Add the IBM_CS_LS_DATASTACK_MEG custom property. Enter a value that is greater than the default value of 5 megabytes.

Units	megabytes
Default	5

Setting up IP addresses for high availability manger communications

You might want to perform this task if you are installing the Application Server on a machine on which multiple network interface cards (NICs) are installed.

If your system has multiple NICs installed, you must select a preferred IP address, or a range of IP addresses in the case of a multicast configuration, which maps to the NIC you want the high availability manager to use for communication within a core group.

1. If the core group transport is configured as either channel framework (the default) or unicast, specify the preferred IP address for the `DCS_UNICAST_ADRESS` port. To specify the preferred IP address for the `DCS_UNICAST_ADRESS` port., go to the configuration for each process that is running on the node in question, and change the Host field associated with the `DCS_UNICAST_ADRESS` port. For a given physical machine this action probably involves the node agent process as well as the application server processes. If the machine is running the deployment manager process that also needs to be updated. Basically all processes running on the physical machine need to be told which IP address to use for communication.

   To specify a preferred IP address for a node agent process:

   a. In the administrative console, click **System administration > Node agents**
   b. Select the desired node.
   c. Under **Additional properties** , click on **Ports** to bring up the list of ports that node agent process uses.
   d. Click on the port called `DCS_UNICAST_ADDRESS`.
   e. Enter the preferred IP address in the Host field and then click **Apply** to save your changes.

   Repeat these steps for the application server processes and the deployment manager process, if one is running on this machine:

   - For the application server processes, in the administrative console, click **Servers > Application Servers >** *server_name* to bring up the configuration page for the desired server process.

- For a deployment manager process, in the administrative console, click **System administration > Deployment manager**.

2. If the core group transport is configured as multicast, use core group properties to specify the multicast IP address range associated with the NIC. If the core group transport is configured as multicast, then the multicast IP address range associated with the NIC that the high availability manger should use for communication must be configured using core group properties instead of on a per process basis.
   a. In the administrative console, click **Servers > Core groups > Core group settings**
   b. Select the desired core group.
   c. Specify the first IP address in the range associated with the NIC the high availability manager should use in the Multicast group IP start field
   d. Specify the last IP address in the range associated with that NIC in the Multicast group IP end field.
   e. Click **Apply** to save your changes.

The high availability manager uses the specified IP address or address range for communication within the core group.

Changing the configuration of a high availability group

High availability groups are dynamically created components of a core group. They cannot be configured directly but are directly affected by static data, such as policy configurations, which are specified at the core group level.

A high availability group has been established for a core group.

You might want to perform this task if you want to remove a high availability group from a cell.

You can use the administrative console to activate or disable all of the members of a high availability group.

1. In the administrative console, click **Servers > Core groups > Core group settings**.
2. Click on the core group whose high availability groups you want to view.
3. Click on the Runtime tab
4. Specify a match criterion in the Group name properties field that matches the high availability group you want to enable or disable. (You can specify an asterisk (*) to get a list of all the high availability groups that are part of this core group.)
5. Click show groups.
6. The Status column of the **High availability groups** panel shows the state of the high availability groups in this core group that match the specified match criterion.
7. In the Select column, indicate the high availability group whose state you want to change and then click **Enable** or **Disable**.

   Enable activates members of the high availability group that are currently in the disabled state. If all of the members are already active, no action is taken.

   Disable disables all of the high availability group members that are active.

It can take several minutes for this change to take affect.

High availability group servers collection

Use this page to determine how many high availability group members are active on a particular application server.

To view this administrative console page, click **Servers > Core groups > Core group settings >** *core_group*. Click on the **Runtime** tab and specify group name properties for a high availability group. (You can specify an asterisk (*) to get a list of the servers that are hosting active members for all the high availability groups in this core group.) Then select **Show severs**.

Server	Specifies the name of a server on which there are active high availability group members. This field is read-only.
Node	Specifies the node on which each server is running. This field is read-only.
Version	Specifies the version of the WebSphere Application product on which each node is running. This field is read-only.
Active members	Specifies the number of high availability group members that are currently active on that server. This field is read-only.

High availability group settings

Use this page to manage the state of a high availability groups.

To view this administrative console page, click **Servers > Core groups > Core group settings >***core_group*. Click on the **Runtime** tab. In the Group name properties field, specify a match criterion for a specific high availability group, or specify an asterisk (*) to get a list of all the high availability groups that are part of this core group. (A match criterion is one or more name=value pairs that match attributes contained in a high availability group's name.) Then click **Show groups**.

High availability group	Specifies the names of the high availability groups. The name of a high availability group is a set of name=value pairs or attributes, separated with commas. For example, name=productiongroup,policy=abc,ibm=websphere could be the name of a high availability group. This field is read-only.
Quorum	Specifies if quorum is enabled for each high availability group. This field is read-only.

Policy

Lists the policies that have match criteria that matches properties contained in the name of that high availability group. There should only be one policy listed for a high availability group. However, if multiple policies have match criteria that equally match properties in a high availability group's name, all of the policies with matching criteria are listed, and the ERROR icon appears in the status column.

For example, if you have a high availability group named name=productiongroup,policy=abc,ibm=websphere, and MyPolicy1 has the match criteria name=productiongroup, and MyPolicy2 has the match criteria policy=abc, both MyPolicy1 and MyPolicy2 are considered matching policies and are listed in the Policy column.

This field is read-only.

Status

Indicates, with icons, whether or not only one policy is associated with a high availability group. If the OK icon displays in this column, a single policy is associated with that high availability group. If the ERROR icon displays in this column, multiple policies are associated with that group.

If the ERROR icon displays for a high availability group, you must adjust the match criteria for one or more of the policies listed in the Policy column for that group so that the correct policy is the only one associated with that high availability group.

The match criteria for multiple policies can match some of the same properties in a group's name as long as one policy has a match criteria that matches more of the properties in that group's name than the match criteria of any of the other policies. For example, if you have a high availability group with a name that consists of the following name and value pairs:

```
name=productiongroup,policy=abc,ibm=websphere
```

and MyPolicy1 has the match criteria `name=productiongroup` and MyPolicy2 has the match criteria `name=productiongroup,ibm=websphere` MyPolicy2 is considered the matching policy because it has more match criteria that matched the properties contained in the high availability group's name.

This field is read-only.

Use the **Disable** button if you want to prevent all of the members of a high availability group from being activated. One of the few times you might want to use this button is if you are planning to remove or delete the server on which this group is running.

Use the **Enable** button to enable all of the members of a high availability group that were previously disabled. These members can then be activated according to the policy associated with that group.

High availability group members

Use this page to manage the state of the individual members of a high availability group. This page lists the current members of the selected high availability group.

To view this administrative console page, click **Servers > Core groups > Core group settings >***core_group*. Click on the **Runtime** tab. In the Group name properties field, specify a match criterion for a specific high availability group, or specify an asterisk (*) to get a list of all the high availability groups that are part of this core group. (A match criterion is one or more name=value pairs that match attributes contained in a high availability group's name.) Then click **Show groups** and select one of the high availability groups listed.

Name	Specifies the name of a high availability group member.
Node	Specifies the node on which each high availability group member is running.
Version	Specifies the version of the WebSphere Application Server product on which each node is running.
Status	Indicates the state of the high availability group members. High availability group members are either idle, activated, or disabled. The usual states are idle or activated. One of the few times you might want to disable a member is if it is running on a server that you plan to remove or delete. • If a group member is idle, it cannot be assigned any work. • If a group member is activated, it can be assigned work. • If a group member is disabled, it must be enabled before it can be activated.

Only use the **Disable** buttons if you want to prevent a group member from being activated.

Use the **Enable** button to enable a group member that was previously disabled.

Use the **Activate** button to activate an idle group member.

Use the **Deactivate** button to make a group member idle.

Creating a policy for a high availability group

Before creating a new policy, you should understand the following requirements for matching a high availability group to a policy:

- A single high availability group cannot match to more than one policy. If the same match criteria is used for multiple policies, the high availability groups with names that match this criteria do not activate when the server is started and an error message, indicating that multiple policies match this group, is logged.
- The same policy can be used for multiple high availability groups.
- A hierarchy is used when matching a high availability group to a policy. If multiple properties are used for the match criteria, the policy that has the most matches with the high availability group name is the policy that is used.

You might want to perform this task if you have a high availability group that requires a policy that is different from the policy governing other high availability groups in the same core group. For example:

- A One of N policy is in effect for all of your high availability groups
- There is a high availability group within the core group that needs to have 5 members active at all times.

You can create a policy with M of N specified as the policy type and 5 specified for the Number of active members, and specify a match criterion that associates this policy with the high availability group that requires this policy.

Important:

- Do not change any of the default policies that are shipped with the WebSphere Application Server product. If you need to use different polices, create new policies and specify a match criterion that matches multiple attributes contained in the name of the high availability group. A policy with a greater number of match criterion matches overrides the IBM provided default policies. The IBM provided policies are still available for your use if you encounter a problem with the policies you create.
- Before changing the policy type for a high availability group, make sure you fully understand how the application server processes that are contained in that high availability group are configured and how they will be affected by the policy change. You must also verify that the component that uses that particular high availability group supports the new policy type. For example, if the high availability group for the service integration bus is using a one of N policy, because it only wants one server to be active at any given point in time, and you change the policy associated with that group to All Active, the service integration bus high availability support no longer functions properly and data corruption might occur.

To create a new policy:

1. In the administrative console, click **Servers > Core groups > Core group settings** *core_group_name* **> Policies > New**
2. Select the new policy you want in effect for a specific high availability group. If you need to define a new policy, the policy options are:
   - All active policy: Under this policy, all of the group members are activated.
   - M of N policy: Under this policy, *M* group members are activated. The number represented by *M* is defined as part of the policy details.
   - No operation policy: Under this policy, no group members are activated.
   - One of N policy: Under this policy, only one group member is activated.

- Static policy: Under this policy, the active members of a group are statically assigned.

Multiple policies can be defined if different high availability groups require different policies. However, only one policy can be associated with a given high availability group. See for more information about these policies.

3. Click **Next**.
4. Specify a name for the policy in the **Name** field. The name must be unique within the scope of the core group. The name should be meaningful to other administrators. For example, if you are setting up a new policy for the service integration bus, you might name the policy `My Service Integration Bus Policy`.
5. Specify a value for the **Is alive timer** field if the default value is too long or too short a time period. This value determines how frequently the high availability manager checks the health of the high availability group members. The default value is 0 seconds.
   - If you specify -1 the Is alive timer is disabled.
   - If 0 (zero) is specified, the value specified for the Is alive timer at the core group services level is used for high availability groups associated with this policy.
   - If an integer between 1 and 2147483647, inclusive, is specified, this value is used for high availability groups associated with this policy.
6. Select **Quorum** if you want to enable quorum checking. When selected, quorum checking is enabled for a high availability group governed by this policy. Quorum is a mechanism that can be used to protect resources shared across members of the group in the event of a failure.

   CAUTION:
   Quorum is an advanced hardware function and should not be enabled unless you thoroughly understand how to properly use this function. If not used properly, this function can cause data corruption.

   The Quorum setting in the policy will only have an effect if the following items are true:
   - The group members are also cluster members.
   - **GroupName.WAS_CLUSTER=***cluster_name* must be specified as one of the name=value pairs contained in the dynamically generated name of a high availability group to which this policy applies. (A component that is using the high availability group function must include the name of its high availability group as part of its component code.)

   When enabled, any group using this policy will not achieve quorum until a majority of the members are running. For example, if there are `n` members in the group, (n/2) + 1 servers must be online in order to achieve quorum. No group members will be activated until quorum has been achieved.

   The quorum mechanism is designed to work in conjunction with a hardware control facility that allows application servers to be shut down if a failure causes the group to be partitioned.
7. For M of N and One of N policies, select the Fail back option if, when the most preferred server is available, you want it to take back the workload from the servers that did the failover.
8. For M of N and One of N policies, select the Preferred servers only option if you only want group members activated on servers included in the list of preferred servers for the group. If you select this option, you must set up a list of preferred servers after you click **OK**. A description of how to set up this list is provided in a later optional step.

9. For an M of N policy, specify in the Number of active members field the number of group members that you want to be activated.
10. Click **Apply** and then select **Match criteria** .
11. On the next panel, click **New** and then specify one of the name and value pairs that you want to include in the match criterion for this policy. The name and value pair must match one of the name=value attributes contained in the name of a high availability group to which you want to associate this policy.

    You, as the WebSphere Application Server administrator, do not have any direct control over the high availability group name. The implementer of the high availability group code decides which properties are used for the group name. You can only control the policy match criterion. To determine the name of a high availability group that is part of a core group, in the administrative console page, click **Servers > Core groups > Core group settings** and select the appropriate core group. Then click on **Runtime Tab** and look under the Group name properties field for the names of the high availability groups associated with that core group.

    You should set the match criterion for a new policy to two or more of the name=value attributes contained in the high availability group's name to ensure that this policy is used instead of the default policy that IBM provides.

    To Specify one of the name and value pairs that you want to include in the match criterion for this policy:

    a. In the Name field, enter the name of one of the name=value attributes contained in the name of the high availability group with which you want to associate this policy. For example, the service integration bus high availability group has the name:

       ```
       IBM_hc=AcceptanceCluster WSAF_SIB_BUS=SSS_Bus WSAF_SIB_MESSAGING_ENGINE=
          AcceptanceCluster.000-SSS_Bus temp_property=WSAF_TEMP type=WSAF_SIB
       ```

       You can specify `IBM_hc`, `WSAF_SIB_BUS`, `WSAF_SIB_MESSAGING_ENGINE`, `temp_property` or `type` in the Name field.

    b. In the Value field, enter the value of the attribute you specified in the Name field. For example, if you specify `IBM_hc` in the Name field, you must specify `AcceptanceCluster` in the Value field for this match criterion to match an attribute included in the name of the service integration bus high availability group.

    c. Optional: Add a description of this match criterion in the Description field. For example, you might specify `First attribute` to indicate that this name=value pair matches the first attribute contained in the group name.

    d. Click **OK**.

    e. Repeat these steps for each additional attribute you want to include as part of your match criterion. For example, you can specify `type` for the Name field and `WSAF_SIB` for the Value field to include this name and value pair as part of your match criterion for this policy.

    Using this example, the following high availability group-to-policy association is established:

High availability group	High availability group name	Policy name	Policy match criterion

service integration bus	IBM_hc=AcceptanceCluster WSAF_SIB_BUS=SSS_Bus WSAF_SIB_MESSAGING _ENGINE=Acceptance Cluster.000-SSS_Bus temp_property=WSAF_TEMP type=WSAF_SIB	My Service Integration Bus Policy	IBM_hc=Acceptance Cluster,type=WSAF_SIB

12. For a Static policy, under **Additional Properties**, select **Static group servers** to select the servers that you want activated. Use **Add** to move core group servers into this list.
13. **Optional:** Select preferred servers for the policy. If you selected the Preferred servers only option, you should set up a list of preferred servers. If you do not set up this list, no group members will be activated.
    a. Under **Additional Properties**, select **Preferred servers**.
    b. Use **Add** to move core group servers into the list of preferred servers. You can use **Move up** and **Move down** to adjust the order of the servers within the list. Make sure that the most preferred server is at the top of the list and the least preferred server is at the bottom.
14. Click **Save**, select **Synchronize changes with Nodes** and then click **Save** again to save your changes.

