WebSphere Application Server V5.0

Runtime Architecture

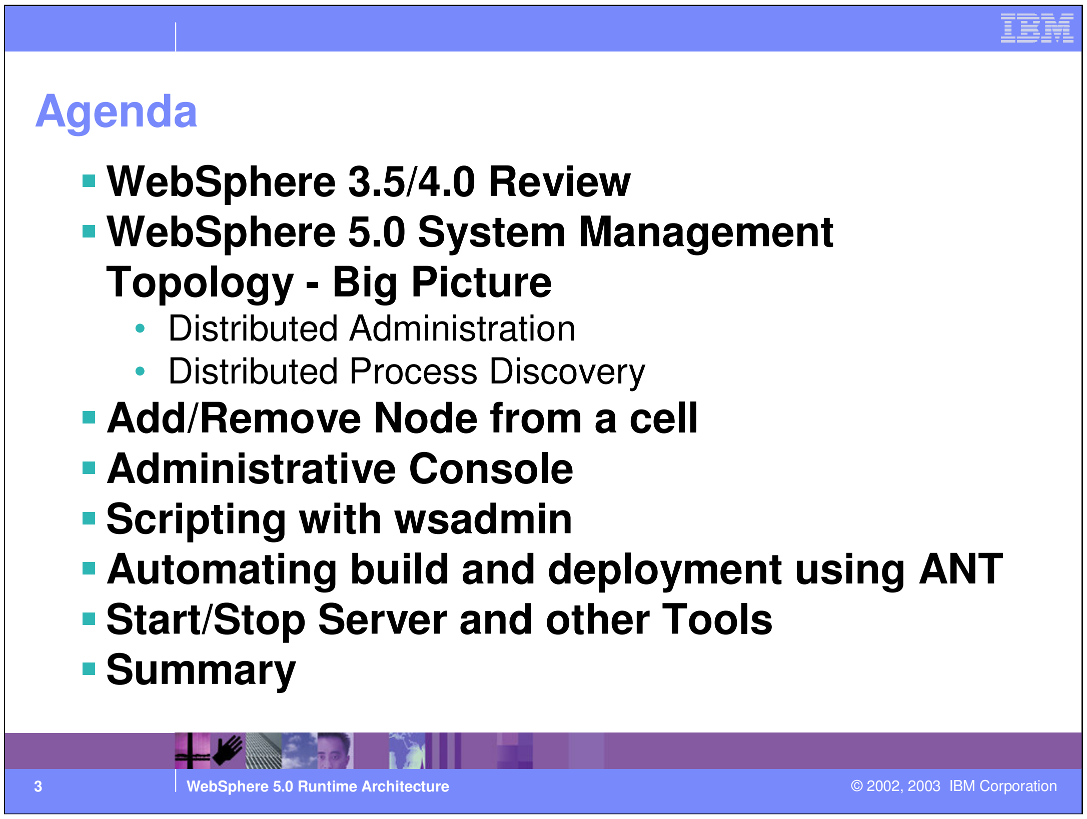
WebSphere software

e business software

Updated 3/08/2003

Objectives

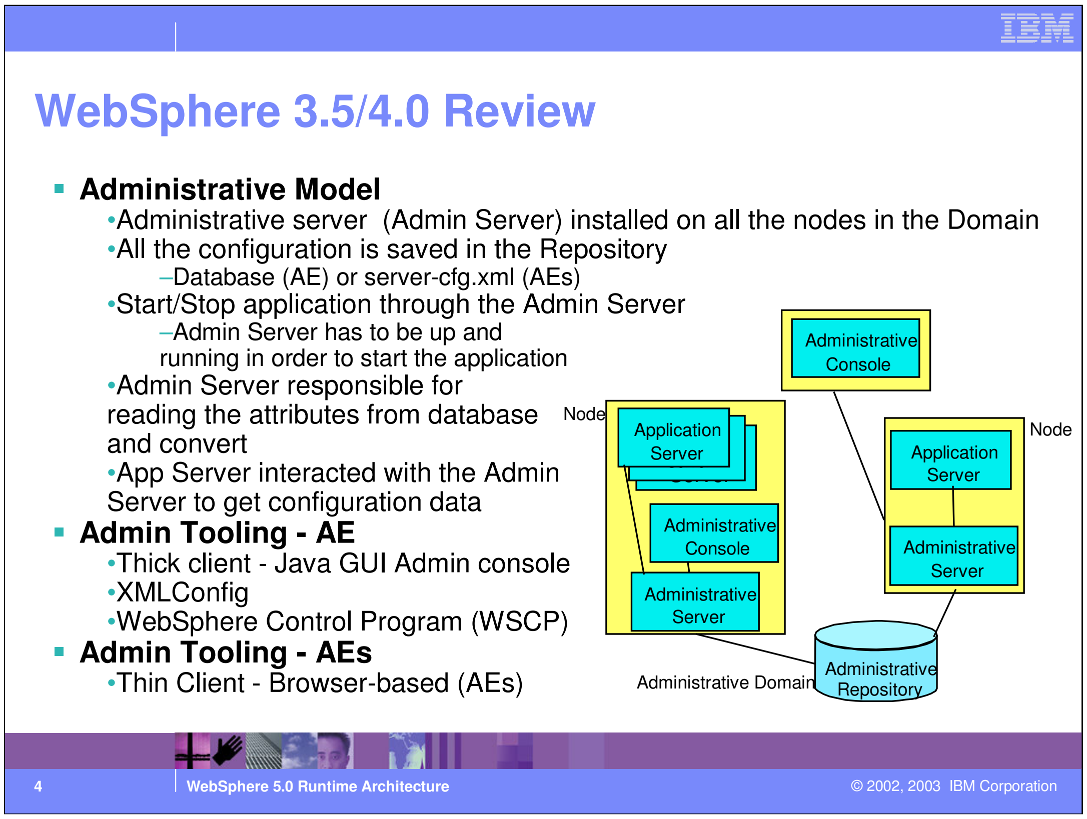
- **You will learn to administer WebSphere Application server Network Deployment**
  - add a node to a cell
  - remove an existing node from a cell
  - You will learn to describe file synchronization
- **You will learn to use the Admin clients**
  - Web-based Admin console
  - wsadmin scripting utility
- **You will learn to write ANT scripts**
- **You will learn to use other command line tools**
  - startserver, stopserver, startManager, stopManager....



Agenda

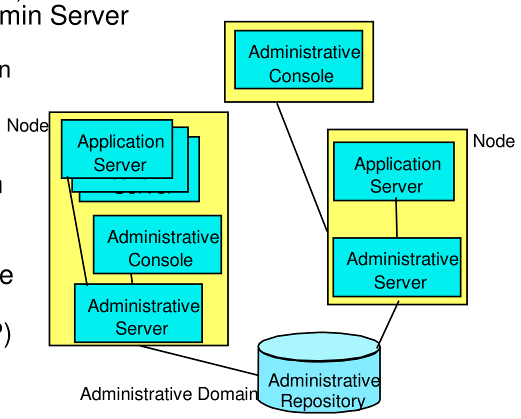
- **WebSphere 3.5/4.0 Review**
- **WebSphere 5.0 System Management Topology - Big Picture**
  - Distributed Administration
  - Distributed Process Discovery
- **Add/Remove Node from a cell**
- **Administrative Console**
- **Scripting with wsadmin**
- **Automating build and deployment using ANT**
- **Start/Stop Server and other Tools**
- **Summary**

3 **WebSphere 5.0 Runtime Architecture** © 2002, 2003 IBM Corporation



WebSphere 3.5/4.0 Review

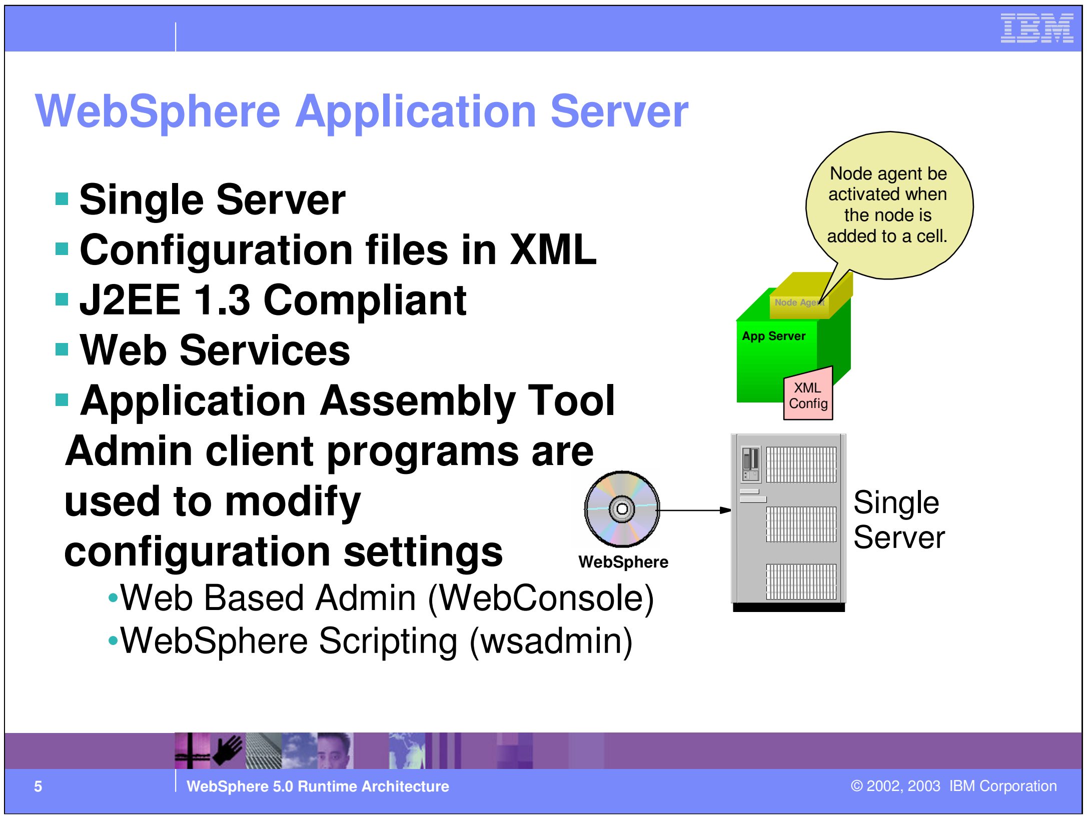
- **Administrative Model**
  - •Administrative server (Admin Server) installed on all the nodes in the Domain
  - •All the configuration is saved in the Repository
    - –Database (AE) or server-cfg.xml (AEs)
  - •Start/Stop application through the Admin Server
    - –Admin Server has to be up and running in order to start the application
  - •Admin Server responsible for reading the attributes from database and convert
  - •App Server interacted with the Admin Server to get configuration data
- **Admin Tooling - AE**
  - •Thick client - Java GUI Admin console
  - •XMLConfig
  - •WebSphere Control Program (WSCP)
- **Admin Tooling - AEs**
  - •Thin Client - Browser-based (AEs)



4 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

In WebSphere 3.5 and WebSphere 4.0 Advanced Edition , the configuration was saved in a database. Also the Admin Server had to be installed on all the nodes in the domain. The Admin Server had to be up and running to start and stop an application. In WebSphere 4.0 Advanced Edition, the Admin Console is a thick Java client. Command line tools such as XMLConfig and WSCP complimented the Admin Console. However a web based admin console was available for the WebSphere 4.0 Single Server Edition.