The new policy goes into affect after it is saved and synchronized. You do not have to stop and restart the affected application servers. In the future, you can change this policy's settings for the Fail back and Preferred servers only options, and create or update the list of preferred servers associated with this policy without stopping and restarting the affected application servers.

Core group policies

Use this page to create or update the policy that determines how many members of a high availability group should be active at a given point in time. A high availability group member is active if it is able to accept work.

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or *existing core group* **> Policies**.

Click **New** to define a new policy. After a policy is defined there are several fields that you can no longer change. To change those fields, delete and redefine the policy. Click **Delete** after selecting a policy to delete the selected policy.

All of the policy fields on this page are read-only. To change the values specified in any of these fields, click on the name of the policy you want to change. When the console page **Core group settings** > *group_name* > **Policies** > *policy name* displays, you can edit the policy properties.

Specifies the name of a defined policy that activates members of a high availability group.

Description
Displays the description of the policy.

Policy type

Specifies the type of policy that is implemented for a high availability group that is associated with this policy.
Restrictions:

1. If you are setting up a policy for a transaction manager, you must select One of N as the policy type.
2. If you are setting up a policy for a service integration bus you must select One of N or Static as the policy type. The default policy that IBM provides for a service integration bus uses a One of N policy type.

Following is a list of valid policy types:

All active
The All active policy indicates that the high availability manager keeps all of the application components that are running on all of the servers in the high availability group active at all times.

M of N
The M of N policy is similar to the One of N policy. However, it enables you to specify the number (M) of high availability group members that you want to keep active if it is possible to do so. The number of active members must be greater than one and less than or equal to the number of servers in the high availability group. If the number of active servers is set to one, this policy is a match for the One of N policy.

No operation
The No operation policy indicates that no high availability group members are activated when the server is started.

One of N
The One of N policy provides singleton failover. It keeps a singleton running on one server. If a failure occurs, the high availability manager starts the singleton on another server. One of N is the default policy for WebSphere Partitioning Facility (WPF). If you select this policy, make sure that:

- All external resources are available to all candidate servers.
- A remote database or a Cloudscape database on a network attached storage (NAS) device is available to all servers in the high availability group to enable messaging between these servers.
- The transaction logs are on a NAS device that is available to all servers that are in the high availability group.

Static The Static policy indicates that you must statically define or configure the active members of the high availability group.

Match criteria
Specifies one or more name=value pairs that are used to associate this policy with a high availability group. These pairs must match attributes that are contained in the name of a high availability group before this policy is associated with that group.

Core group policy settings

Use this page to define a policy for a high availability group. Even though a policy is defined at the core group level, it only applies to a high availability group that is contained within the core group.

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or *existing core group* > **Policies** > *policy_name*.

Name
Specifies the name of the policy. This name must be unique within the scope of a core group.

Policy type

Specifies the policy type that was selected when this policy was created. This is a read-only field. If you want to change the policy type, you must delete this policy and then create it again specifying a different policy type. If this is an IBM provided policy, do not delete it. Instead create a new policy and specify more of the attributes contained in the high availability group's name as the match criterion for this new policy. The policy with the greatest number of matches to attributes in a group's name is the policy that is associated with that group.

Description

You can use this field to provide meaningful information about this policy. For example, the Clustered TM Policy provided with the product has TM One-Of-N Policy in the description field.

Is alive timer

The interval of time at which the high availability manager will check the health of the active group members that are governed by this policy. If a group member has failed, the server on which the group member resides is restarted.

Data type

Integer

- Valid values are -1 to 600 seconds, inclusive.
- If -1 (negative 1) is specified, this function is disabled.
- If 0 (zero) is specified, the frequency at which the high availability manager checks the health of the active group members is determined by the time interval specified at the application server process level.
- If a value larger than 0 (zero) is specified, the high availability manager uses the time interval specified here, instead of the one specified at the application server process level, when determining how frequently it should check the health of the high availability group members using this policy.

Default

0 (zero)

Quorum

When selected, quorum checking is enabled for a group governed by this policy. Quorum is a mechanism that can be used to protect resources that are shared across members of the group in the event of a failure.

CAUTION:
Quorum is an advanced hardware function and should not be enabled unless you thoroughly understand how to properly use this function. If not used properly, this function can cause data corruption.

The Quorum setting in the policy will only have an effect if the following items are true:

- The group members are also cluster members.
- GroupName.WAS_CLUSTER=*clustername* must be specified as a property in the group name of any high availability group matching this policy.

When enabled, any group using this policy will not achieve quorum until a majority of the members are running. For example, if there are `n` members in the group, (n/2) + 1 servers must be online in order to achieve quorum. No group members will be activated until quorum has been achieved.

The quorum mechanism is designed to work in conjunction with a hardware control facility that allows application servers to be shut down if a failure causes the group to be partitioned.

Fail back

When selected, if a failure occurs, work items assigned to the failing server are moved to the server that is designated as the most preferred server for the group. This field only applies for M of N and One of N policies.

Preferred servers only

When selected, group members are only activated on servers that are on the list of preferred servers for this group. This field only applies for M of N and One of N policies.

Number of active members

This field only applies for the M of N policy. Use this field to specify how many of the high availability group members are to be activated.

The options available under **Additional Properties** are determined by the type of policy selected. One or more of the following options are available:

Custom properties	Use this link to specify custom properties for the policy.
Match criteria	Use this link to set up a match criterion for the policy.

Preferred servers	Use this link to set up a list of servers that are given preference when group members are activated.
Static group servers	Use this link to set up a list of the specific servers that are activated.

New core group policy definition

Use this page to create a new policy for a high availability group.

When you create a new policy, the first page that displays lets you select a policy type. To view the administrative console page where you select a policy type, click **Servers > Core groups > Core group settings > New** ***or select an existing core group*** **> Policies > New**.

Select one of the following policy types:

- All active policy: Under this policy, all of the group members are activated.
- M of N policy: Under this policy, *M* group members are activated. The number represented by *M* is defined as part of the policy details.
- No operation policy: Under this policy, no group members are activated.
- One of N policy: Under this policy, only one group member is activated.
- Static policy: Under this policy, only specified group members are activated.

See for more information about these policies and restrictions that apply for specific high availability groups.

After selecting a policy, click **Next** to continue.

Preferred servers

Use this page to define the ordered list of *preferred servers* for the selected policy. The policy gives preference to the servers in this list when activating group members. If there are no preferred servers in the list or none of the servers in the list are active (able to accept work), any available application server will be designated by the coordinator for the high availability group associated with this policy. The coordinator activates and deactivates high availability group members based on the policy that is in effect for that group.

To view this administrative console page, you must be working with a policy that has a policy type of M of N or One of N. If your policy has one of these policy types, click **Servers > Core groups > Core group settings > New >** or *select an existing core group* **> Policies > New >** or *select an existing policy*. Under **Additional Properties**, select **Preferred Servers**.

Use **Add** and **Remove** to move servers into and out of the list of preferred servers. Use **Move up** and **Move down** to adjust the order within the list of preferred servers. Make sure that the most preferred server is at the top of the list and the least preferred server is at the bottom.

Click **OK** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Changes to the preferred servers list take affect as soon as they are saved and synchronized. You do not have to stop and restart the affected application servers.

Preferred coordinator servers

Use this page to define the ordered list of core group servers on which the coordinator servers reside.

To view this administrative console page, click **Servers > Core groups > Core group settings > New >** or *select an existing core group* **> Preferred coordinator servers**.

Use **Add** and **Remove** to move servers into and out of the list of core group servers on which coordinator servers reside. Use **Move up** and **Move down** to adjust the order within this list. Make sure that the most preferred server is at the top of the list and the least preferred server is at the bottom.

Click **OK** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Matching criteria collection

Use this page to view criteria that are defined for a policy.

To view this administrative console page, click **Servers > Core groups > Core group settings > New >** or *select an existing core group* **> Policies > New** or *select an existing policy* **> Match Criteria**.

Click **New** to create a new match criteria for the policy. Click the name of a match criteria to change any of that criterion's properties.

Name	Specifies the name of the match criterion.
Value	Specifies the value that is paired with the name to determine a match in a policy test.
Description	A description of the match criterion.

Match criteria settings

Use this page to view match criterion defined for a policy.

To view this administrative console page, click **Servers > Core groups > Core group settings > New >** or *select an existing core group* **> Policies > New** or *select an existing policy* **> Match Criteria >** *criterion name*.

Use the name, value, and description fields to define a match criterion for the policy. The name and value fields should match a name=value attribute included in the name of a high availability group that you want associated with this policy.

Name	Specify the name of the match criterion. Create meaningful and consistent criterion names to help define their use.
Value	Specify the value that is to be paired with the name to determine a match in a policy test.

Description	Use this optional field to provide more information about the use of this match criterion. This field can be particularly useful in environments with multiple system administrators. For example, if your company has its own service integration bus policy, the description for the match criterion for that policy might be ″Our Local Integration Bus Match Criterion″.

Click **Apply** to define the criterion. Click **Save** and synchronize changes with nodes after defining new criteria.

Static group servers collection

Use this page to designate which servers will be made active for the high availability groups associated with a policy that has Static for its policy type.

This option is available under **Additional Properties** only if Static is selected as the policy type. To view this administrative console page, click **Servers > Core groups > Core group settings >** *existing_core_group* **> Policies >** *static_policy_name* > **Static group servers**.

Use **Add** and **Remove** to move servers into and out of the list of servers that are designated as static group servers.

Click **Apply** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Changing the policy of a high availability group

Important:

- Do not change any of the default policies that are shipped with the WebSphere Application Server product. If you need to use different polices, create new policies and specify a match criterion that matches multiple attributes contained in the name of the high availability group. A policy with a greater number of match criterion matches overrides the IBM provided default policies. The IBM provided policies are still available for your use if you encounter a problem with the policies you create.
- Before changing the policy type for a high availability group, make sure you fully understand how the application server processes that are contained in that high availability group are configured and how they will be affected by the policy change. You must also verify that the component that uses that particular high availability group supports the new policy type. For example, if the high availability group for the service integration bus is using a one of N policy, because it only wants one server to be active at any given point in time, and you change the policy associated with that group to All Active, the service integration bus high availability support no longer functions properly and data corruption might occur.
- Adhere to the following requirements for matching a high availability group to a policy:
  - A single high availability group cannot match to more than one policy. If the same match criteria is used for multiple policies, the

high availability groups with names that match this criteria do not activate when the server is started and an error message, indicating that multiple policies match this group, is logged.

- The same policy can be used for multiple high availability groups.
- A hierarchy is used when matching a high availability group to a policy. If multiple properties are used for the match criteria, the policy that has the most matches with the high availability group name is the policy that is used.

Example: If you have two policies, Policy A and Policy B, and

- The match criteria for policy A is `type=WAS_TRANSACTIONS`.
- The match criteria for policy B is both `type=WAS_TRANSACTIONS` and `IBM_hc=TestCluster`.
- When the transaction manager high availability group is created, it is automatically associated with Policy B because it has the closet match to the group name, which is GN_PS=carbideCell\carbide\ServerA IBM_hc=TestCluster type=WAS_TRANSACTIONS

To change the policy for a high availability group:

1. Determine if there are any policy restrictions for this high availability group. includes information about any policy restrictions for the IBM provided high availability groups.
2. Create a new policy that has an approved policy type and configuration.
3. Use the **Core Group Runtime** panel in the administrative console to see a list of high availability groups exist for this core group and determine which ones need a new policy. While looking at this panel, you should note the group name properties for all of the groups you are changing.
4. Update the match criteria for each new policy so that it matches the original match criteria and at least one other attribute from the name of the high availability group with which you want to associate the new policy. Two policies can not have exactly the same match criteria because only one policy can be associated with a given high availability group.
5. Click **Save**, select **Synchronize changes with Nodes** and then click **Save** again to save your changes.

Adding members to a core group

Two or more core groups must be defined for a single cell.

You might need to perform this task after you perform one of the following tasks:

- You create a new core group and want to move some of the members of another core group into this new core group.
- You create a new application server and want to move it to a core group other than the default core group. When a new application server is created, it is automatically added to the default core group.
- You want to move a node agent from an existing core group that contains multiple node agents to a newly created core group.

When adding new members to a core group, remember that:

- Each server process and Java virtual machine (JVM), including the node agents on all managed nodes, the deployment manager, and application servers within a cell, can only belong to one of the core groups defined for that cell.

- Core groups cannot overlap each other. If a cluster is defined for a cell, all of the cluster members must belong to the same core group.

1. In the administrative console, click **Servers > Core groups > Core group settings >***core_group_name* **> Core group servers**.
2. In the Select column, select the servers that you want to move.
   - If you are populating a new core group, at least one node agent or deployment manager must be moved to the new core group. Because node agents are created during profile creation, they are all initially members of the DefaultCoreGroup core group.
   - Make sure all of the servers you select to move are in stopped state. If you move a server that is not in stop state, depending on the server type (application server, deployment manager, or node agent), you might have to restart the server multiple times before it functions properly as a member of the new core group.
3. Click **Move**. The **Core groups > DefaultCoreGroup > Core group servers > Move** administrative console panel is displayed showing the servers you want to move and the core group to which these servers currently belong.
4. Indicate in the To core group field the core group to which you want these servers moved.
5. Click **Apply**.
6. Click **Synchronize changes with Nodes** and then **Save** to save your changes. If you fail to perform this step, and the server you are moving is a node agent, the node agent will not be functional after you restart it. It will continue to belong to its original core group and message HMGR0087I will be issued to the SystemOut.log file, indicating that a core group merger failed. Restarting the node agent again will solve this problem and make the node agent a functional member of the core group to which it was moved.

The appropriate servers reside in each core group defined for the cell.

Start the servers you moved. If you moved a node agent, you do not have to restart the application servers associated with that node agent.

Core group servers

A core group server is an application server, a deployment manager, or a node agent that is a member of a high availability core group. Use this page to move servers into a different core group. All members of a cluster must be in the same core group. If you select one or more members of a cluster, all of the members of that cluster must be moved.

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or *Select an existing core group for editing* > **Core group servers**.

Name	Specifies the names of the servers in the core group. This field is read-only. Click on this field to the specify custom properties for this server.
Node	Specifies the node that contains the core group server. This field is read-only.
Version	Specifies the product version of the node in the cell. A Version 6 deployment manager node can own a Version 5 managed node. Version 5 managed nodes are easily identified in this field. This field is read-only.

Type	Identifies the server process type, which can be either deployment manager, node agent, or application server. Standalone application servers in the cell, managed nodes, and cluster members all display as application server types. This field is read-only.
Cluster Name	Specifies the cluster name if the core group server is part of a cluster. If the core group server does not belong to a cluster, this field is blank. This field is read-only.

To move one or more servers to another core group, select the check box next to the names of the servers that you want to move and click **Move**.

Core group server settings

Use this page to create or change custom properties for a server in a core group.

To view this administrative console page, click **Servers > Core groups > Core group settings >** ***Select an existing core group*** **> Core group servers >** ***Select an existing server***.

Extended information about the fields on the panel:

Node	Displays the name of the node that contains the core group server. This field is read-only.
Name	Displays the name of the core group server. This field is read-only.
Custom Properties	Click on this field to define or edit a custom property for the core group server.

Core group server move settings

Use this page to move one or more servers into a different core group.

Servers can be moved from one core group to another, as long as the following core group requirements are not violated:

- A non-empty core group retains at least one node agent or deployment manager as a member of that group. (The high availability manager configuration change listeners are only available on the node agent or deployment manager servers.
- All members of a cluster must be members of the same core group. If one or more of the servers you are moving belongs to a cluster, you must move all of the members of that cluster. (A core group can span multiple WebSphere clusters.)

To view this administrative console page, click **Servers > Core groups > Core group settings >***core_group* **> Core group servers**. Select the servers to be moved and then click **Move**.

Extended information about the core group fields:

Move selected servers	This field lists the servers that you selected to be moved. It can not be edited.
From core group	This field specifies the name of the core group that you are moving the servers from. It can not be edited.

To core group	From the pull-down list, select the core group that these servers will be moved into.

Click **Apply** to make your changes effective. Click **Save** and save and synchronize your changes with all managed nodes.

Routing high availability group work to a different server

How you route work items for a high availability resource to a different application server during runtime depends on how your application server environment is configured.

You might need to perform this task if you have to change the application server that is handling work items for a high availability resource, such as the service integration bus. For example, if a core group includes a cluster of application servers, and a messaging engine is configured for that cluster, any of the servers in that cluster can handle work items for the messaging engine. However, if the high availability group with which the messaging engine is associated is governed by a One of N policy, only one of these application servers can be active at a given point in time.

If you are using the WebSphere Application Server default configuration for a high availability environment, you can change the application server that is handling the work items for a high availability resource. This change can be performed as a run-time operation (the change exists only in memory) or it can be done as a permanent configuration change. To make this change a permanent configuration change, perform one of the following steps:

1. Create a preferred servers list for the policy that governs the high available group. In the administrative console, you can click **Show groups** on the Runtime tab for your core group if you need to determine the high availability group name for the resource that is involved in the change.

   For example, to change which server is handling the work items for the service integration bus, create a preferred servers list for the Default SIBus Policy, which is the default policy for the service integration bus high availability group.

   a. To view this administrative console page, click **Servers > Core groups > Core group settings > DefaultCoreGroup > Policies > Default SIBus Policy > Preferred servers**
   b. Add the desired servers to the Preferred servers list.
   c. Click **Save**, select **Synchronize changes with Nodes** and then click **Save** again to save your changes.

   As soon as you save your changes, all work items for the service integration bus will be routed to the new preferred server.
2. Create a new static policy for the high availability group to which the resource belongs. Make sure that the match criteria you specify for this new policy includes at least two of the properties specified as part of that high availability group name. To determine the high availability group name for a resource, make sure the application server that is currently handling the work items for the resource is active, and then click **Show groups** on the Runtime tab, in the administrative console, for your core group and enter an asterisk (*) in the Group name properties field.

   For example, to move a messaging engine, create a new policy for the messaging engine high availability group that has Static as the policy type. The

match criteria for this policy must include at least two of the properties that are contained in that high availability group name. The two property match will cause the new policy to be used instead of the default policy for this group.

a. Determine the high availability group name for the messaging engine you want to move. In the administrative console, click **Core groups > Core group settings > DefaultCoreGroup**, and then click on the Runtime tab. Enter an asterisk (*) in the Group name properties field. A list of the high availability groups that are contained in that core group are displayed in an administrative console panel. An example of a group name that might display on this list follows:

```
IBM_hc=AcceptanceCluster,WSAF_SIB_BUS=SSS_Bus,WSAF_SIB_MESSAGING_ENGINE=
      AcceptanceCluster.000-SSS_Bus,temp_property=WSAF_TEMP,type=jms
```

   There is not a space following the equal sign at the end of the first line. The line was split here to enable the page to be printed.

b. Create a new static policy. In the administrative console:
   1) Click **Core groups > Core group settings > DefaultCoreGroup > Policies > New**
   2) Select Static Policy for the policy type and click **Next**.
   3) Enter a policy name.
   4) Enter a policy description.
   5) Default isAlive timer setting should be 0.
   6) Do not select Quorum.
   7) Click **Apply** to save the new policy.
   8) Under Additional Properties, select **Match Criteria** and specify at least two of the properties that exist in the name of the high availability group for the messaging engine. Using the preceding group name, you might specify the following two match criteria:

```
WSAF_SIB_MESSAGING_ENGINE=AcceptanceCluster.000-SSS_Bus
type=jms
```

   9) Under Additional Properties, select **Static group servers** and specify the name of the application server on which you want the messaging engine to be active.
   10) Click **OK** and **Save** to save your changes.

New work items for the high availability group are now routed to a different server.

Configuring the core group bridge service

The core group bridge service can be configured for communication between core groups. A core group is a statically defined component of the high availability manager. To configure communication between core groups, use an access point group. An access point group is a collection of core groups that communicate with each other.

Configure two or more core groups that need to communicate with each other. For more information about core groups, see “Core groups” on page 377. To configure core groups, see “Creating a new core group” on page 374.

You must configure the core group bridge service whenever two or more core groups are configured in the same cell. You can also configure the core group bridge to share traffic among core groups that are in different cells. Configure the

core group bridge to communicate between cells only when the service is required by another WebSphere Application Server component. By configuring the core group bridge service, the availability status of the servers in each core group is shared among all the configured core groups. For more information, see "Core group communications using the core group bridge service." You can configure core groups to communicate in the following ways:

- Configure core group communication between core groups that are in the same cell. Configuring communication between any core groups that are in the same cell is required. For more information, see "Configuring communication between core groups that are in the same cell" on page 413.
- Configure core group communication between core groups that are in different cells. You can configure each cell to communicate with one or more other cells. For more information, see "Configuring communication between core groups that are in different cells" on page 419.
- Configure communication between core groups using a proxy peer access point. Sometimes, your core group might not have access to the core group that you want to communicate with. However, if you can access a core group that can communicate with the inaccessible core group, you can create a proxy peer access point. For more information, see "Configuring core group communication using a proxy peer access point" on page 425.

Multiple core groups can communicate with each other.

Continue configuring the high availability environment. See "Setting up a high availability environment" on page 368 for more information.

Core group communications using the core group bridge service

The core group bridge service can be configured to share availability information about internal WebSphere Application Server components between core groups. For example, by configuring the core group bridge service, each core group can be aware of the status of all of the application servers that are configured in all of the core groups. Use access point groups to define the core groups that communicate. Do not use the core group bridge service to share application information among core groups.

A core group is a statically defined component of the high availability manager. Each cell must have at least one core group. WebSphere Application Server creates a default core group called **DefaultCoreGroup** for each cell. For more information about core groups, see "Core groups" on page 377. Two or more core groups can be set up to communicate with each other and share workload management information by defining access point groups. The core groups that communicate can be in the same cell or in different cells.

Core group bridge overview

To configure communication between core groups, you must configure an *access point group*. An access point group is a collection of the core groups that communicate with each other. Add a *core group access point* to the access point group for each core group that needs to communicate.

A *core group access point* is a collection of server, node, and transport channel chain combinations that communicate for the core group. Each core group has one or more defined core group access points. The **DefaultCoreGroup** has one default core group access point. However, you might consider configuring more than one

core group access point for a core group if that particular core group needs to be connected to other core groups that are on different networks.

The node, server, and transport channel chain combinations that are in a core group access point are called *bridge interfaces*. A server that hosts the bridge interfaces is a *core group bridge server*. The transport channel chain defines the set of channels that are used to communicate with other core group bridge servers. Each transport channel chain has a configured port that the core group bridge uses to listen for messages from other core group bridge servers.

Each core group bridge server must have at least one bridge interface for each core group access point. However, to prevent failure of your configuration, configure multiple bridge interfaces for each core group access point. Then, if one server in one of the bridge interfaces fails, the other bridge interface can still receive information and the core group access point remains functional.

If you are configuring communication between core groups that are in the same cell, create one access point group and add a core group access point for each core group that needs to communicate. The following diagram shows an example configuration in a single cell:

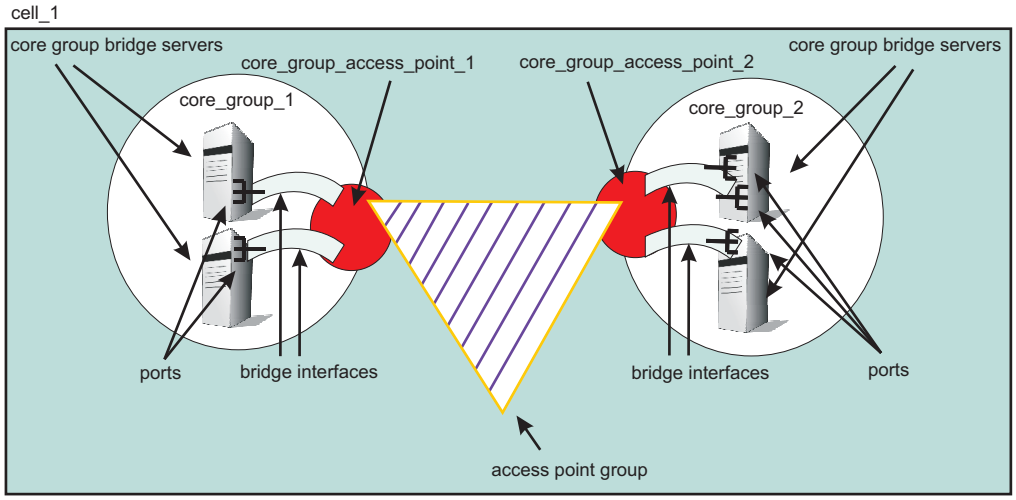


Figure 8. Core group bridge configuration in a single cell

If you are configuring the core group bridge between core groups that are in different cells, you still use an access point group. However, you must create and configure the access point group for each cell. Each cell has an access point group that contains a core group access point for the core group that is in the cell, and a *peer access point* for each peer cell.

A peer access point references a core group access point that is configured in a different cell. Each access point group must have one peer access point for each different cell. Do not configure multiple peer access points that reference the same cell.

Each peer access point has one or more *peer ports* or one *proxy peer access point*.

A peer port corresponds to a bridge interface that is defined in the peer cell. You can define several peer ports for each peer access point.

Define a proxy peer access point if the peer access point cannot be reached directly by using a peer port, but can be reached by using another peer access point. The proxy peer access point specifies a peer access point that can communicate with the peer core group that cannot be reached directly. The proxy peer must have defined peer ports. Specify one proxy peer or one or more peer ports, but not both.

The following diagram shows a core group bridge configuration between two different cells that is using peer access points with peer ports.

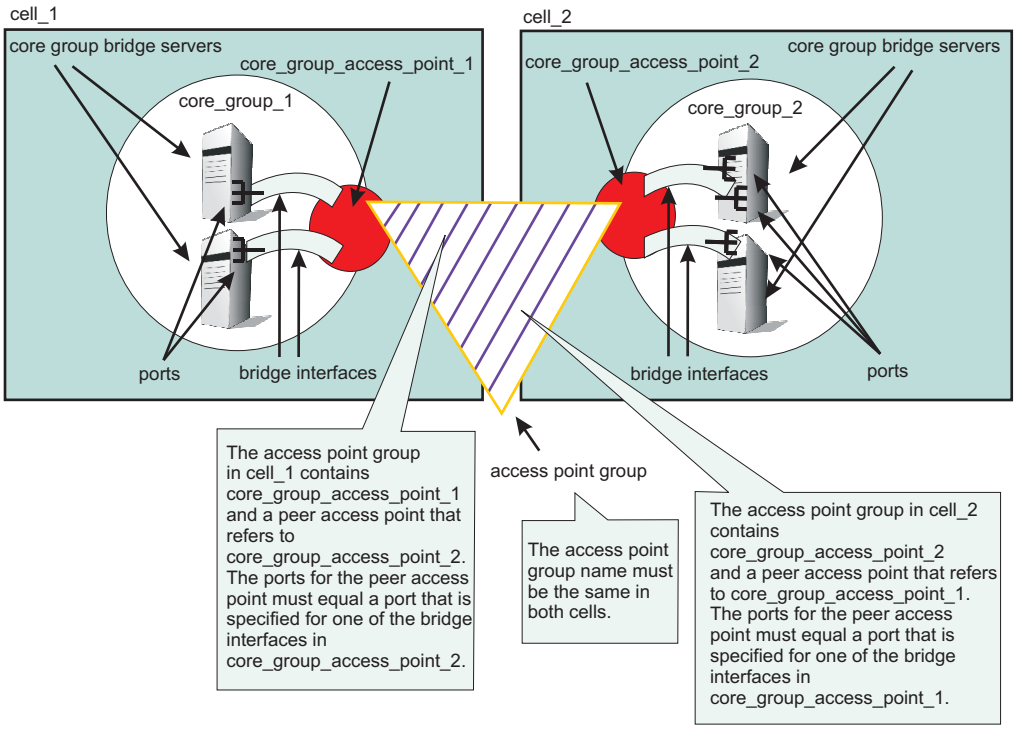


Figure 9. Core group bridge configuration in two different cells

Configuration scenarios

You can configure core groups to communicate in the following ways:

- "Communication between core groups that are in the same cell"
- "Communication between core groups that are in different cells" on page 411
- "Communication between core groups across different networks" on page 412
- "Communication between core groups using a proxy peer access point" on page 412

Communication between core groups that are in the same cell

All core groups that are in the same cell must be configured to communicate with each other. To configure core group communication within a cell, create one access point group with one core group access point for each core group. Select one or more servers to be core group bridge servers, and define a bridge interface for each server. The following image shows an example of three core groups that are in the same cell and are connected by one access point group. The sample configuration

shows how communication between core groups in the same cell is configured in the administrative console.