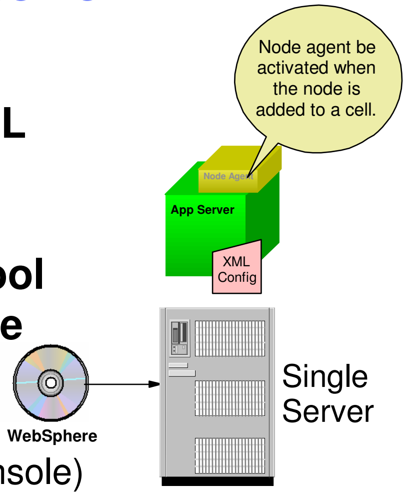
Unlike earlier versions, WebSphere Application Server Version 5 nodes do not use a relational database for the administrative repository for configuration data but store the data in XML files instead. The administrative console Web application lets you update the configuration. You can also use the wsadmin scripting facility in interactive or batch mode to perform any administrative console operation.



WebSphere Application Server

- Single Server
- Configuration files in XML
- J2EE 1.3 Compliant
- Web Services
- Application Assembly Tool

Admin client programs are used to modify configuration settings
- •Web Based Admin (WebConsole)
- •WebSphere Scripting (wsadmin)

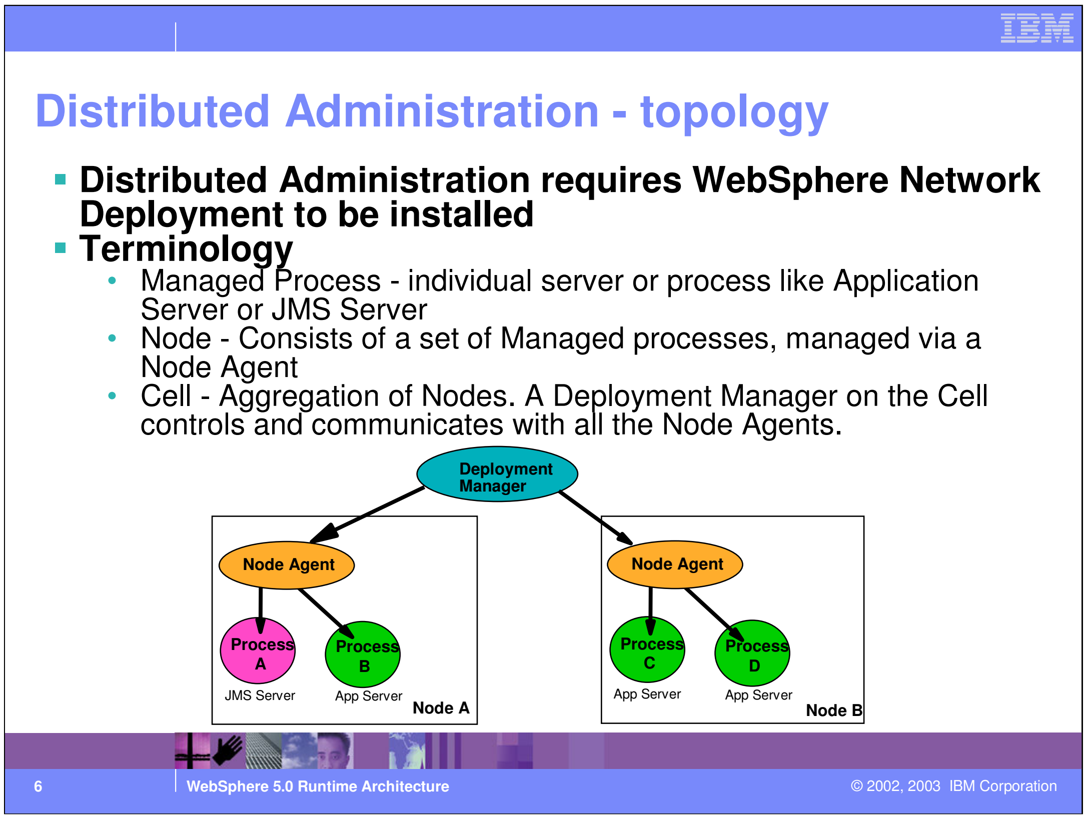


5 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

WebSphere Application Server resides on a single node. That is, each machine holds a separate installation of the product that is unaware of installations on other machines. The configuration files are in XML.

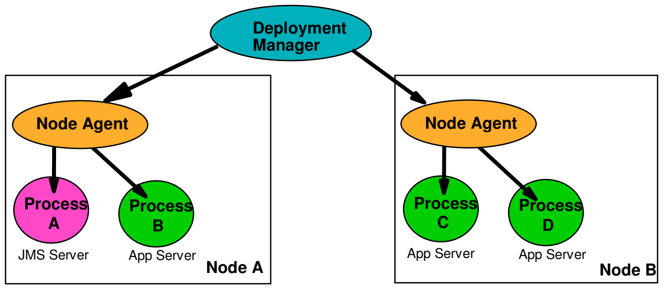
In the Base Application Server, the node agent code is there. The server.xml for the node agent is created when addNode is done - Basically the node agent is not configured for a Base Application server environment.

WebSphere Application Server Network Deployment provides centralized administration of multiple nodes, allowing you to administer nodes on multiple machines.



Distributed Administration - topology

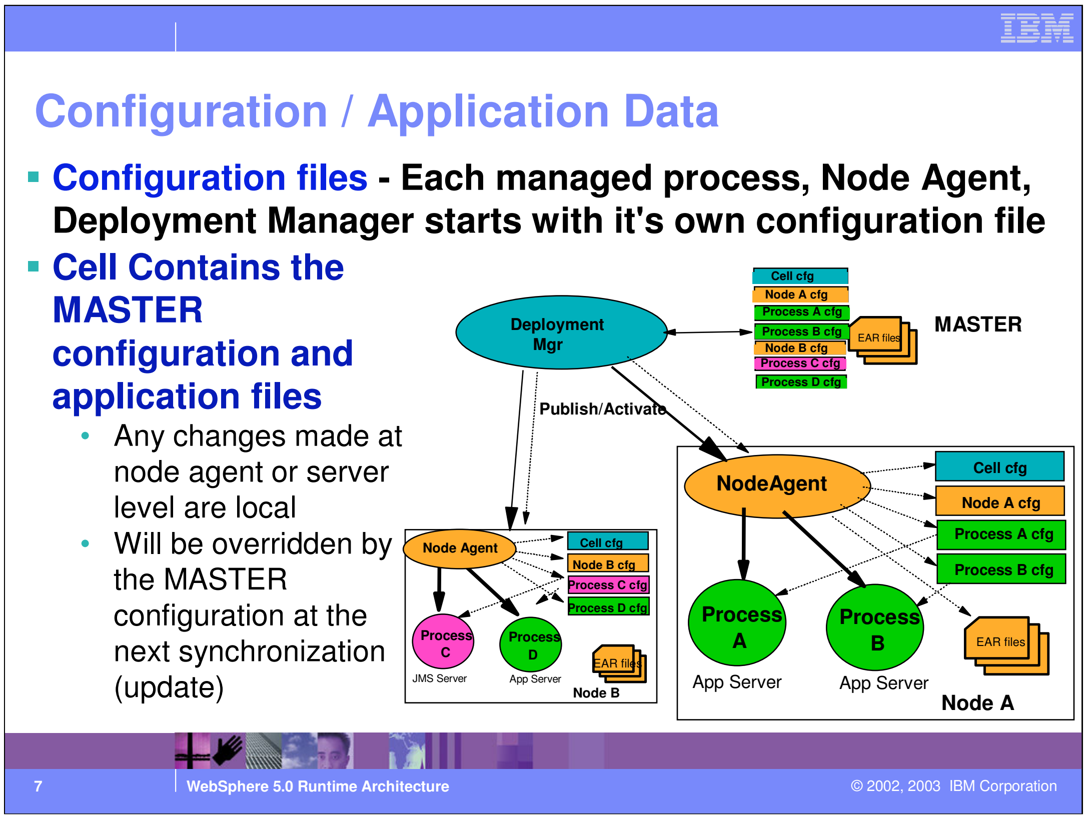
- **Distributed Administration requires WebSphere Network Deployment to be installed**
- **Terminology**
  - Managed Process - individual server or process like Application Server or JMS Server
  - Node - Consists of a set of Managed processes, managed via a Node Agent
  - Cell - Aggregation of Nodes. A Deployment Manager on the Cell controls and communicates with all the Node Agents.



6 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

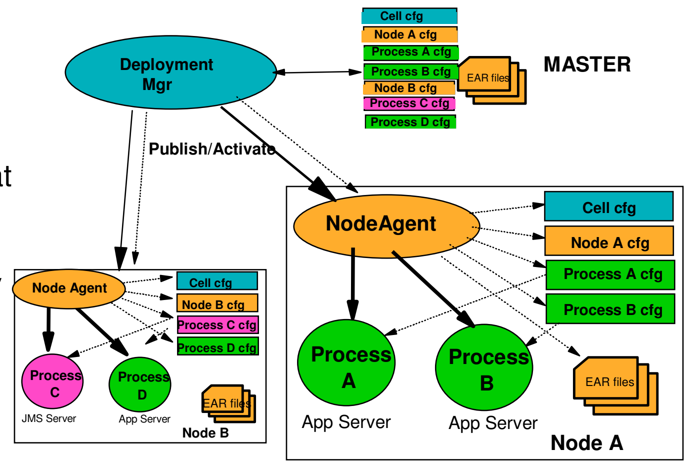
Network Deployment allows you to manage multiple WebSphere Application Server, Version 5 nodes from a single, central location. You can install Network Deployment on any machine in the network to create a network deployment manager cell. The machine on which you install Network Deployment does not require WebSphere Application Server, Version 5 to be installed. Once you have installed and started the network deployment instance, you can use the addNode tool provided with WebSphere Application Server, Version 5 to add a WebSphere Application Server, Version 5 instance (node) to the network deployment cell. Once a node has been added to a network deployment cell, the network deployment manager for the cell assumes responsibility for the configuration of any application server nodes that belong to the cell. The network deployment manager creates configuration files for each WebSphere Application Server node which has been added to its cell.

A node is a set of managed servers on a physical machine in a topology composed of one or more machines. A node contains a WebSphere Application Server installation. A managed server is a single WebSphere Application Server JVM instance, running in its own process. A node cannot span multiple machines, but a machine can have multiple nodes, each with multiple managed servers.