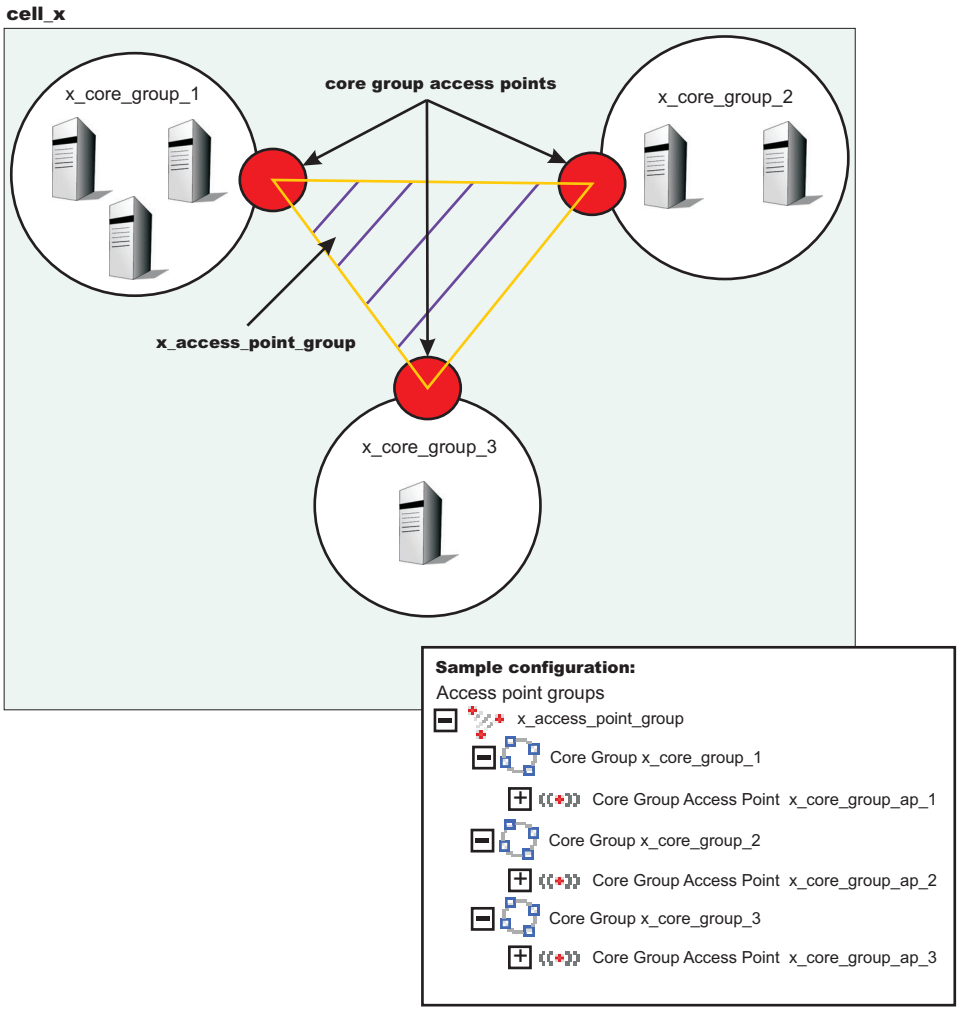


Figure 10. Communication between core groups that are in the same cell

See "Configuring communication between core groups that are in the same cell" on page 413 for more information.

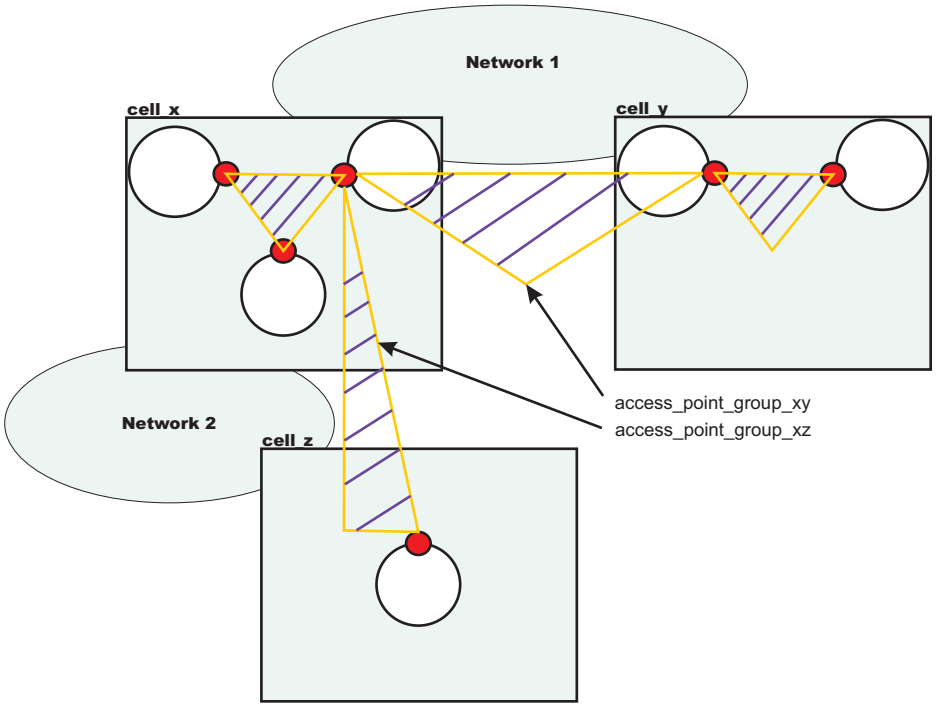
Communication between core groups that are in different cells

When two core groups in different cells need to communicate, add peer access points to the access point groups. To configure the core group in *cell_x* to communicate with the core group in *cell_y*, add a peer access point to the access point group in *cell_x*. The peer access point is equal to the core group access point for *cell_y*. Configure an access point group that has the same name for *cell_y* that contains the core group access point for the core group in *cell_y* and a peer access point that corresponds to the core group access point for the core group in *cell_x*. Each peer access point also contains peer ports that identify bridge interfaces in the opposite cells. See "Configuring communication between core groups that are in different cells" on page 419 for more information.

Communication between core groups across different networks

In this scenario, one core group is configured to communicate with two or more core groups in different cells across two or more networks. For example, a core group in *cell_x* needs to communicate with core groups in *cell_y* and *cell_z*. Create two access point groups in *cell_x*. The first access point group, *access_point_group_xy*, in *cell_x* contains a core group access point and a peer access point for the core group in *cell_y*. The second access point group in *cell_x*, *access_point_group_xz*, contains a core group access point and a peer access point for the core group in *cell_z*. *Cell_y* has an access point group named *access_point_group_xy*, which has a core group access point and a peer access point for *cell_x*. *Cell_z* has an access point group named *access_point_group_xz* which has a core group access point and a peer access point for *cell_x*.

Configuring access point groups across multiple networks



See “Configuring communication between core groups that are in different cells” on page 419 for more information.

Communication between core groups using a proxy peer access point

Use a proxy peer when the core groups cannot directly communicate. The two core groups must have access to a single core group that can pass information between the two core groups. To understand what a proxy peer access point does, consider a connecting flight when flying on an airplane. To fly from Pittsburgh to London you first have to fly to New York City, where you change planes and then fly to London. New York City is the *proxy peer access point* for London.

When defining a proxy peer, *x_core_group_2* in *cell_x* cannot communicate directly with the core group in *cell_z*. However, both core groups can communicate with the core group in *cell_y*. To configure communication between *cell_x* and *cell_z*, you must configure two access point groups. The core group access point in *cell_y* is in both of these access point groups, named *access_point_group_xy* and

access_point_group_yz. The following image shows an overview of a proxy peer configuration.

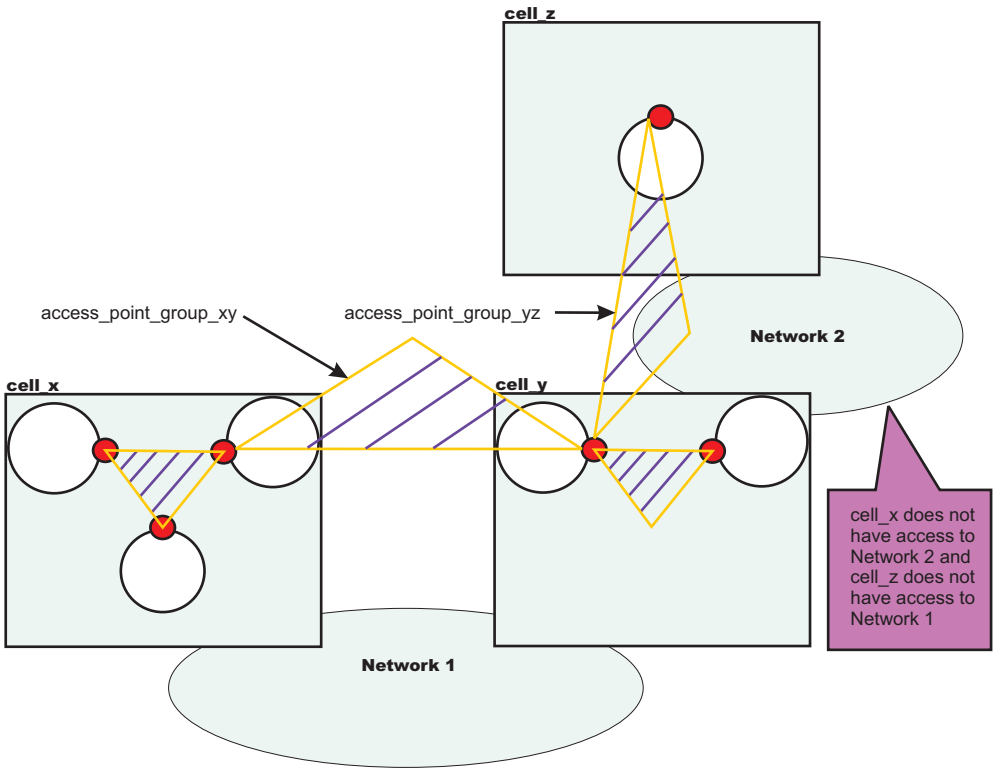


Figure 11. Core group communication using a proxy peer access point

See "Configuring core group communication using a proxy peer access point" on page 425 for more information.

Configuring communication between core groups that are in the same cell

Use this task to configure communication between core groups that are in the same cell.

Configure two core groups with application servers that are in the same cell. For more information about when and how to configure multiple core groups, see "Creating a new core group" on page 374. Any time you configure multiple core groups that are in the same cell, you must configure communication between the core groups.

Each core group has one or more defined core group access points. A core group access point is a collection of network entry points for the core group. The network entry points that are in a core group access point are called bridge interfaces. An application server that hosts the bridge interfaces is called a core group bridge server.

To configure communication between core groups, define an access point group. An access point group defines the core groups that can communicate with each other. Each access point group has one core group access point for each core group. Select one or more servers to be core group bridges, and define a bridge interface for each server.

See "Core group communications using the core group bridge service" on page 408 for more information about the core group bridge service.

1. Configure an access point group to define the core groups that need to communicate. An access point group contains the core group access points for the core groups that need to communicate. Core group access points define the set of servers that provide access to the core group. To configure communication between core groups that are in the same cell, you can choose an existing access point group or create a new access point group. To create an access point, complete the following steps:
   a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups > New**.
   b. Enter a name for the access point group that is unique within the cell.
   c. Add core group access points to your access point group. Choose any available core group access points for the core groups that need to communicate in the cell. A default core group access point is created whenever a core group is created. The access point group that you create must have a core group access point for each core group in the cell.

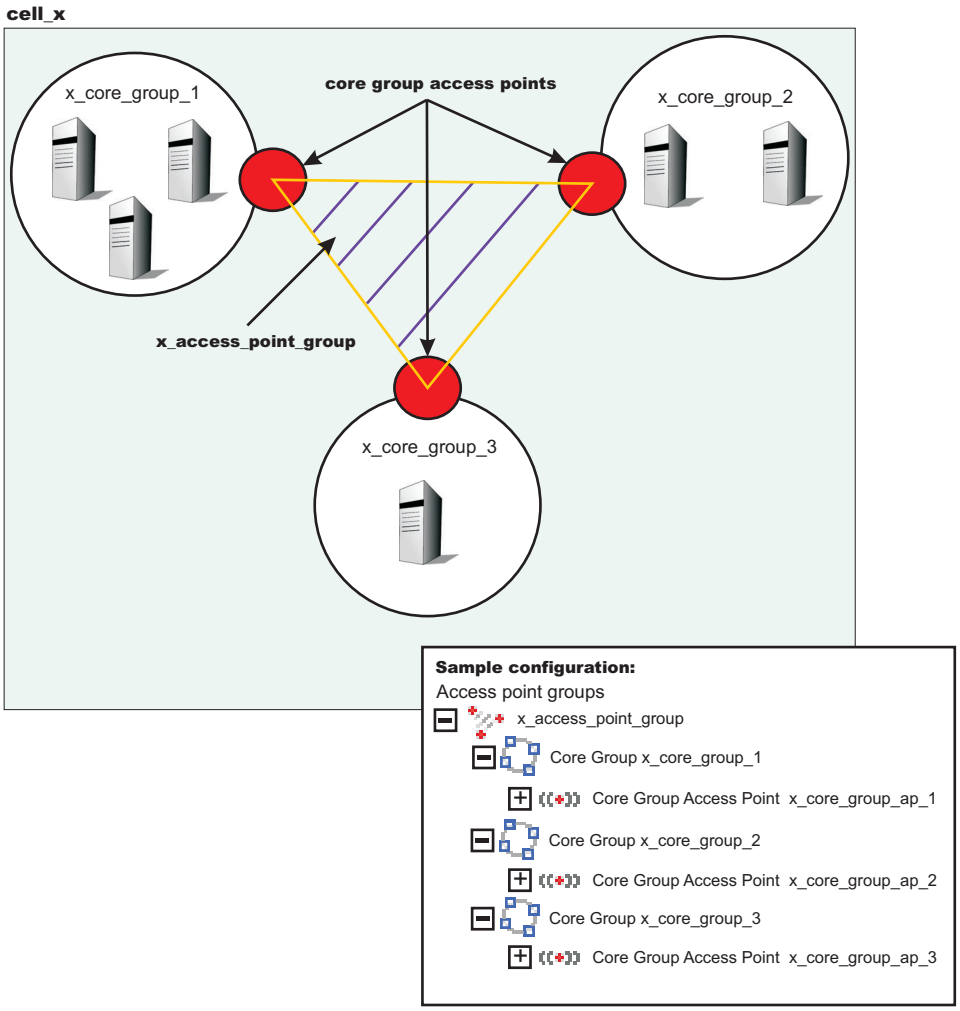
      Restriction: Do not add any peer access points. Add peer access points only if you are configuring communication with a core group in a different cell. If you need to communicate with core groups that are outside of the cell, you must create another access point group that has one core group access point and one or more peer access points. See "Configuring communication between core groups that are in different cells" on page 419 for more information.

   If you use an existing access point group, choose an access point group that does not have peer access points. To configure an existing access point group, perform the following steps:
   a. In the administrative console, click **Servers > Core groups > Core group bridge settings**. Your current configuration with any existing access point groups is displayed.
   b. Verify that the access point group does not have any peer access points. Peer access point groups are used for communication between core groups in different cells. Click the access point group you want to configure and ensure that no peer access points are listed.
   c. Click **Access point groups >** *access_point_group_name* **> Core group access points**.
   d. Add core group access points to your access point group. Choose any available core group access points for the core groups that need to communicate. The access point group you create should have a core group access point for each core group in the cell.
2. Create bridge interfaces for each core group access point. The bridge interfaces that you add provide access to the core group. Create at least one bridge interface for each core group access point. To back up your configuration, you should configure two or more bridge interfaces for each core group access point. If a core group has multiple core group access points, each core group access point must contain the same number of bridge interfaces for the same set of servers. To configure bridge interfaces, perform the following steps:
   a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points**.
   b. Click a core group access point in the access point group. Click **Show Detail**.

c. To create a new bridge interface, click **Bridge interfaces > New**.
d. Select a node, server, and transport channel chain combination for the bridge interface. Click **OK**. All the core group access points in the access point group must have transport channel chains with the same port name. The transport channel chain can be the DCS or DCS-secure channel chains that are created for the DCS_UNICAST_ADDRESS transport chain.
e. Consider creating at least two bridge interfaces for each access point. If one bridge interface fails, the other can still be active.
f. Repeat these steps to create bridge interfaces for each core group access point in your access point group.

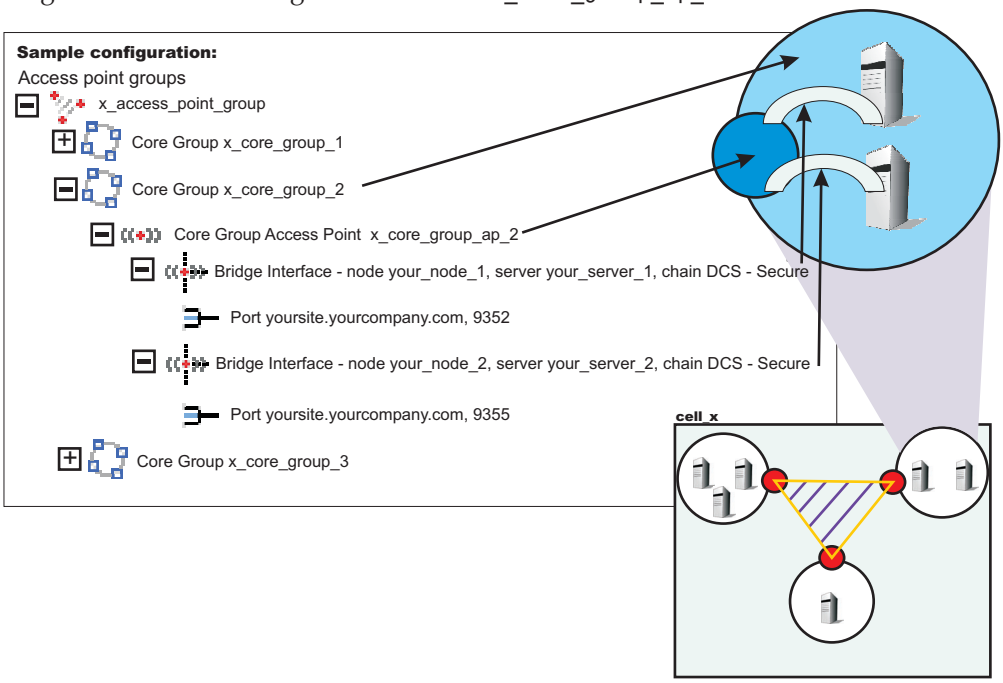
The core groups that are in the same cell and configured in an access point group can communicate.

In *cell_x*, there are three core groups: *x_core_group_1*, *x_core_group_2*, and *x_core_group_3*. Each core group already has a core group access point. The following image shows an access point group between the core groups in *cell_x* and an example of the configuration in the administrative console.



Perform the following steps to configure communication between the three core groups in *cell_x*:

1. Create an access point group named `x_access_point_group`. Add a core group access point to the access point group for each core group that is in the cell. In this example, add `x_core_group_ap_1`, `x_core_group_ap_2`, and `x_core_group_ap_3` to `x_access_point_group`.
2. Create bridge interfaces for each core group access point. The following diagram shows the bridge interfaces for `x_core_group_ap_2`:



   Create two or more bridge interfaces for each core group access point.

By creating an access point group and adding all core groups in the cell to the access point group, you enabled communication between all the core groups that are in *cell_x*.

You can configure this cell to communicate with core groups in other cells. See "Configuring communication between core groups that are in different cells" on page 419 and "Configuring core group communication using a proxy peer access point" on page 425 for more information.

Core group access point settings

Use this page to configure your core group access points. The core group access point defines the set of core group bridge servers that provide access to the core group. A core group bridge server is an application server that is configured to run the core group bridge service. Define unique core group access points for each different network over which you want the core group in this cell to connect to other core groups that are defined in another cells. The core group access point has a collection of bridge interfaces. Each server that is used as a bridge must have a unique bridge interface for every core group access point in the core group.

For example if you define access points *access_point_a* and *access_point_b* and servers *server_x* and *server_y* are configured as core group bridges in a core group, *server_x* must have unique bridge interfaces for both *access_point_a* and *access_point_b*. The *server_y* server must also have unique bridge interfaces for both *access_point_a* and *access_point_b*.

When a you create a core group, a core group access point is automatically created. Do not delete the last access point in a core group.

The core group access point that is automatically defined belongs to a default access point group. You can use the default core group access point and access point group to configure communication between core groups. You must create and configure bridge interfaces for the default core group access point. See "Bridge interface settings" on page 418 for more information about bridge interfaces.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >***access_point_group_name* **> Core group access points >***core_group_access_point_name* >**Show detail**.

Name:

Specifies the name for the core group access point.

Core group:

Specifies the core group that is associated with this core group access point.

Core group access point collection

Use this page to configure your set of core group access points. Core group access points define the set of servers that provide access to the core group. At least one core group access point must be defined for each core group in the local cell.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >***access_point_group_name* **> Core group access points**.

Available core group access points:

A list of core group access points that are available to add to the access point group.

Core group access points in *access point group***:**

A list of core group access points that are in the specified *access point group*.

Bridge interface collection

Each core group access point has a collection of bridge interfaces. This collection defines the interfaces on the set of servers that provide access to the core group. All the servers in this collection have the core group bridge service enabled. The core group bridge service provides communication between core groups. A bridge interface defines the node, the server, and the chain for the core group access point. The chain defines the transport channels that are used by the server for receiving information.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >***access_point_group_name* **> Core group access points >***access_point_name* **> Show detail > Bridge interfaces**.

Server:

Specifies the node and the server combinations that are bridge interfaces for the core group access point.

Chain:

Specifies the transport channel chain that is used for transport by the bridge interface. For all the core group access points in an access point group, the transport channel chains must resolve to the same host. To ensure that the transport channel chains resolve to the same host, use the same chain for all of the core group access points in an access point group.

Bridge interface settings

A bridge interface is a particular node and server that runs the core group bridge service. The core group bridge service is the service that provides communication between core groups. A bridge interface is defined by a unique combination of a node, server, and transport chain. You cannot configure a cluster of servers to run the same bridge interface. A transport chain represents a network protocol stack that is operating within an application server.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >***access_point_group_name* **> Core group access points >***access_point_access_point_name* **> Show detail > Bridge interfaces >***server_node*.

Bridge interfaces: Displays a list of server, node, and chain combinations that are available to become bridge interfaces for your core group access point.

Node: Displays the node on which the core group bridge service is running.

Server:

Displays the application server on which the core group bridge service is running. The server you select can have other responsibilities as well as being a core group bridge server. The server does not have to be running as a standalone node.

Chain:

Displays the transport channel chain that is used by the core group bridge service for this core group access point. The transport channel chain can be the DCS or DCS-secure channel chains that are created for the DCS_UNICAST_ADDRESS transport chain for each application server.

Bridge interface creation

A bridge interface specifies a particular node and server that runs the core group bridge service. A bridge interface is defined by a unique combination of a node, a server, and a transport chain. A transport chain represents a network protocol stack that is operating within an application server.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >***access_point_group_name* **> Core group access points >***access_point_name* **> Show detail > Bridge interfaces > New**.

Available bridge interfaces:

Specifies the node, server and transport channel chain combinations that are available to become bridge interfaces for this core group access point. Only bridge interfaces with transport chains that contain Transmission Control Protocol (TCP) inbound channels using the same port name as existing bridge interfaces in this core group access point are displayed. The bridge interfaces that are already used by any core group access point are not displayed.

Configuring communication between core groups that are in different cells

Use this task to configure core group communication between core groups that are in different cells.

A core group is a statically defined component of the high availability manager. For more information about core groups, see "Core groups" on page 377. For this task, configure core groups that are in different cells. For more information about when and how to create multiple core groups, see "Creating a new core group" on page 374. Before you configure a core group to communicate outside a cell, enable core group communication between the core groups that are within each cell. For more information, see "Configuring communication between core groups that are in the same cell" on page 413.

Configure the core group bridge service between cells to share the availability status of the servers in each core group among all the configured core groups. Enable the core group bridge service to communicate between cells only when it is required by another WebSphere Application Server component.

When you configure communication between core groups that are in different cells, you must configure an access point group that has the same name in each cell. Each access point group has exactly one core group access point and one or more peer access points. The peer access point corresponds to the core group access point that is in the other cell. Configure one access point group between the cells. Do not create multiple access point groups that allow communication between the same two cells.

Access point groups and access points are abstractions. Therefore, they are not considered a point of failure. An access point fails only if all of servers fail that are specified in the bridge interfaces for that access point. Create one or more bridge interfaces for each core group access point to prevent failure of access point groups. See "Configuring communication between core groups that are in the same cell" on page 413 for more details about bridge interfaces.

Create separate access point groups for each unique network over which the core groups communicate.

1. Configure an access point group that has peer access points for a core group in a different cell. You can select an access point group or create a new access point group. To create a new access point group perform the following steps:
   a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups > New**.
   b. Type a **Name** for the access point group that is unique within both of the cells.
   c. Add one core group access point for the core group in the local cell. When configuring an access point group that communicates outside of the cell, always configure exactly one core group access point and one or more peer access points.
   d. Add an existing peer access point for the core group in the remote cell.
   e. Confirm the changes to save your new access point group.

   To add peer access points to an existing access point group, perform the following steps:
   a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Core group access points**. Check that this access point group contains only one

core group access point. You cannot configure an access point group that communicates both inside and outside of the cell. Any access point group for core group communication outside of the cell must contain one core group access point and one or more peer access points.

b. Add existing peer access points or create a new peer access point. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points**. To add an existing peer access point, move the peer access point to your access point group. To create a new peer access point, click **New**. The name that you enter for the peer access point must be unique in the cell. To create new peer access points, you must have the following configuration information for the core group in the remote cell:
   - The name of the cell for the core group
   - The name of the core group
   - The core group access point for that core group
   - The host and port information for the core group

   When you save the new peer access point, it is automatically added to your access point group.

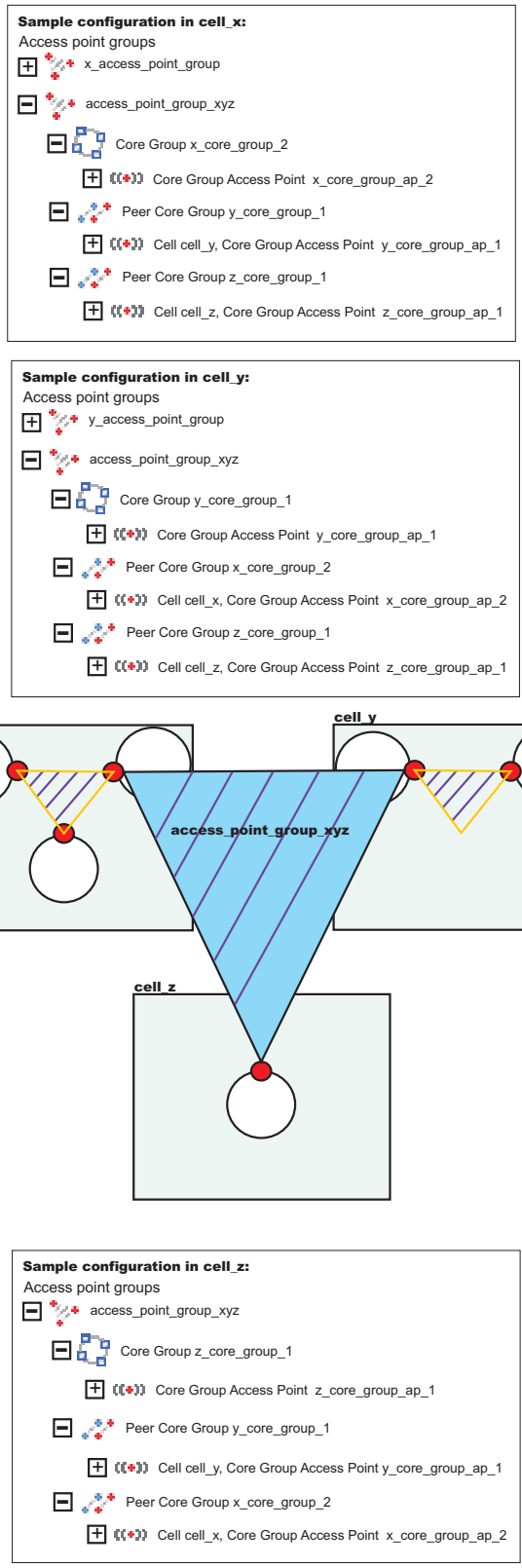
2. Configure peer ports for each peer access point. A peer port identifies the host name and port of an application server that is defined as a bridge interface in the other cell, so find the host name and port information before completing this step.
   a. In the administrative console, click **Servers > Core groups > Core group communications > Access point groups >***access_point_group_name* **> Peer access points**. Select a peer access point that is in your access point group and click **Show Detail**.
   b. Enable this peer access point to use peer ports. Select **Use peer ports**.
   c. Configure the peer ports. Click **Peer ports**. You can select and remove exiting peer ports, or create new peer ports.

   Configure only one peer access point for each cell that you need to communicate with. Do not configure multiple peer access points in one cell that correspond to the same remote cell.

3. Repeat these steps for the core group in the other cell. The access point group that you create must have the same name as the access point group that you defined in the first cell. The peer access points that you create in this access point group correspond to the core group access point in the first access point group.

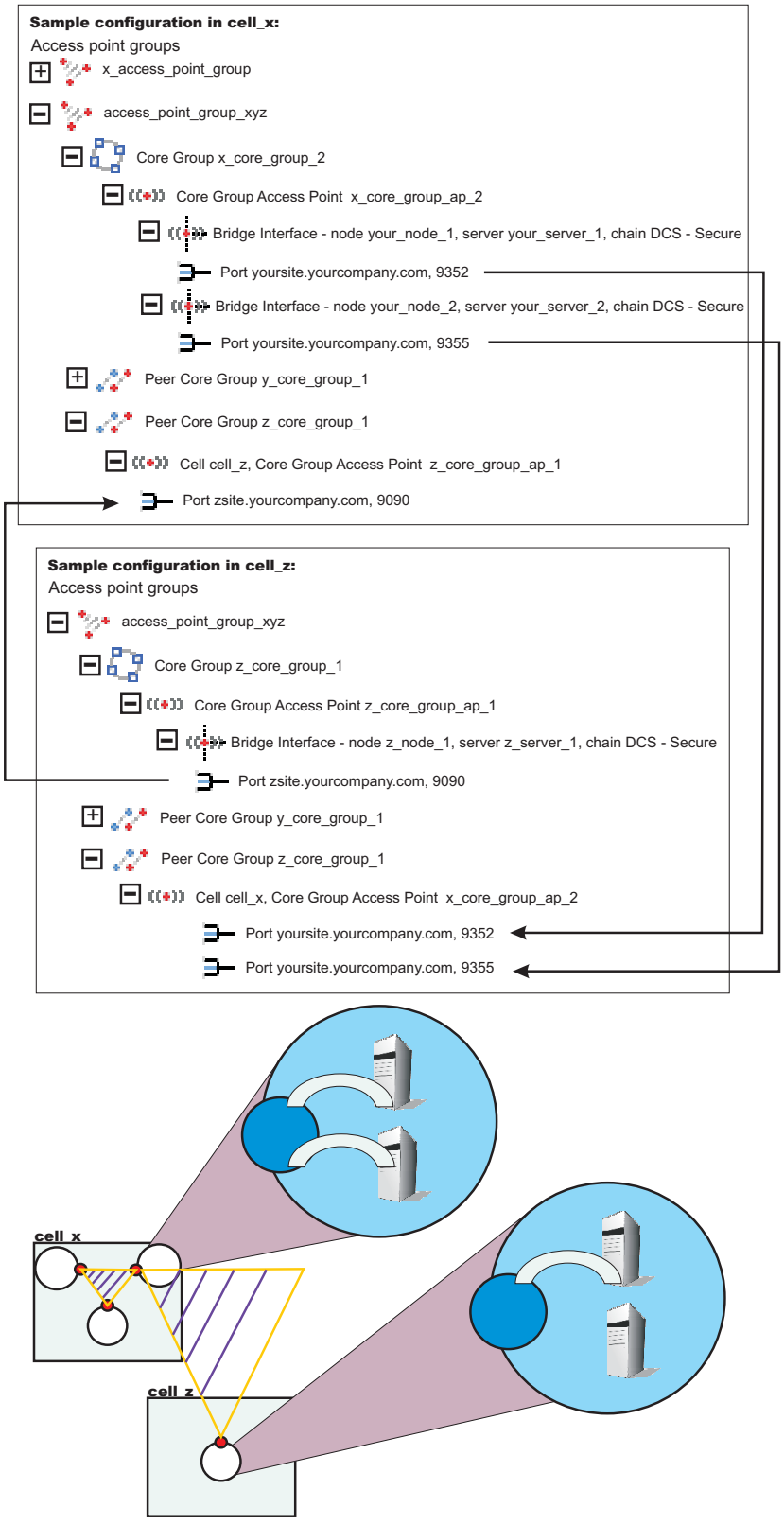
When you configure core groups in different cells to communicate with each other, all the core groups in each cell receive information from the core groups in the other cells.