Configuration / Application Data

- **Configuration files - Each managed process, Node Agent, Deployment Manager starts with it's own configuration file**
- **Cell Contains the MASTER configuration and application files**
  - Any changes made at node agent or server level are local
  - Will be overridden by the MASTER configuration at the next synchronization (update)



Configuration files

Each process has all the data (both, configuration and application) to start itself.

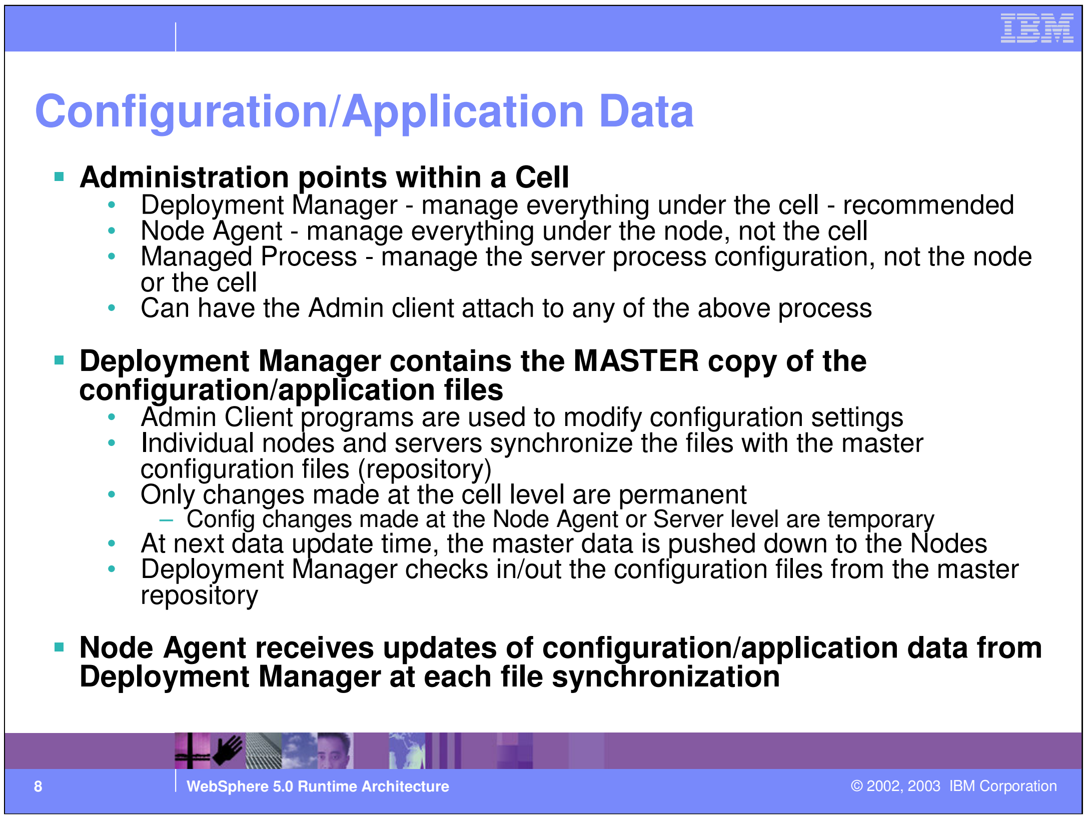
The configuration data is in XML files, and the application data in EARs.

In the WebSphere Network Deployment environment, you can have the admin client attach to any process (Deployment Manager, NodeAgent or individual servers) to make changes to the configuration. For changes to be permanent, t should be done at the Deployment manager level - it contains the master repository. Individual nodes and servers synchronize the files with the master config data.

Config changes made by connecting to a the NodeAgent or Server is temporary until the next File synchronization pushes the master config data from the cell.

Deployment Manager checks in/out the configuration files from the config repository.

NodeAgent request the changes from the Deployment Manager for any new config changes.

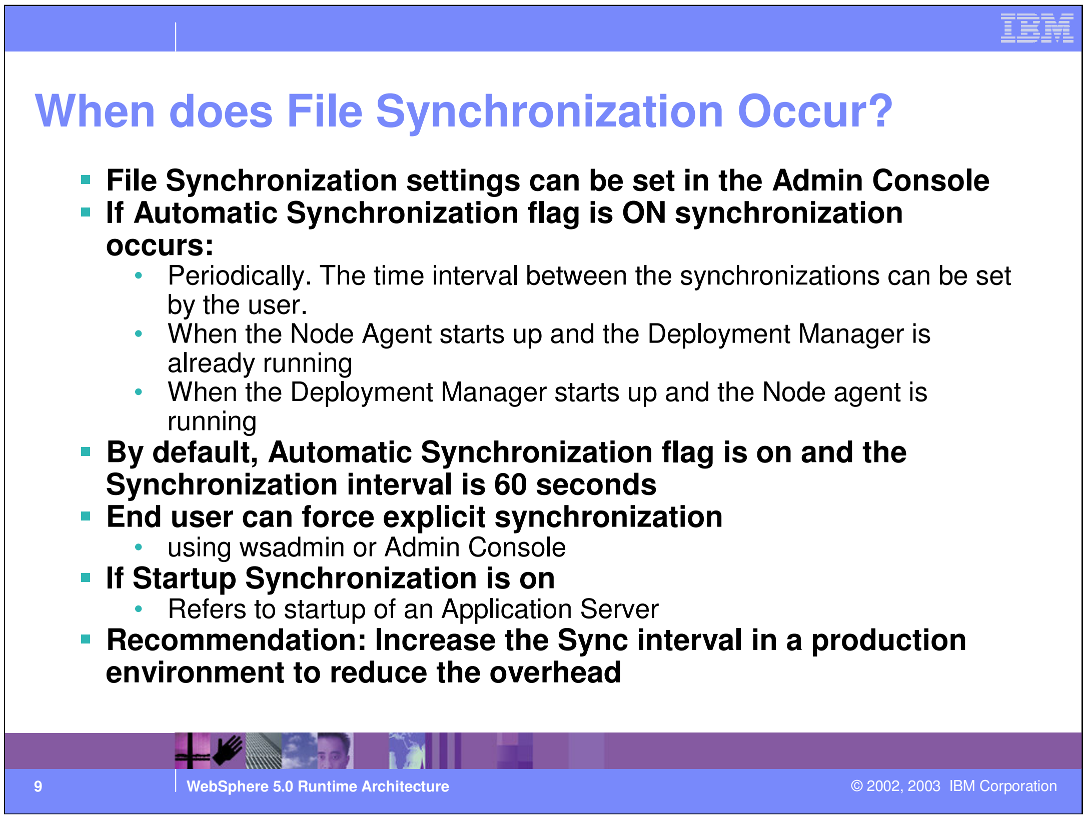


Configuration/Application Data

- **Administration points within a Cell**
  - Deployment Manager - manage everything under the cell - recommended
  - Node Agent - manage everything under the node, not the cell
  - Managed Process - manage the server process configuration, not the node or the cell
  - Can have the Admin client attach to any of the above process
- **Deployment Manager contains the MASTER copy of the configuration/application files**
  - Admin Client programs are used to modify configuration settings
  - Individual nodes and servers synchronize the files with the master configuration files (repository)
  - Only changes made at the cell level are permanent
    - Config changes made at the Node Agent or Server level are temporary
  - At next data update time, the master data is pushed down to the Nodes
  - Deployment Manager checks in/out the configuration files from the master repository
- **Node Agent receives updates of configuration/application data from Deployment Manager at each file synchronization**

8 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

A cell retains master configuration files for each server in each node in the cell. Each node and server also have their own local configuration files. Changes to a local node or server configuration file are temporary, if the server belongs to the cell. While in effect, local changes override cell configurations. Changes at the cell level to server and node configuration files are permanent. Synchronization occurs at designated events, such as when a server starts.



When does File Synchronization Occur?

- **File Synchronization settings can be set in the Admin Console**
- **If Automatic Synchronization flag is ON synchronization occurs:**
  - Periodically. The time interval between the synchronizations can be set by the user.
  - When the Node Agent starts up and the Deployment Manager is already running
  - When the Deployment Manager starts up and the Node agent is running
- **By default, Automatic Synchronization flag is on and the Synchronization interval is 60 seconds**
- **End user can force explicit synchronization**
  - using wsadmin or Admin Console
- **If Startup Synchronization is on**
  - Refers to startup of an Application Server
- **Recommendation: Increase the Sync interval in a production environment to reduce the overhead**

9 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

The configuration synchronization service is the administrative service responsible for keeping the configuration documents that are distributed across the WebSphere Application Server cell up to date.

This service runs in the deployment manager and node agents, and ensures that configuration changes made to the cell repository will be propagated to the appropriate node repositories.

Note: The cell repository is considered the master repository, and configuration changes made to node repositories are not propagated up to the cell.

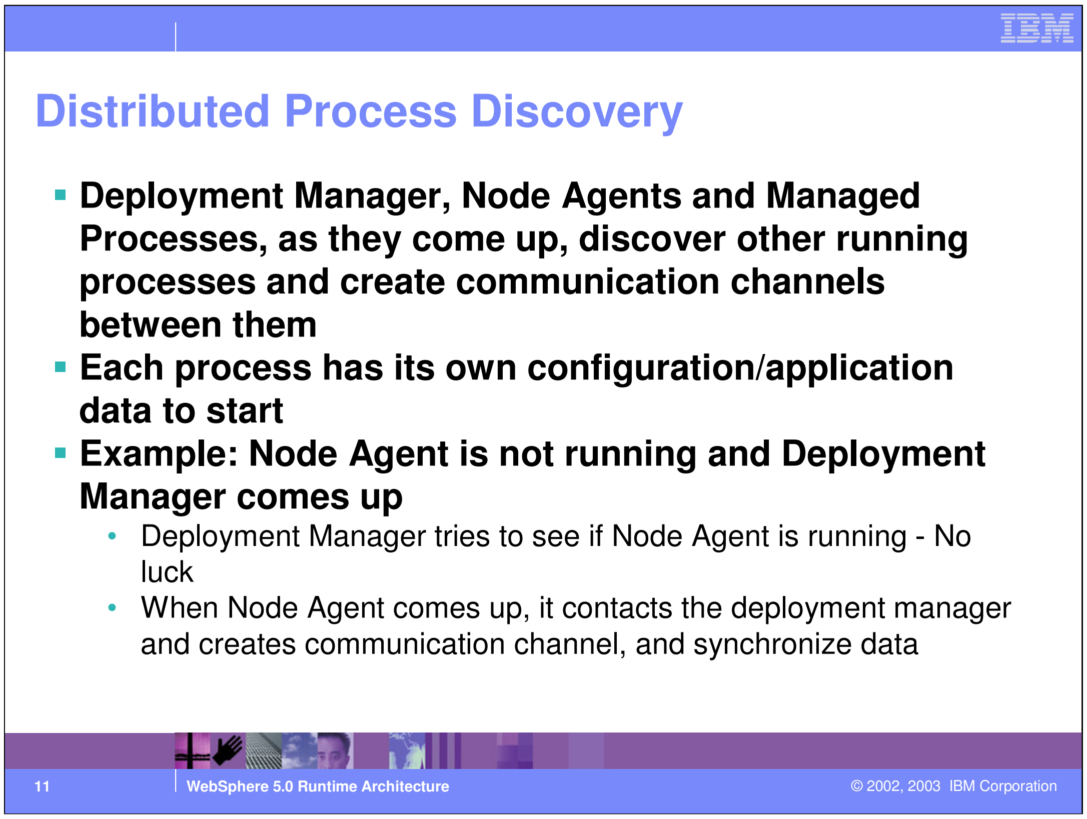
During a synchronization operation a node agent checks with the deployment manager to see if any configuration documents that apply to the node have been updated. New or updated documents are copied to the node repository, and deleted documents are removed from the node repository.