Following is an example of a configuration between three core groups that are in three different cells. Each cell has one access point group for communication between core groups in the cell. Each cell also has a defined access point group, *access_point_group_xyz*, which contains one core group access point group for the core group that is in the cell, and one core group access point for each of the core groups in the other two cells.



The following sample shows the relationship between bridge interfaces and peer ports for the communication between *cell_x* and *cell_z*. In *cell_x*, there are two bridge interfaces defined. In *cell_z*, there is a peer access point for

x_core_group_ap_2 that has peer ports defined that correspond to the bridge interface information that is defined in *cell_x*.



As a result, *core_group_x* , *core_group_y* and *core_group_z* can communicate with each other.

Continue configuring core group communication and setting up the high availability environment. See "Configuring the core group bridge service" on page 407 and "Setting up a high availability environment" on page 368 for more information.

Peer access point settings

Each peer access point is used to communicate with core groups in other cells. A peer access point corresponds to a core group access point in the peer cell. The peer access point communication settings are specified by using one or more peer end points or a proxy peer.

A peer access point must contain either peer ports or a proxy peer access point, but not both. When the peer access point is directly accessible within its access point group, specify peer ports. When the peer access point can be reached only indirectly, use a proxy peer access point. A proxy peer access point is used to identify the communication settings for the peer access point that cannot be accessed directly. The proxy peer access point specifies a peer access point that can communicate with the appropriate destination core group. The specified proxy peer access point must be a peer access point that has defined ports.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points >***peer_access_point_name* **> Show detail**.

Name:

Specifies the name of the peer access point. The name must be unique within the local cell.

Cell:

Specifies the cell in which the peer access point resides.

Core group:

Specifies the core group in which the peer access point resides.

Core group access point:

Specifies the name of the core group access point that is in the peer cell.

default	defaultCoreGroupAccessPoint

Use peer ports:

Specifies that you are using peer ports instead of a proxy peer access point. Use peer ports when the peer access point is directly accessible within its access point group. Click **Peer ports** to specify the peer ports for the peer access point.

Use proxy peer access point:

Specifies that you are using a proxy peer access point instead of peer ports. A proxy peer is defined when the peer access point can be reached only indirectly

through another peer access point. A proxy peer is used to identify the communication settings for the peer access point that cannot be accessed directly. The proxy peer specifies a peer access point that can communicate with the destination core group. The specified proxy peer must be a peer access point that has defined peer ports.

Proxy peer access point:

Specifies the specific peer access point that is used to access a core group.

Peer access point collection

Peer access points are used to communicate with core groups that are in other cells. A peer access point collection is the set of peer access points that are used to communicate with the core group access point in this access point group. Specify a single peer access point for each remote cell. This collection cannot contain two peer access points for the same cell.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points**.

Available peer access points:

A list of peer access points that are available to join the access point group.

Peer access points in *access_point_group***:** A list of peer access points that are in the selected access point group.

Peer port settings

Use this page to define a peer port. A peer port identifies the host name and port of an application server that is a bridge interface in another cell. This application server is using the core group bridge service to communicate with other core groups. Each peer access point can have one or more peer ports. Each port identifies a bridge interface of a core group bridge service in the peer cell.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points >** *peer_access_point_name* **> Show detail > Peer ports >** *peer_port_name*.

Host:

Specifies the host name on which the core group bridge in the remote cell is listening.

Port:

Specifies the port number that is associated with the host on which the core group bridge in the remote cell is listening.

Peer port collection

Use this page to define the peer ports for the peer access point. Each peer port identifies a bridge interface of a core group bridge service in the peer cell. Each peer access point that does not have a proxy peer must have one or more peer ports.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >** *access_point_group_name* **> Peer access points >** *peer_access_point_name* **> Show detail > Peer ports**.

Host:

Specifies the host name that is used by the bridge interface in the remote cell.

Port:

Specifies the port that is used by the bridge interface in the remote cell.

Configuring core group communication using a proxy peer access point

Use this task to configure communication between two core groups when they cannot communicate with each other directly through peer ports.

Configure a proxy peer access point to communicate with a core group if you cannot use peer ports. Use a core group that has configured a peer access point with peer ports to the core group that you want to communicate with. Before completing this task, make sure that you have access to a core group that can communicate with the core group that you cannot communicate with directly. To configure communication between core groups that are in different cells, see "Configuring communication between core groups that are in different cells" on page 419. If there are multiple core groups in each of the cells, they must be configured to communicate with each other. See "Configuring communication between core groups that are in the same cell" on page 413 for more information.

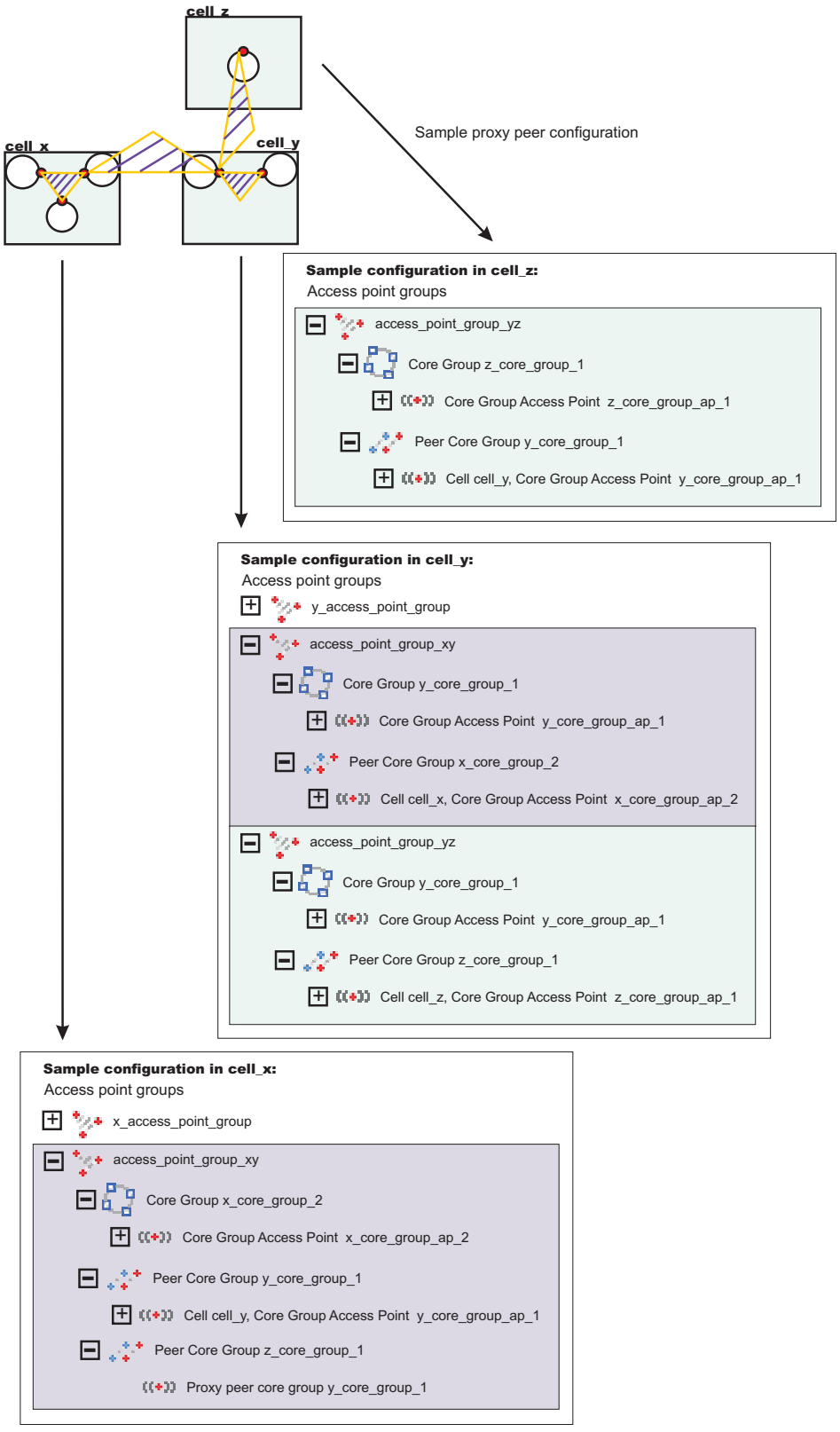
Use a proxy peer to communicate with a core group that you cannot access directly. This task describes how to configure a proxy peer with three core groups that are each in different cells. *Core_group_x* and *core_group_y* can communicate directly with each other through peer ports. *Core_group_y* and *core_group_z* can also communicate with each other though peer ports. However, *core_group_x* cannot communicate with *core_group_z*. To establish that communication, *core_group_x* has a peer access point that is a proxy peer. The proxy peer is the peer access point to *core_group_y*. For more information, see "Core group communications using the core group bridge service" on page 408.

1. Configure *core_group_x* and *core_group_y* to communicate with each other by creating an access point group. For more information, see "Configuring communication between core groups that are in different cells" on page 419.
2. Configure *core_group_y* and *core_group_z* to communicate with each other by creating another access point group. For more information, see "Configuring communication between core groups that are in different cells" on page 419. When you do this step, create a second access point group in cell 2. *Core_group_2* communicates with both *cell_x* and *cell_z* over two different networks.
3. Configure a peer access point that has a proxy peer. Create a new peer access point in the access point group that you created between *core_group_x* and *core_group_y*.
   a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups >***access_point_group_name***> Peer access points**. Create a new peer access point, or select an existing access point.

b. Enter a unique name for the peer access point. For the other cell, core group, and core group access point values, use the properties of *core_group_z*.
c. Click **Use a proxy peer access point**. Select the proxy peer access point that is the peer access point that you created in *cell_x* that refers to the core group access point in *cell_y*.

Core_group_x can communicate with *core_group_z* by using a proxy peer.

The following example shows the configurations in each cell when you configure communication between *cell_x* and *cell_z* using a proxy peer access point:



By completing this task, you enabled one-way communication between *core_group_x* and *core_group_z*. If you want to configure communication both ways, you must repeat these steps, configuring a peer access point in *core_group_z* that contains a proxy peer.

Core group communication

The core group bridge is the service that enables communication between core groups. A core group is a statically defined component of the high availability manager. Use this page to view the structure of your access point groups. Access point groups link core groups that are in the same cell or in different cells and allow the core groups to communicate with each other.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings**.

Each access point group is a collection of core group access points. If you are configuring communication between core groups that are in the same cell, configure an access point group with a core group access point for each core group in the cell. If you are configuring communication between core groups in different cells, configure an access point group that has one core group access point for the local cell and a peer access point for each other cell.

Each core group access point has one or more bridge interfaces. Each peer access point has a proxy peer or one or more peer ports. A bridge interface is a server that is configured to communicate with other core groups by using a particular transport channel chain. Click **Access point groups** to configure the settings for each access point group that is configured.

- **Access point group** - An access point group defines the core groups that communicate with each other. Each access point group consists of a collection of core groups.
  - **Core group** - Specifies a core group that is in this access point group. Core groups are referenced by core group access points.
    - **Core group access point** - The core group access point defines the set of servers that provide access to the core group. Bridge interfaces define the servers that are in the core group access point.
      - **Bridge interface** - A bridge interface defines a node, server, and chain for the core group access point. The chain defines the transport channels that the server uses for receiving information. Typically, all the bridge interfaces in a core group access point use the same chain.
  - **Peer core group** - Specifies a core group in a different cell. Define peer access points to communicate with peer core groups.
    - **Peer access point** - Each peer access point is used to communicate a core group in a different cell. Each peer access point corresponds to a core group access point that is in the peer cell. Use one or more peer ports or one proxy peer to define the communication settings.
      - **Peer port** - Each peer port identifies a bridge interface of a core group bridge in the peer cell.
      - **Proxy peer** - A proxy peer is used to identify the communication settings for a peer access point that cannot be accessed directly through peer ports. A proxy peer specifies a peer access point that can communicate with the destination core group. The specified proxy peer must be a peer access point that has defined ports.

Access point group collection

Use this page to view the sets of access point groups. Access point groups define the set of core groups that communicate with each other. Access point groups that connect multiple cells must have one core group access point and a single peer access point for each remote cell. Access point groups that provide communications between core groups in the same cell must contain only core group access points.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups**.

Name:

Specifies the name of the access point group. The access point group name must be unique within the cell.

Access point group settings

Use this page to modify the core group access points and the peer access points that belong to this access point group. An access point group defines the set of core groups that communicate with each other. Group the access points to support communication. Access points can be either peer access points or core group access points. Define core group access points so that core groups in the same cell can communicate. Define peer access points so that core groups in different cells can communicate.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups >***access_point_group_name*.

From this page, you can edit the core group access points or the peer access points that belong to the access point group.

Name:

Specifies the name of the access point group. The access point name must be unique within a cell.

Controlling application rollout and workload routing in a high availability configuration

Prepare your environment for high availability Application Update.

Workload routing is controlled by stopping the Application Server on which the cluster member being updated resides. This results in a quiesce of that server. All existing requests already in the server are allowed to complete, but no new requests are accepted. Both the Sysplex Distributor and the WebSphere Application Server's Web server plug-in will route work away from the quiescing server. After all work has completed, start the application update process on this server:

1. Disable all forms of Automatic Synchronization, across all nodes in the Cell and save the changes. Perform one of the following processes to complete this step:
   - Using the Administrative Console:
     a. Select **System Administration** > **Node agents** > *node_agent_name* > **File Synchronization Service**.
     b. Unselect the **Automatic Synchronization** and **Startup Synchronization** checkboxes.
     c. Select the **Synchronize changes with nodes** checkbox.
     d. Click **Save**.
   - Use wsadmin scripting to specify the following commands and then restart all affected node agents:

```
set node NODE
set na_id [$AdminConfig getid /Node:$node/Server:nodeagent/]
set syncServ [$AdminConfig list ConfigSynchronizationService $na_id]
$AdminConfig modify $syncServ {{autoSynchEnabled false}}
```

```
$AdminConfig modify $syncServ {{synchOnServerStartup false}}
$AdminConfig save

set nodeSync [$AdminControl completeObjectName type=NodeSync,node=$node,*]
$AdminControl invoke $nodeSync sync
```

Note: For a production environment, it is reasonable to always run the node agent with automatic synchronization disabled. However, it is advisable for startup synchronization to be enabled for the node agent so that it can acquire configuration updates that occur when the node agent is down. Startup Synchronization can be left enabled provided you can ensure that you will not restart the node agent manually, through automation, or through automatic restart manager during the application update process.

2. Update the application in the Master Configuration Repository on the deployment manager server. Perform one of the following processes to complete this step:
   - Using the Administrative Console:
     a. Select **Applications** > **Enterprise Applications**.
     b. Select the checkbox next to the application you want to update.
     c. Complete the application update process.
     d. Save your changes to the Master Configuration. **DO NOT** select the **Synchronize changes with nodes** checkbox .
   - Use wsadmin scripting to issue the following command:

     ```
     set app_loc /path/to/app
     set app_options {application options ie: -appname app}
     set options [list -update]  lappend options $app_options
     $AdminApp install $app_loc $options
     $AdminConfig save
     ```

     At this point, you have the updated the version of your application (App v2 in the following figure) in your Master Configuration. However, the original version of your application (App v1 in the following figure) is still running in the cluster that has Cluster members on LPAR1 and LPAR2.

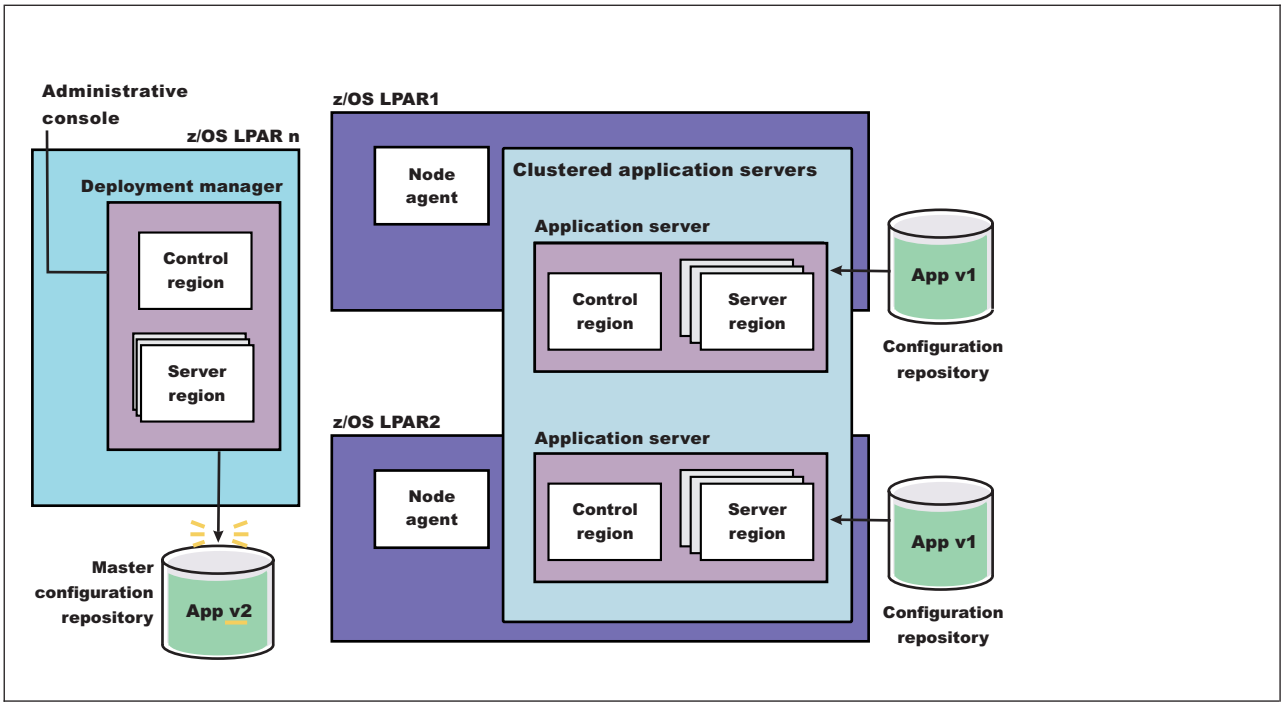


Figure 12. Install application update. This figure illustrates the first stage of an application update in a high availability environment.

3. Stop the Application Server on LPAR1 and manually synchronize the node to the updated version of the application. This step may take time to complete because the server must wait for all currently assigned work items to complete before shutting down.

   Perform one of the following processes to complete this step:

   - Using the Administrative Console:
     a. Select **Servers** > **Application Servers** .
     b. Select the checkbox next to the cluster member you want to stop and update. This cluster member should be on LPAR1.
     c. Select **STOP**. The Cluster Stop method should not be used, because it will stop all Servers within the cluster and the application will no longer be available.
   - Use wsadmin scripting: to issue the following commands:

     ```
     set node NODE
     set server SERVER
     $AdminControl stopServer $server $node
     ```

   - Issue the following command from the MVS Console:

     ```
     STOP short_server_name
     ```

     For example:

     ```
     STOP BBOS001
     ```

4. Synchronize the node. Perform one of the following processes to complete this step:

   - Using the Administrative Console:
     a. Select **System Administration** > **Nodes**.
     b. Select the checkbox next to the node you want to synchronize, and then select **Full Resynchronize**.
   - Use wsadmin scripting to issue the following commands:

```
set node NODE
set nodeSync [$AdminControl completeObjectName type=NodeSync,node=$node,*]
$AdminControl invoke $nodeSync sync
```

As illustrated in the following figure, the updated version of the application (App v2) now resides in the node on LPAR1.

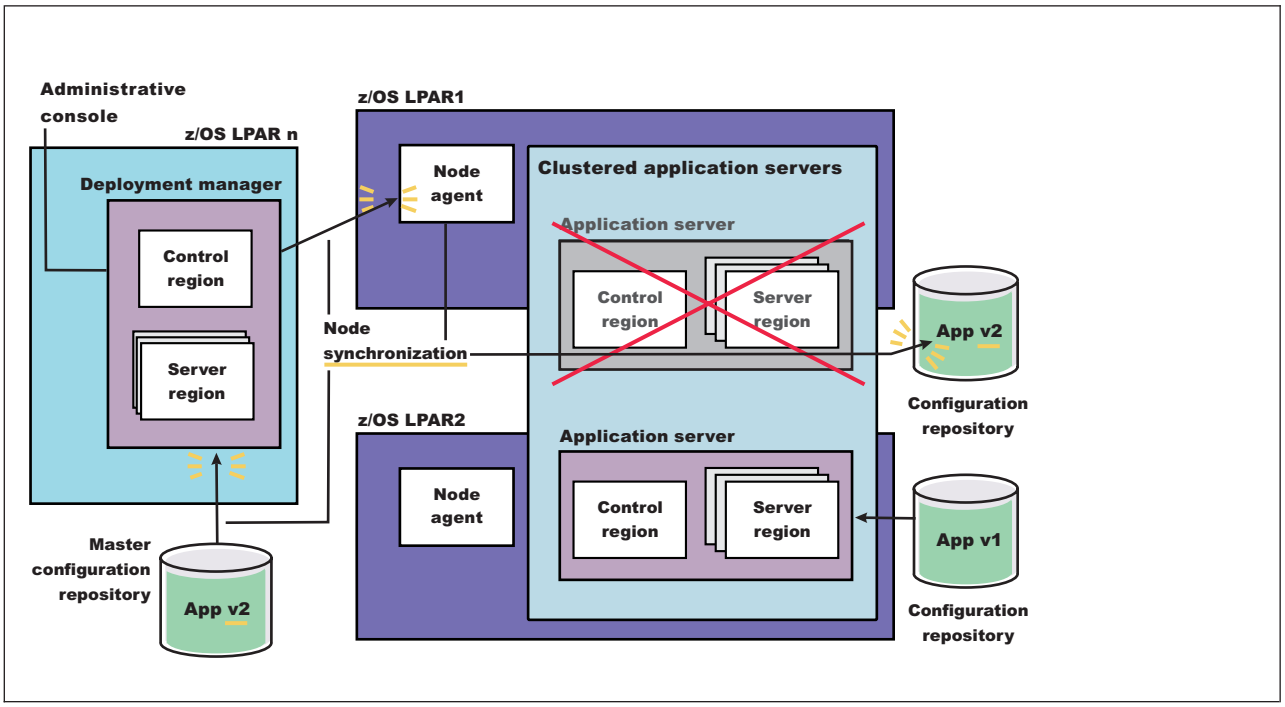


Figure 13. Update the node on LPAR1. This figure illustrates the first stage of an application update in a high availability environment with two LPARs.

5. Restart the server on LPAR1. Perform one of the following processes to complete this step:
   - Using the Administrative Console:
     a. Select **Servers**.
     b. Select the checkbox next to the server you want to start, and then select **START**.
   - Use wsadmin scripting to issue the following commands:

     ```
     set node NODE
     set server SERVER
     $AdminControl startServer $server $node
     ```

   - Issue the following command from the MVS Console:

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

     For example:

     ```
     START BBO5ACR,JOBNAME=BBOS001,ENV=PLEX1.SY1.BBOS001
     ```

   When this server comes back up, it will be running the new version of the application (App v2),

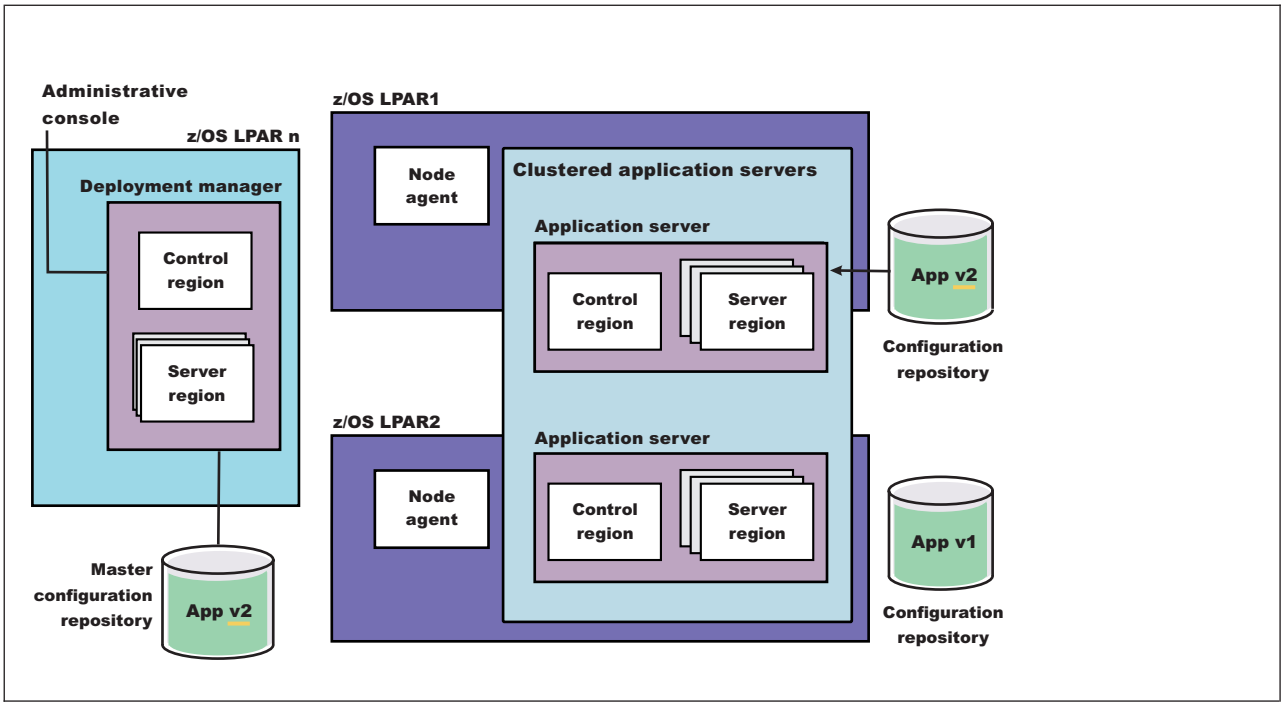


Figure 14. Restart the server on LPAR1. This figure illustrates the completion of the first stage of an application update in a high availability environment.

6. With the new version of the application running on LPAR1, repeat the preceding three steps on the other LPARs in the cluster to update them with the new version of the application. The following figure illustrates what your configuration will look like in a two LPAR cluster.

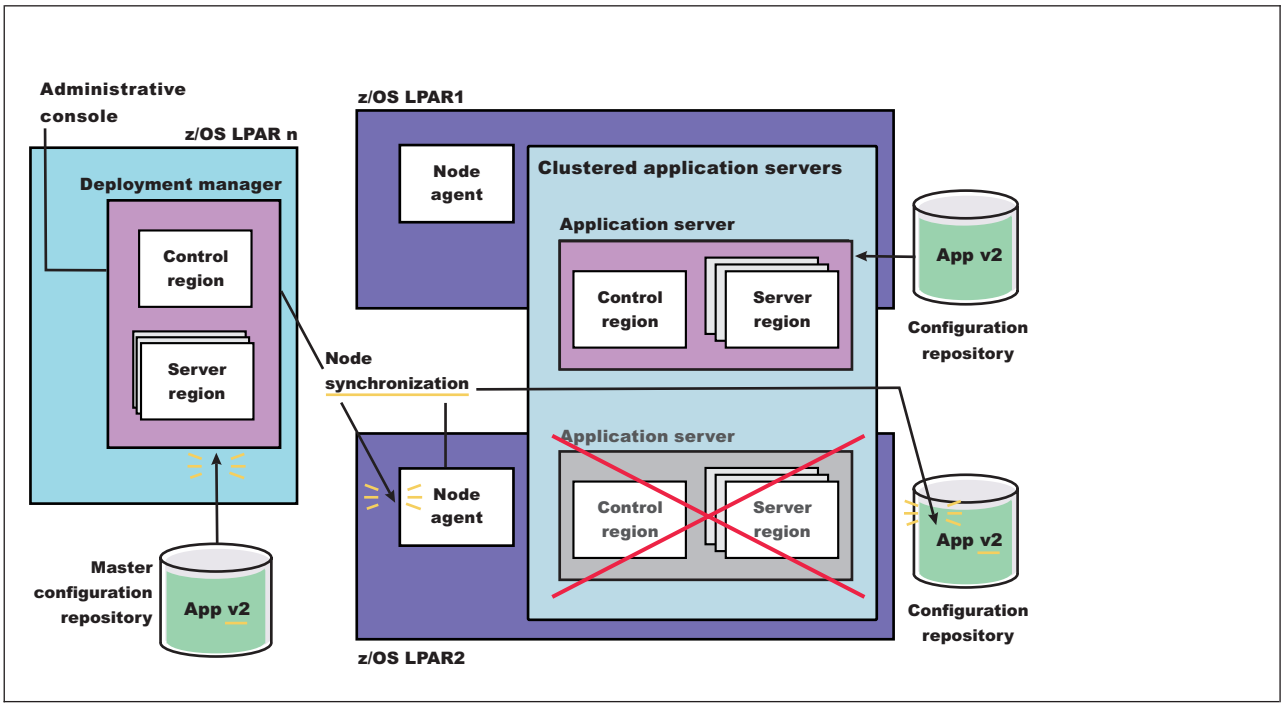


Figure 15. Update the node on LPAR2. This figure illustrates the second stage of an application update in a high availability environment.