To modify File Synchronization settings, in the admin Console, expand System Administration->Node Agents. In the Work Area pane, click on the NodeAgent. Click on File Synchronization Service.

Automatic Synchronization flag : When enabled, the node agent will automatically contact the Deployment Manager node every SynchInterval to attempt to synchronize the node's configuration repository with the master. This flag is enabled by default and the Synch Interval is set to 60 seconds.

What is Synchronized?

- **Deployment Manager maintains a repository of file names**
- **MBeans are instrumented to notify repository of changes**
  - Changes to XML files through WebSphere Admin tools are propagated
  - Changes to XML files made with generic tools (ie, Notepad) are not recognized unless you manually force a full synch, or use the API to inform the repository of changes.
  - MBean specifics in JMX lecture
- **At synchronization:**
  - files listed in the repository are checked against file system CRC's
  - Only changed files are synchronized to the nodes
  - Changes are marked for synchronization
- **Included in the Deployment Manager's Repository are:**
  - Configuration Files
  - Application Binaries



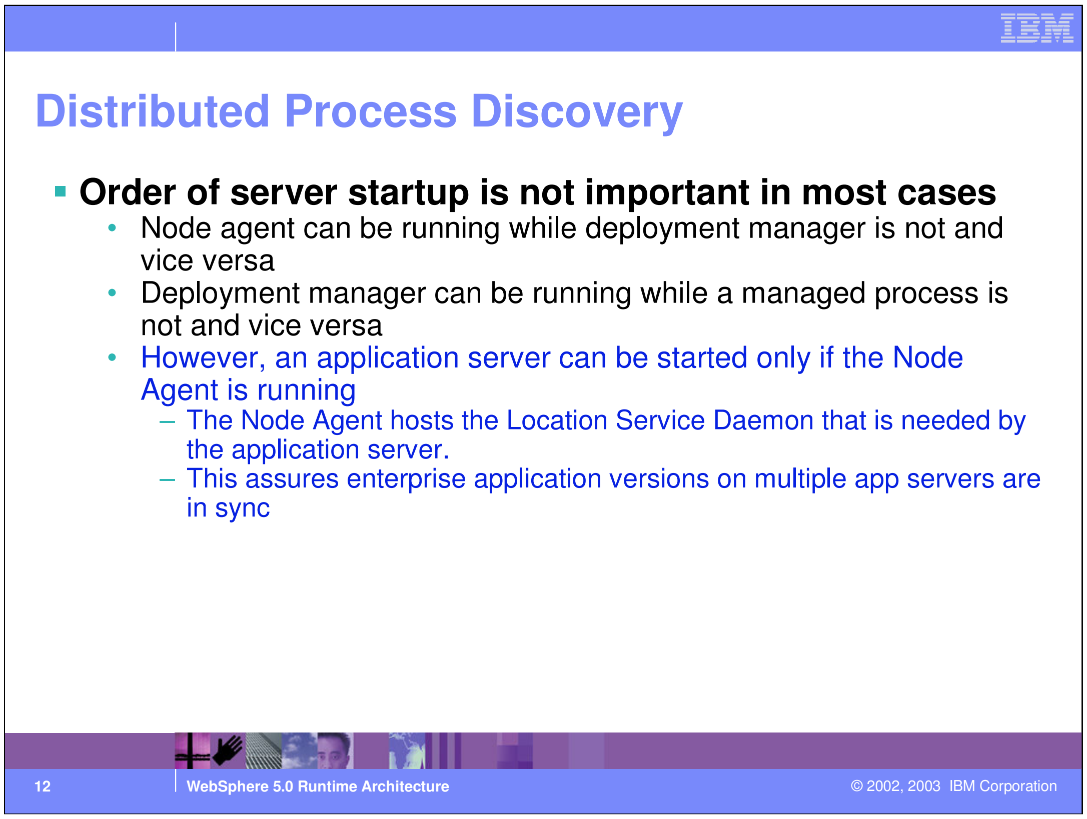
Distributed Process Discovery

- Deployment Manager, Node Agents and Managed Processes, as they come up, discover other running processes and create communication channels between them
- Each process has its own configuration/application data to start
- Example: Node Agent is not running and Deployment Manager comes up
  - Deployment Manager tries to see if Node Agent is running - No luck
  - When Node Agent comes up, it contacts the deployment manager and creates communication channel, and synchronize data

Example: Node Agent is coming up and Managed Process is not running

Node Agent knows all its Managed Processes and will check if all of them are up. If so, it creates the communication channels to those processed.

When Managed Process that was not running comes up, it checks if Node Agent is up and then creates the communication channel.



Distributed Process Discovery

- **Order of server startup is not important in most cases**
  - Node agent can be running while deployment manager is not and vice versa
  - Deployment manager can be running while a managed process is not and vice versa
  - However, an application server can be started only if the Node Agent is running
    - The Node Agent hosts the Location Service Daemon that is needed by the application server.
    - This assures enterprise application versions on multiple app servers are in sync

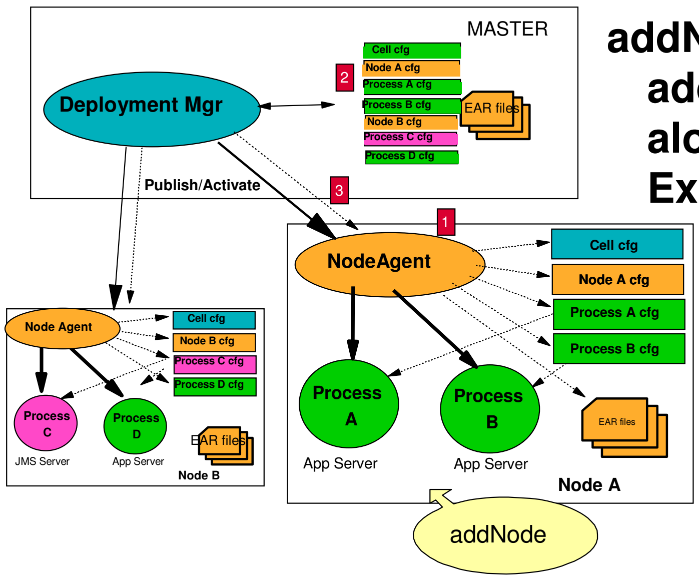
12 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

Node Agents can be started before or after the Deployment Manager.

The only place where start up order is important is with Node Agents and Application Servers. Because the Location Service Daemon runs in the Node Agent process, an application server cannot be started before the Node Agent. One side effect of this limitation is that it guarantees that the node agent has a chance to synchronize before starting application servers - and therefore, all applications installed on multiple nodes will have been updated to the same version when the application server starts up.

Add/Remove Node from a Cell

addNode Utility



addNode Utility - adds a Stand-alone Node to an Existing Cell

1. Creates NodeAgent
2. Copies cfg files to Cell
3. File Sync for the new node
4. Launches the node agent

addNode

- copies BaseApplicationServerCell configuration to a new cell structure that matches the Deployment manager's structure (cell level).

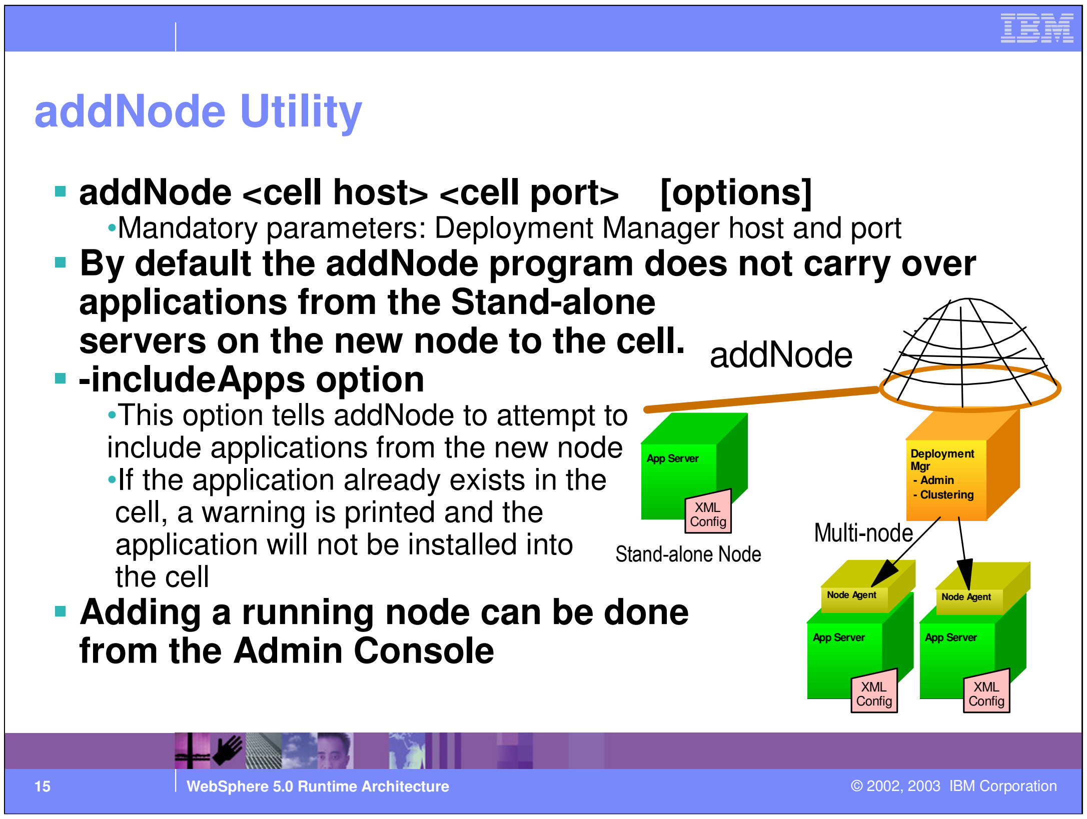
- creates a new node agent definition for the new node.

- calls up to the deployment manager to add the documents from the new node to the cell repository.

- performs the first File Synchronization for the new node (pulls everything back down to the new node).

- launches the node agent for the new node.