The application update process is complete when the new version of the application is running on all of the LPARs in the cluster. The following figure illustrates what a two LPAR cluster will look like after you restart the server on LPAR2.

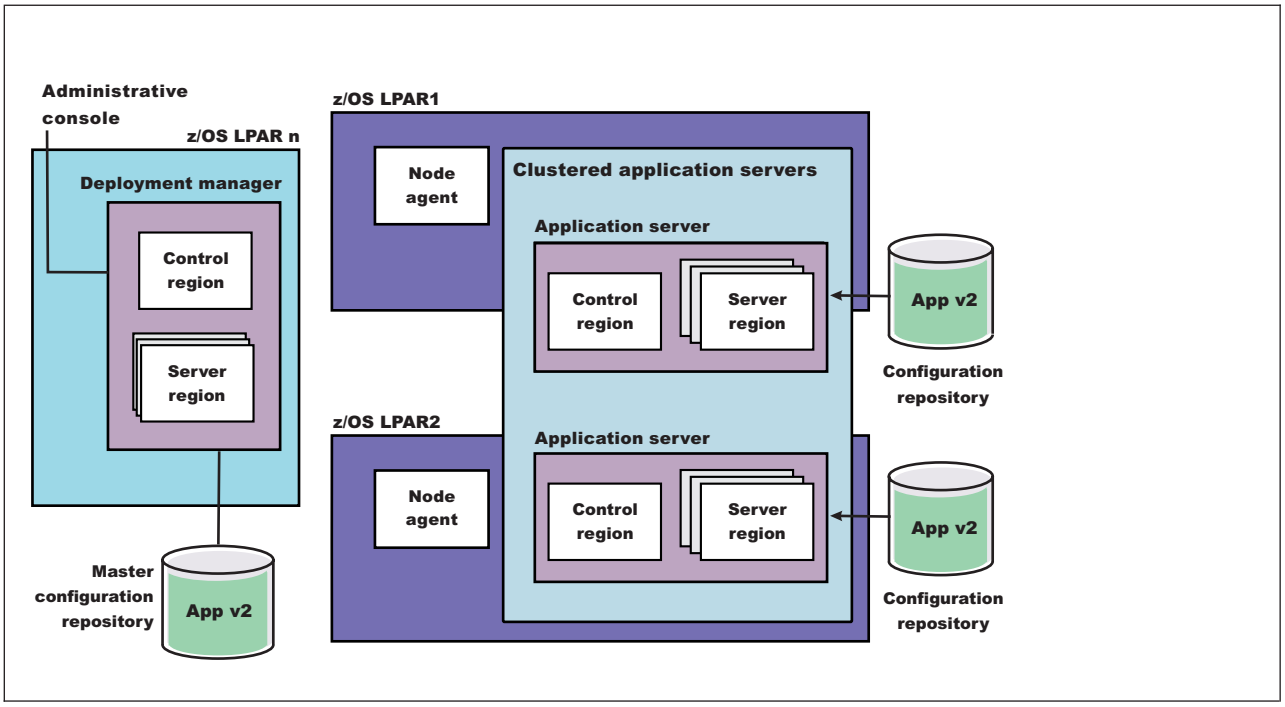


Figure 16. Restart server on LPAR2. This figure illustrates what a two LPAR cluster will look like after you restart the server on LPAR2.

Application update procedure in a high availability environment

Application update involves distributing new application binaries to each of the servers in a cluster during configuration synchronization.

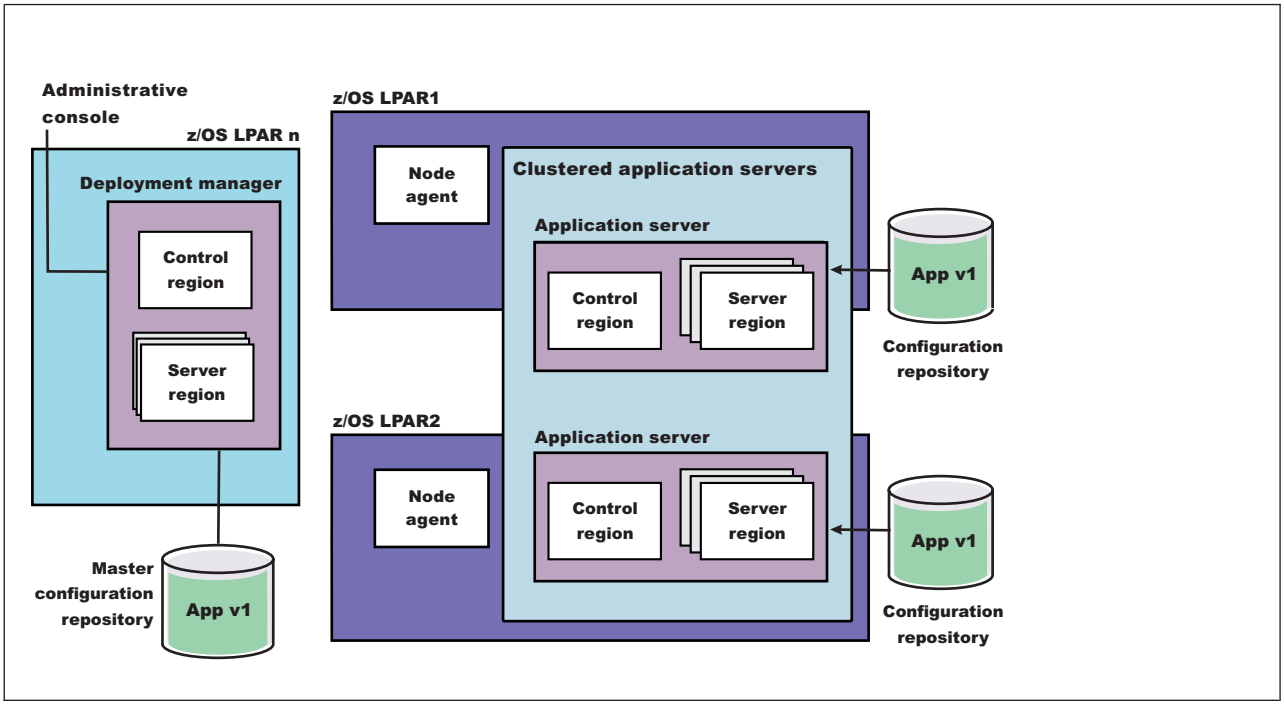


Figure 17. Application steady state - configuration view. This figure illustrates the configuration view of an application in steady state.

This distribution occurs via HTTP and happens even if the servers and the deployment manager are all located on the same LPAR. If the **synchronize with nodes** option is checked when the application update is saved, the synchronization request is sent to each node.

Normally, upon receiving the request, each node asynchronously orchestrates the synchronization process with the deployment manager. During this synchronization process, the application's binaries are downloaded from the deployment manager and stored on the node in the designated location (for example, `installedApps`).

The act of storing the new binaries triggers a configuration change event listener which then stops and restarts the application. Depending on such variables as dispatcher behavior, LPAR weighting, etc., a variance in the order and pace at which each server makes the new application available may be observed.

Because of the concurrent, asynchronous nature of node synchronization, continuous availability of the application being updated is not guaranteed. This is because there is no correspondence between the actual state of the application and the workload routing mechanisms. Client requests may be routed to a server even if that particular application is temporarily unavailable.

In a high availability environment, an application must remain available even during an update process. Therefore, application rollout to each cluster member, as well as workload routing to the cluster members must be carefully controlled to prevent workload from being routed to a cluster member that is undergoing the update process. When these two aspects are carefully controlled, an update can be installed on each cluster member without client requests arriving at a cluster member during the update process.

Setting up a high availability sysplex

- The high availability sysplex must include at least two LPARs. These LPARs should be on separate hardware instances to eliminate hardware 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 DRS or 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. (Using stateful session Enterprise JavaBeans is not a best practice.)

To set up a WebSphere Application Server for z/OS high availability environment:

1. Configure a WebSphere Application Server for z/OS 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 make sure a location service daemon has been defined on each LPAR that has one or more nodes in the same cell. "Modifying z/OS location service daemon settings" on page 153 describes how to define a location service daemon it one is not already defined.
3. Define an application server on each node, and form all of the application servers into a server cluster (see "Creating clusters" on page 333).
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 name for the cell. This IP address enables WLM-balanced routing and fail over between the LPARs for sessionless HTTP requests.

   See tje *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 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. See "Communicating with Web servers" on page 110 for more information.

After you have set up a high availability environment, you can control application rollout and workload routing.

High availability configuration

The objective of any high availability configuration is to eliminate all single points of failure (SPOFs).

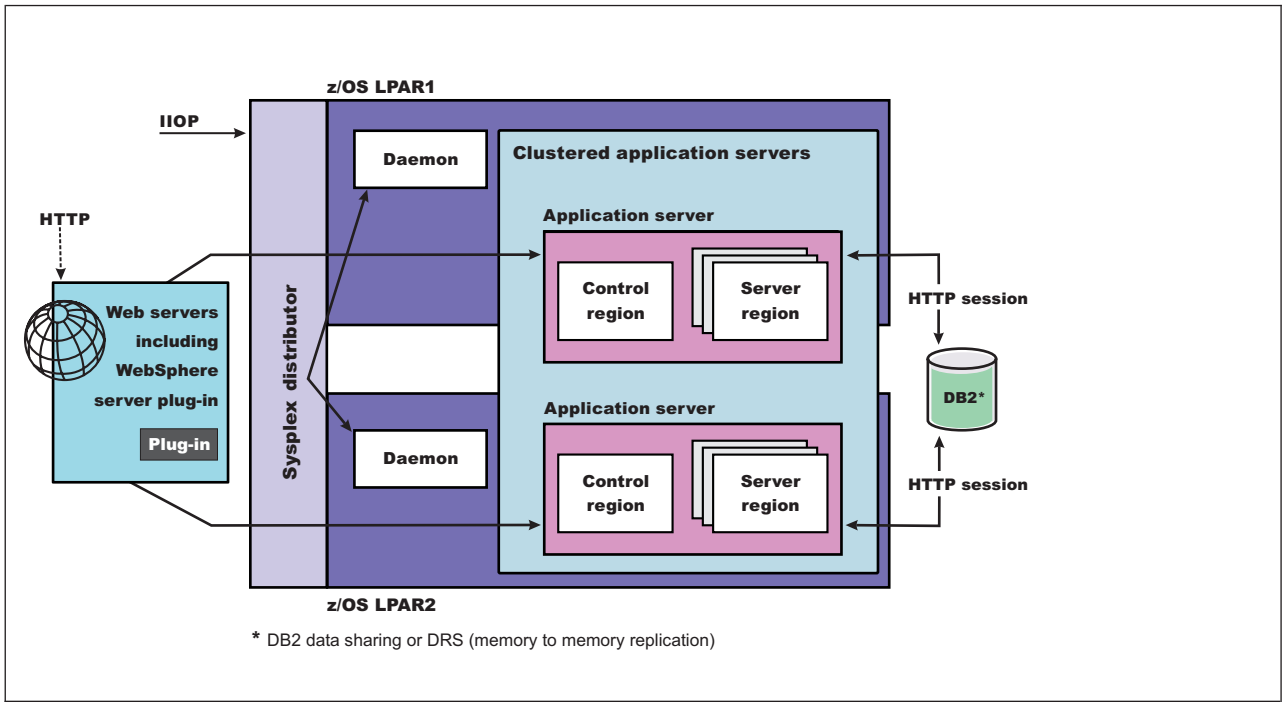


Figure 18. WebSphere Application Server for z/OS High Availability Configuration. This figure illustrates the recommended WebSphere Application Server for z/OS configuration for high availability. The key elements are described in the text that accompanies this figure.

Following are the key elements of a WebSphere Application Server for z/OS high availability configuration:

- Network path redundancy leading up to the Web servers and Applications Servers.
- Redundant Web servers. (There must be at least two LPARs in a high availability sysplex configuration.)
- A highly available sysplex configuration. These LPARs should be on separate hardware instances to eliminate hardware and software Single Points of Failures (SPOFs).
- A WebSphere Application Server for z/OS node on each LPAR that is configured into a Network Deployment cell. The deployment manager server (required, and configured on its own node) can be configured on either LPAR or on a separate LPAR. (The deployment manager server is not depicted in the preceding figure.) Also note, there is a daemon process (WebSphere CORBA Location Service) on each LPAR that has one or more nodes in the same cell.
- An application server defined on each node, and formed into a server cluster with the other application servers in the network.
- A dynamic virtual IP address (DVIPA) defined through the z/OS Sysplex Distributor as the daemon IP name for the cell. This IP address enables WLM-balanced routing and fail over between the LPARs for IIOP requests.
- A dynamic virtual IP address (DVIPA) defined through Sysplex Distributor as the HTTP transport name for the cell. This IP address enables WLM-balanced routing and fail over between the LPARs for sessionless HTTP requests.
- A static IP address is required for each node as an auxiliary HTTP transport name for the cell. This enables directed HTTP routing for sessional HTTP requests.

- A WebSphere Web server plug-in must be installed in each of the Web servers and configured to use the HTTP DVIPA for sessionless requests, and the static IP addresses for sessional requests.
- If using HTTP sessions, session state must be shared between cluster member using DRS or 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. (Using stateful session Enterprise JavaBeans is not a best practice.)

Troubleshooting high availability environment problems

Message HMGR0218I is not displayed after a JVM starts

In a properly set up high availability environment, a high availability manager can reassess the environment it is managing and accept new components as they are added to the environment. For example, when a Java virtual machine (JVM) is added to the infrastructure, a discovery process begins. During startup the JVM tries to contact the other members of the core group. When it finds another running JVM, it initiates a join process with that JVM that determines whether or not the JVM can join the core group. If the new JVM is accepted as a member of the core group, all of the JVMs, including the new one, log message HMGR0218I . This message is also displayed on the administrative console.

Message HMGR0218I indicates the number of application servers in the core group that are currently online. If this message is not displayed after a JVM starts, either a configuration problem or a communication problem has occurred. To fix this situation, verify that the application server is running on a current configuration, by either using the deployment manager to tell the node agent to synchronize, or use the syncNode command to manually perform the synchronization. If the JVM still cannot join the core group, a network configuration problem exists.

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.

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.

Trademarks and service marks

For trademark attribution, visit the IBM Terms of Use Web site (http://www.ibm.com/legal/us/).