The progress of each step is printed as it proceeds.



addNode Utility

- **addNode <cell host> <cell port>   [options]**
  - •Mandatory parameters: Deployment Manager host and port
- **By default the addNode program does not carry over applications from the Stand-alone servers on the new node to the cell.**
- **-includeApps option**
  - •This option tells addNode to attempt to include applications from the new node
  - •If the application already exists in the cell, a warning is printed and the application will not be installed into the cell
- **Adding a running node can be done from the Admin Console**

15 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

Usage: addNode cell_host cell_port [-conntype <type>] [-includeapps] [-noagent] [-quiet] [-nowait] [-trace] [-newtracefile] [-username <uid>] [-password <pwd>] [-?] [-help]

The first two arguments are required. They are the host and JMX port for the Deployment manager.

The options for the addNode command are:

-nowait : Tells the addNode command not to wait for successful initialization of the launched node agent process.

-quiet : Suppresses the progress information that the addNode command prints in normal mode.

-trace : Generates trace information into a file for debugging purposes.

-newtracefile : By default the addNode program appends to the existing trace file. This option causes addNode to overwrite the trace file.

-timeout <seconds> : Specifies the waiting time before node agent initialization times out and returns an error.

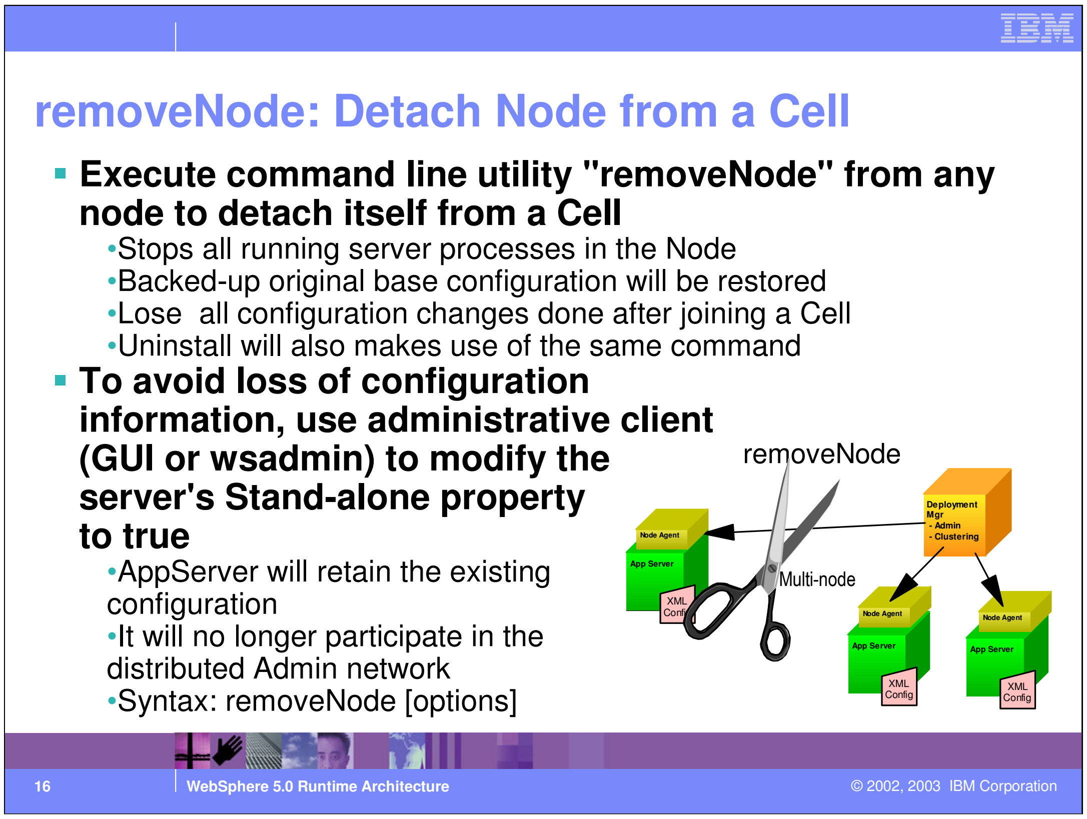
-statusport <portnumber> : Allows an administrator to set the port number for node agent status callback.

-noagent : Tells addNode not to launch the node agent process for the new node.

-username <name> : Specifies the username for authentication if security is enabled.

-password <password> : Specifies the password for authentication if security is enabled.

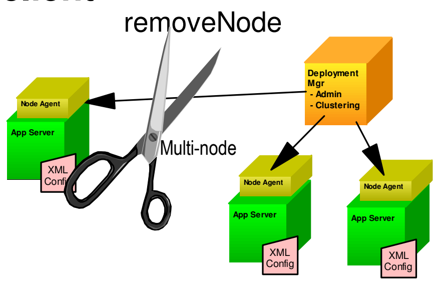
-conntype <type> : Specifies the Java Management Extensions (JMX) connector type to use to connect to the deployment manager. Valid types are Simple Object Access Protocol (SOAP) or Remote Method Invocation (RMI).



IBM

removeNode: Detach Node from a Cell

- **Execute command line utility "removeNode" from any node to detach itself from a Cell**
  - •Stops all running server processes in the Node
  - •Backed-up original base configuration will be restored
  - •Lose all configuration changes done after joining a Cell
  - •Uninstall will also makes use of the same command
- **To avoid loss of configuration information, use administrative client (GUI or wsadmin) to modify the server's Stand-alone property to true**
  - •AppServer will retain the existing configuration
  - •It will no longer participate in the distributed Admin network
  - •Syntax: removeNode [options]



16 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

The removeNode.bat/sh utility removes a node from the cell configuration. Uninstall will call this utility. And users can just use it from the command line to get back to the original base app server environment they started from. It also stops all currently running app servers (that are running as part of an ND network).

Note that they will go back to a previous cell configuration (the one they started from as a Base app package), so they will "lose" any configuration they had in the ND cell. If this is not what is desired, there is a "stand-alone" property in the server.xml file (and available through the admin clients - wsadmin and webui) that can be set to "true" for any ND server and it will no longer participate in the distributed admin network (i.e. it behaves like a single base app server, but retains its current configuration).

To change or view the "stand-alone" setting , click Servers > Manage Application Servers in the console navigation tree, click an application server, and then click Administrative Services.

Stand-alone Specifies whether the server process is a participant in a Network Deployment cell or not. If true, the server does not participate in distributed administration. If false, the server participates in the Network Deployment system.

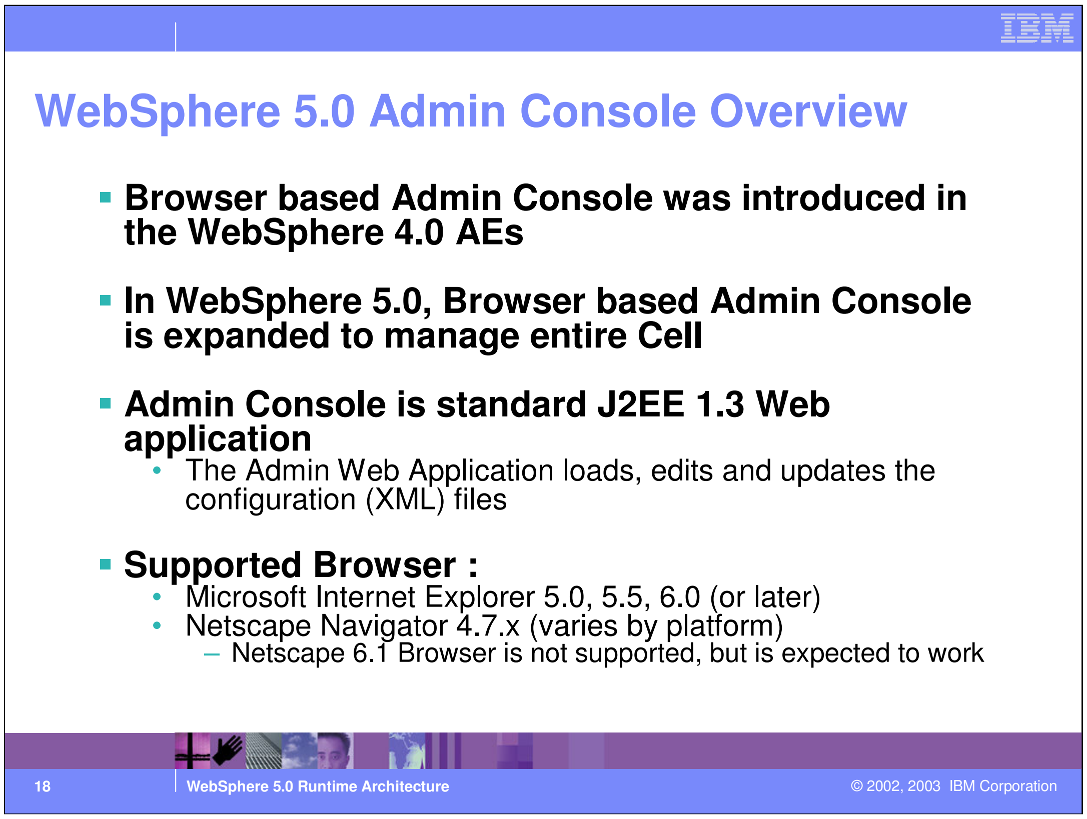
The default value for base WebSphere Application Server installations is true. When addNode runs to incorporate the server into a Network Deployment cell, the values switches to false. So removeNode is a more wide ranging switch (removes the node from the cell and switches back to the original stand-alone base cell), and changing the stand-alone admin property just takes the one server out of the network.

Syntax:: removeNode [options]

The options for the removeNode command are:

-quiet : Suppresses the progress information the removeNode command prints in normal mode.

Administrative Console (Admin Console)



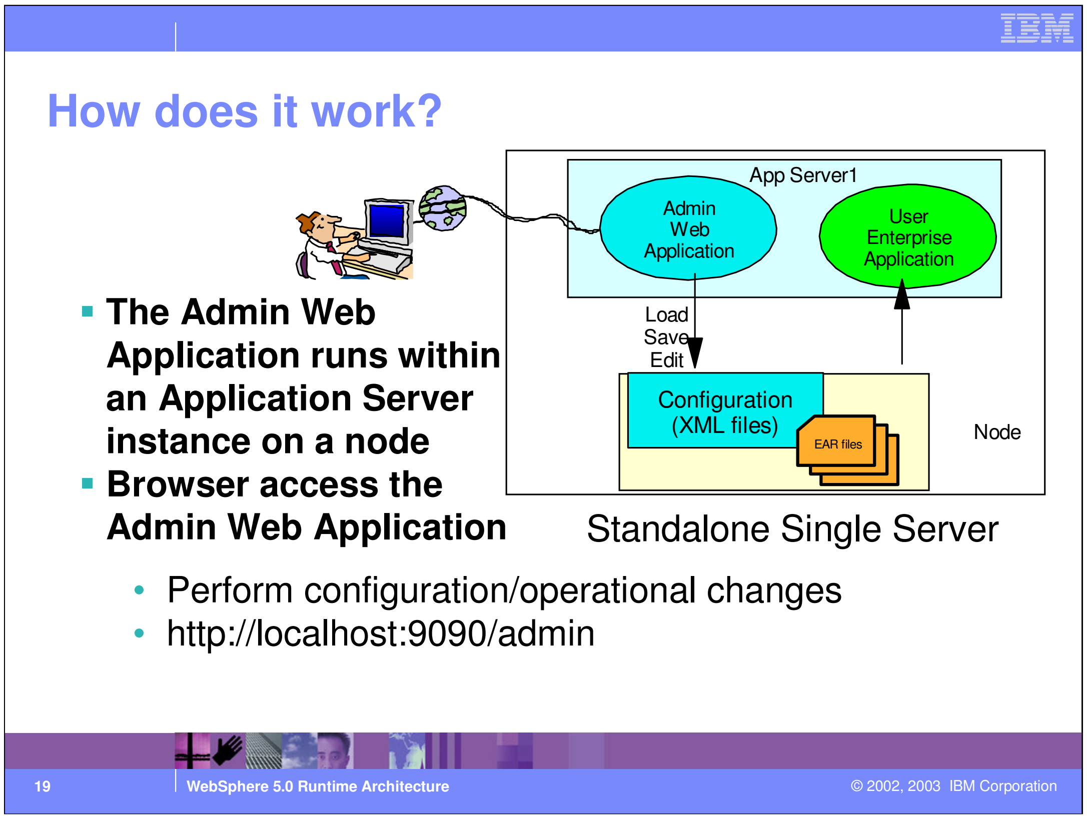
WebSphere 5.0 Admin Console Overview

- **Browser based Admin Console was introduced in the WebSphere 4.0 AEs**
- **In WebSphere 5.0, Browser based Admin Console is expanded to manage entire Cell**
- **Admin Console is standard J2EE 1.3 Web application**
  - The Admin Web Application loads, edits and updates the configuration (XML) files
- **Supported Browser :**
  - Microsoft Internet Explorer 5.0, 5.5, 6.0 (or later)
  - Netscape Navigator 4.7.x (varies by platform)
    - Netscape 6.1 Browser is not supported, but is expected to work

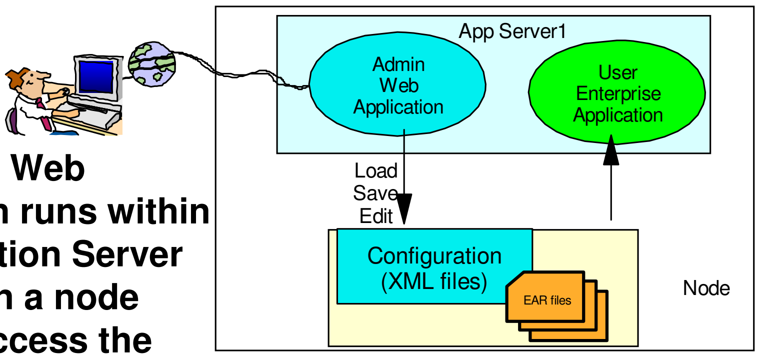
18 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

The WebSphere administrative console is a graphical, Web-based tool that you use to manage the IBM WebSphere Application Server administrative server. The administrative console supports a full range of product administrative activities.

WebSphere release 5.0 Administrative web application build upon the Admin web application architecture and functions which were introduced in the WebSphere 4.0 AEd/s offerings. In the 4.0 time frame, the scope of the Administrative web application was designed to accommodate the requirements necessary to support the configuration capabilities for a single WebSphere Domain, Node, and Application Server, while transitioning over to a new XML-based configuration architecture.



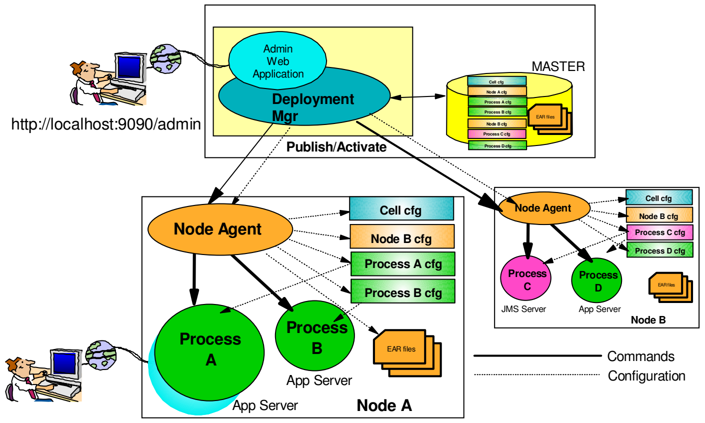
How does it work?



Standalone Single Server

- **The Admin Web Application runs within an Application Server instance on a node**
- **Browser access the Admin Web Application**
  - Perform configuration/operational changes
  - http://localhost:9090/admin

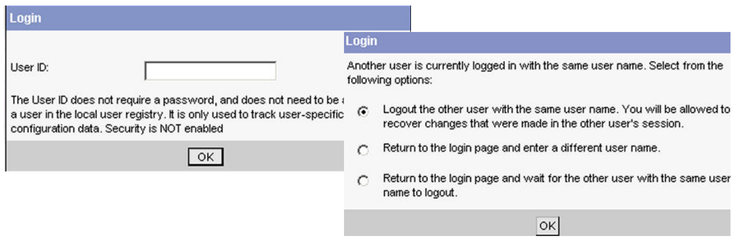
Admin Console in a Multi-node Topology



– In a complex deployment, the Admin Web Application may be configured on a Stand-alone Application Server. Any changes made are local.

Starting the Admin Console

- **Admin Console Application gets installed during WebSphere 5.0 installation**
- **Start the server process on which the Admin Console application is installed**
  - •In the case of Base Application server, issue: startserver server1
- **Access via web Browser "http://<hostname>:9090/admin/"**
- **No Password is required if the Global security is disabled**
- **UserName is used to track and save user-specific configuration data**
  - •Will be able to recover from unsaved previous session changes
  - •user Workspace : <was50-root>/wstemp/USER/workspace


Login

User ID:

The User ID does not require a password, and does not need to be a user in the local user registry. It is only used to track user-specific configuration data. Security is NOT enabled

Login

Another user is currently logged in with the same user name. Select from the following options:

[✓ Logout the other user with the same user name. You will be allowed to recover changes that were made in the other user's session.]

Return to the login page and enter a different user name.

Return to the login page and wait for the other user with the same user name to logout.

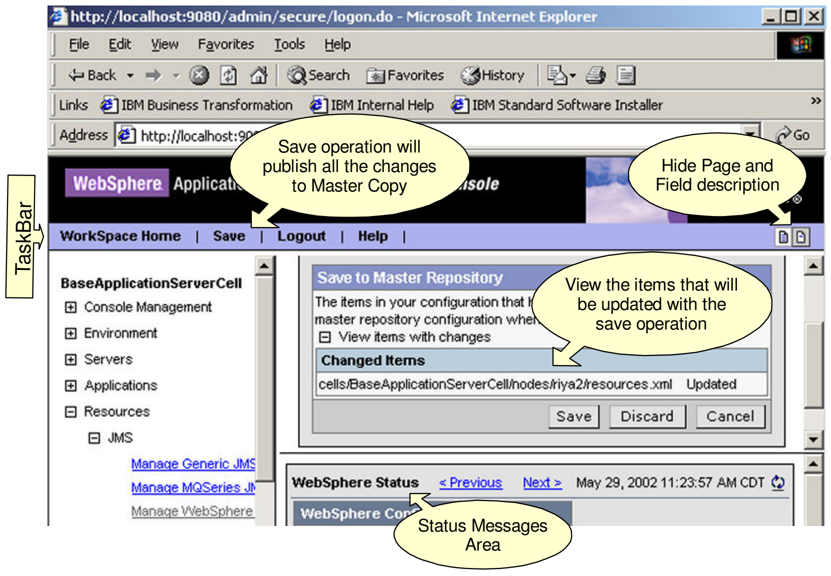
To access the Web-based administrative console, the WebSphere Application Server Web server and application server must be running. After you point a Web browser at the URL for the administrative console, enter your user ID and, if needed, a password on a Login page.

For the Base Application server, Admin Console application is in adminConsole.ear file

Binary : <was_root>/installedApps/<nodename>/adminConsole.ear

Configuration <was_root>\config\cells\<nodename>\applications\adminconsole

WebSphere 5.0 Administrative Console


TaskBar
Save operation will publish all the changes to Master Copy
Hide Page and Field description
View the items that will be updated with the save operation
Status Messages Area
WorkSpace Home | Save | Logout | Help
BaseApplicationServerCell
Console Management
Environment
Servers
Applications
Resources
JMS
Save to Master Repository
View items with changes
Changed Items
Save
Discard
Cancel
WebSphere Status
< Previous
Next >

http://localhost:9080/admin/secure/logon.do - Microsoft Internet Explorer

The items in your configuration that h… master repository configuration when…

cells/BaseApplicationServerCell/nodes/riya2/resources.xml Updated

May 29, 2002 11:23:57 AM CDT

The WebSphere administrative console is a graphical, Web-based tool that you use to manage the IBM WebSphere Application Server administrative server. The administrative console supports a full range of product administrative activities.

The console has these main areas:

The taskbar , The cell tree view , The workspace , The status messages area

You can resize these areas as desired.

The taskbar.

The taskbar provides a way for you to return to the console Home page, to save changes that you have made to administrative configurations, to access product information and to log out of the console.

The cell tree view.

You use the tree view on the left side of the console to survey, select, and manage components in a WebSphere administrative cell.

The workspace

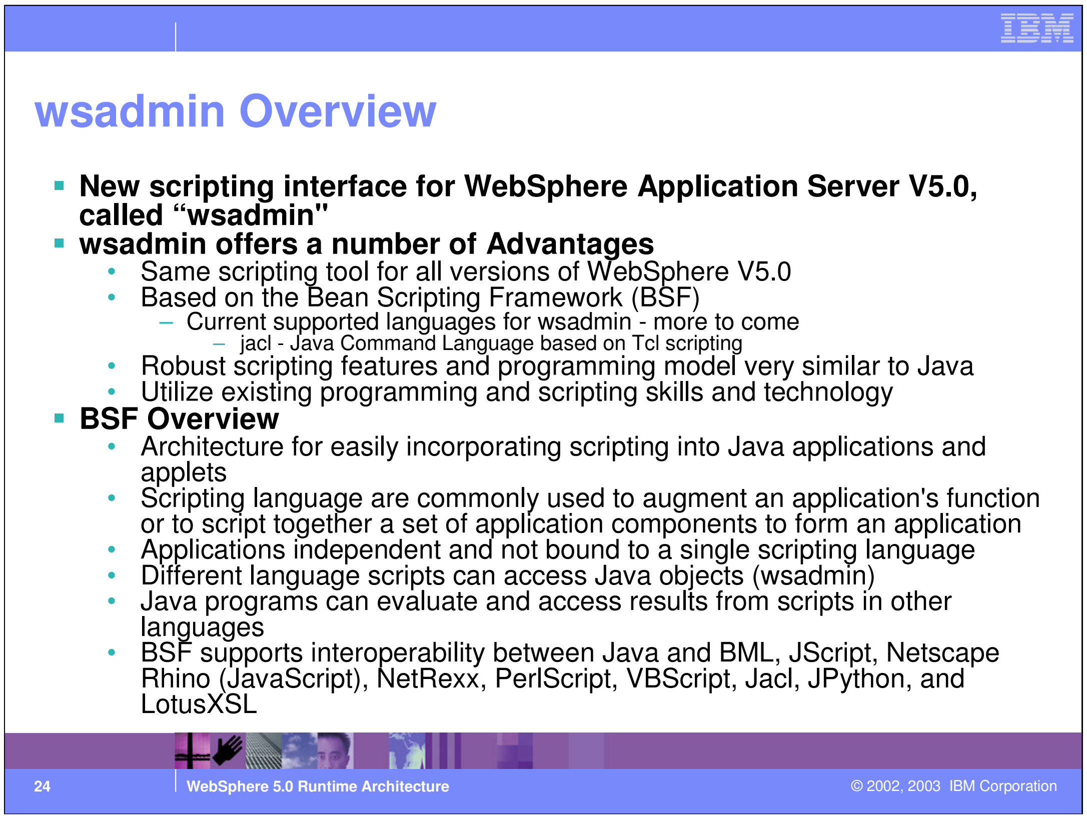
The console workspace provides links to pages that provide step-by-step instructions on installing application, updating applications, and creating application servers. The console also provides a search function for locating and viewing resource objects and some information about adjusting the console to meet your needs.

The status messages area.

The messages area at the bottom of the console lists messages returned by the WebSphere Administrative Server as well as messages about events such as successful completion and fatal errors.

You can customize the contents of the message list and limit the message log size in the Preferences settings.

Scripting with wsadmin



wsadmin Overview

- **New scripting interface for WebSphere Application Server V5.0, called “wsadmin"**
- **wsadmin offers a number of Advantages**
  - Same scripting tool for all versions of WebSphere V5.0
  - Based on the Bean Scripting Framework (BSF)
    - Current supported languages for wsadmin - more to come
      - jacl - Java Command Language based on Tcl scripting
  - Robust scripting features and programming model very similar to Java
  - Utilize existing programming and scripting skills and technology
- **BSF Overview**
  - Architecture for easily incorporating scripting into Java applications and applets
  - Scripting language are commonly used to augment an application's function or to script together a set of application components to form an application
  - Applications independent and not bound to a single scripting language
  - Different language scripts can access Java objects (wsadmin)
  - Java programs can evaluate and access results from scripts in other languages
  - BSF supports interoperability between Java and BML, JScript, Netscape Rhino (JavaScript), NetRexx, PerlScript, VBScript, Jacl, JPython, and LotusXSL

24 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

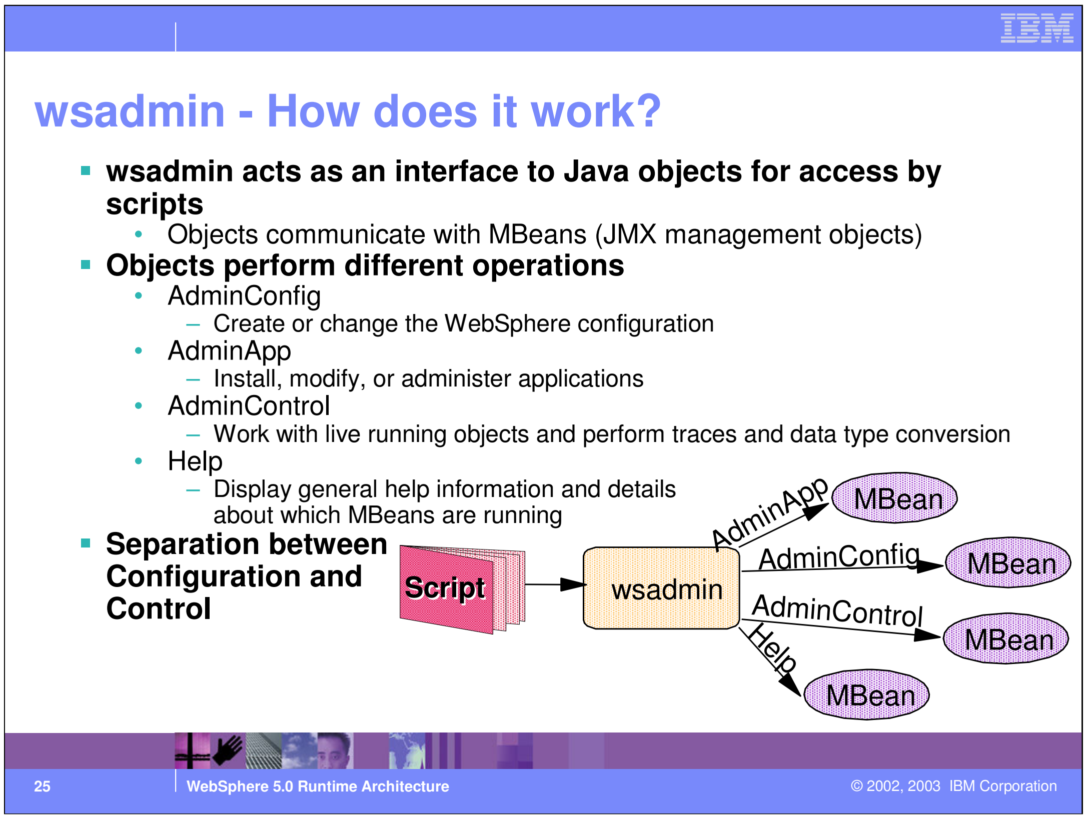
wsadmin is the new scripting interface for WebSphere replacing WSCP and XMLConfig. It is based on the Bean Scripting Framework which offers a number of advantages. The first one being that the same scripting interface will be available for all versions of WebSphere. This is different from WebSphere v4.0 where WSCP and XMLConfig were only available with WebSphere Advanced Edition (AE) and SEAppInstall was only available in WebSphere Advanced Single Server Edition (AEs).

For WebSphere V5.0, Jacl will be supported, with other languages supported in future versions. Jacl is based on the scripting language Tcl. More information about Tcl is available at http://www.tcl.tk/scripting/.

Some of the major advantages of using Jacl and wsadmin is that Jacl offers may robust features and prebuilt functions. The programming model for Jacl is easy for Java developers to use and allows customers to utilize their existing investment in scripting skills.

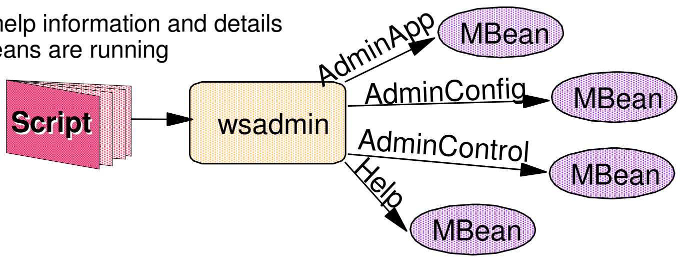
In WebSphere v4.0 there was limited support for some of the Tcl commands. With wsadmin, there is full support of all the Tcl commands.

The advantage of BSF is that scripts are not tied to a specific application or implementation. Any number of languages can access Java objects and interoperate with each other.



wsadmin - How does it work?

- **wsadmin acts as an interface to Java objects for access by scripts**
  - Objects communicate with MBeans (JMX management objects)
- **Objects perform different operations**
  - AdminConfig
    - Create or change the WebSphere configuration
  - AdminApp
    - Install, modify, or administer applications
  - AdminControl
    - Work with live running objects and perform traces and data type conversion
  - Help
    - Display general help information and details about which MBeans are running
- **Separation between Configuration and Control**



25 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

The JMX specification defines an interface called MBeanServer for communicating with MBeans -- the scripting interface called AdminControl is based on this interface.

wsadmin uses the same interface as a user would using the Web Console to make configuration changes and control the WebSphere server.

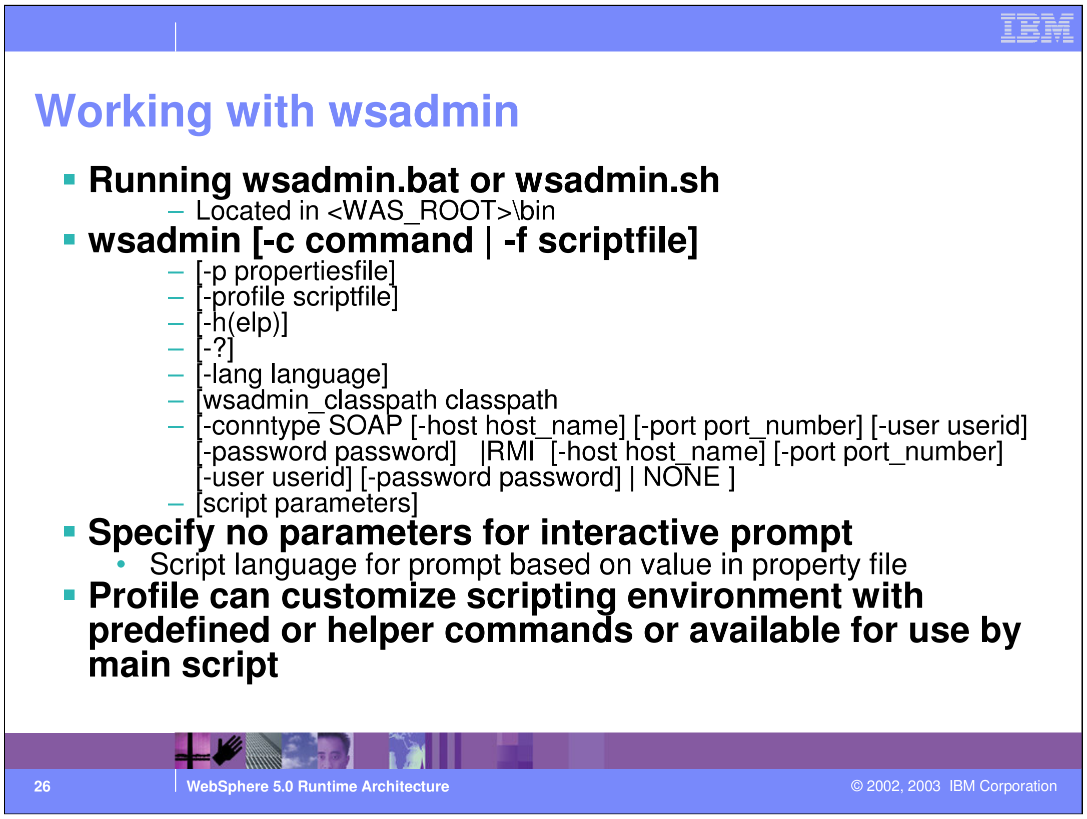
With the different commands available for wsadmin there is a clear distinction between working with configuration settings and objects running on the server. This distinction should ease the understanding of what is being changed when working with wsadmin.

What about wscp

5.0 has new objects than 4.0. So, wscp scripts will need to change

Instead of providing a wscp wrapper which may still involve some changes on the customer side, it was decided that a better alternative would be to provide lots of examples that makes it easy for the customer to migrate

TCL language still being used in wsadmin, so learning curve should not be high for wscp script writers

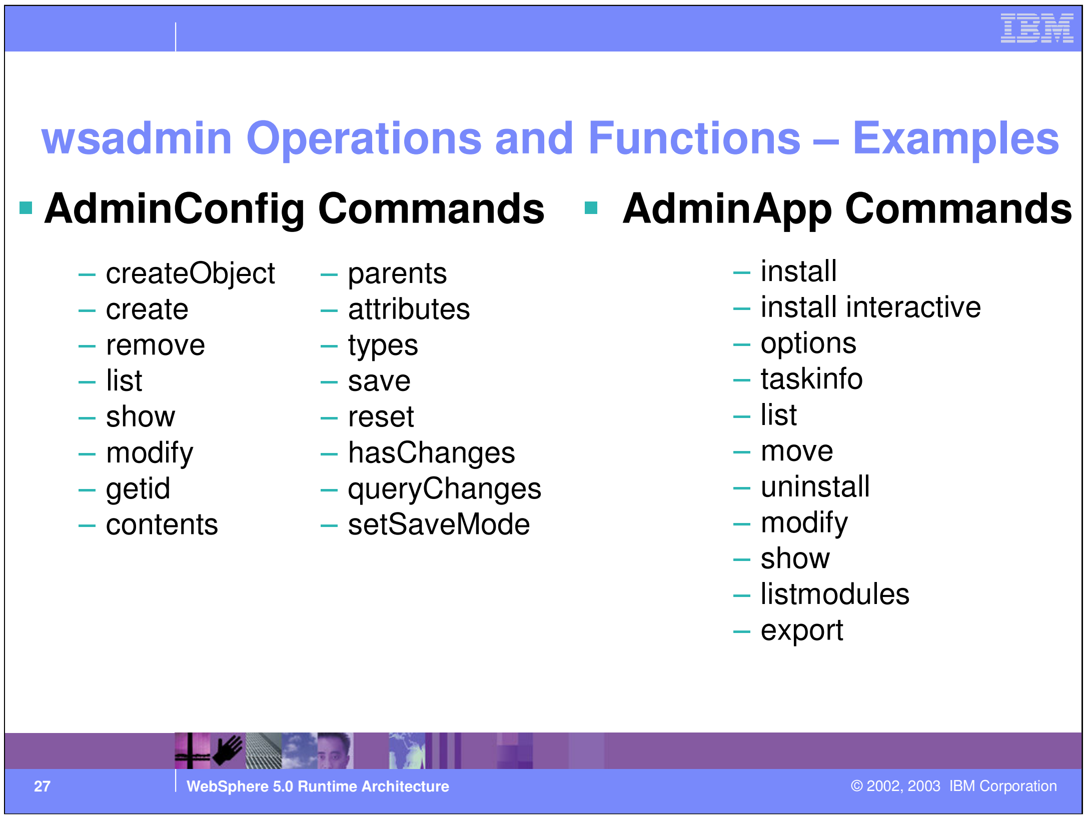


Working with wsadmin

- **Running wsadmin.bat or wsadmin.sh**
  - Located in <WAS_ROOT>\bin
- **wsadmin [-c command | -f scriptfile]**
  - [-p propertiesfile]
  - [-profile scriptfile]
  - [-h(elp)]
  - [-?]
  - [-lang language]
  - [wsadmin_classpath classpath
  - [-conntype SOAP [-host host_name] [-port port_number] [-user userid] [-password password] |RMI [-host host_name] [-port port_number] [-user userid] [-password password] | NONE]
  - [script parameters]
- **Specify no parameters for interactive prompt**
  - Script language for prompt based on value in property file
- **Profile can customize scripting environment with predefined or helper commands or available for use by main script**

26 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

There are basically three ways in which wsadmin can be used. The first way is by specifying wsadmin and the name of the command with the "c" option. You can also specify "f" and the name of script file which will contain a number of individual commands. Finally, you can just call wsadmin and have an interactive prompt started.



wsadmin Operations and Functions – Examples

- **AdminConfig Commands**
  - createObject
  - create
  - remove
  - list
  - show
  - modify
  - getid
  - contents
  - parents
  - attributes
  - types
  - save
  - reset
  - hasChanges
  - queryChanges
  - setSaveMode
- **AdminApp Commands**
  - install
  - install interactive
  - options
  - taskinfo
  - list
  - move
  - uninstall
  - modify
  - show
  - listmodules
  - export

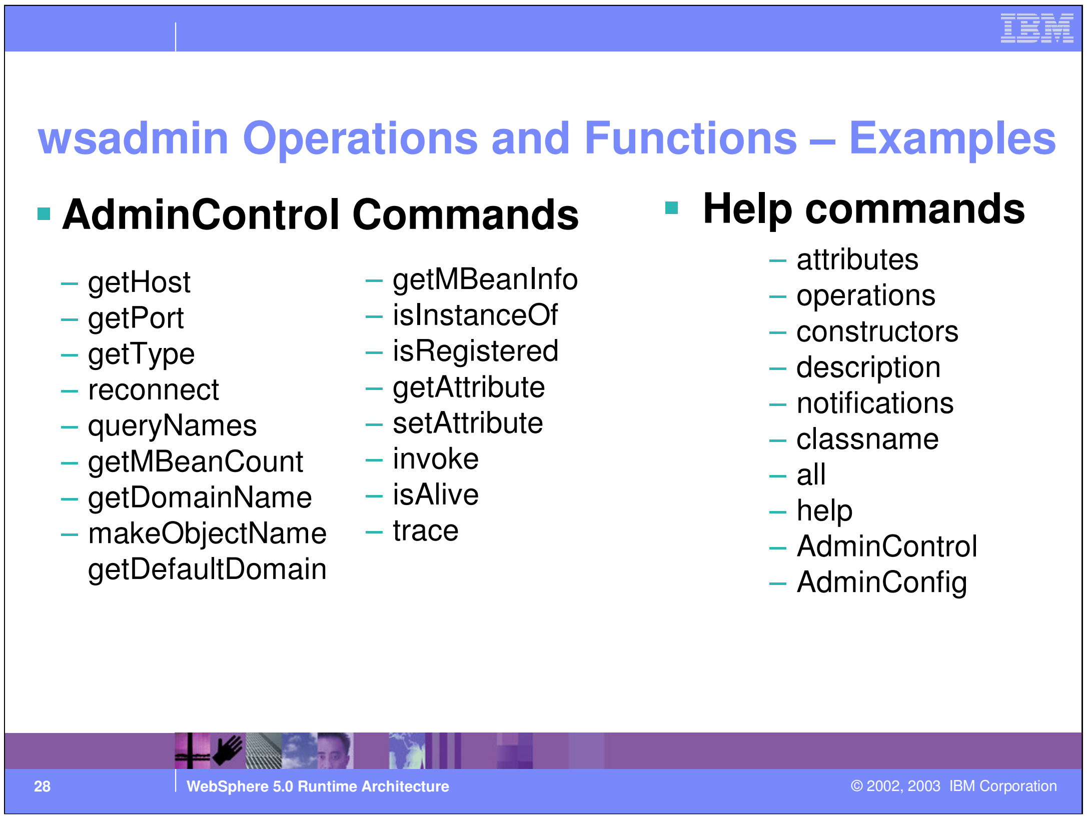
27 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

The list in the chart is a subset of the actual functions and operations available with wsadmin.

The list of commands available in wsadmin can be easily obtained by starting wsadmin and then typing "$AdminConfig" help at the prompt. Short descriptions of each command are also printed out when you list out the commands.

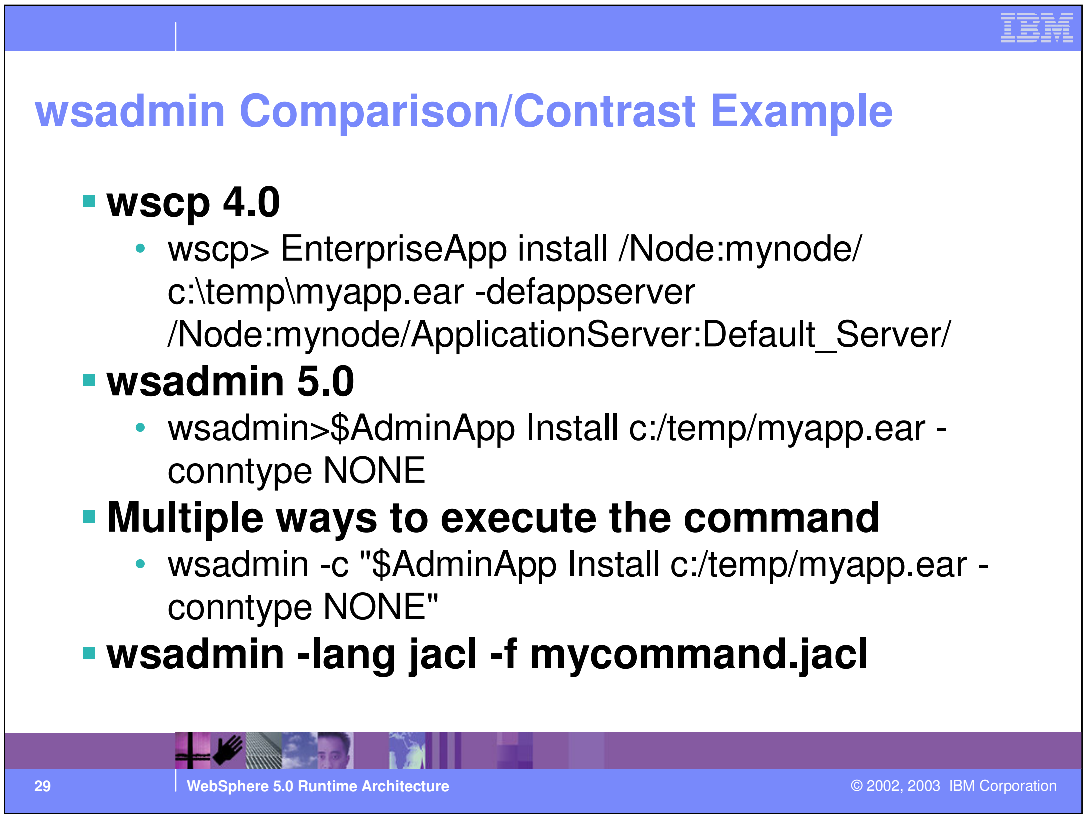
For installing applications the $AdminApp install or $AdminApp installInteractive can be used. This is different from WebSphere v4.0 where you needed to use SEAppInstall or WSCP to install an application from the command line depending on what version you were using (AE or AEs).

The command setSaveMode for the AdminConfig operation is worth noting as it changes the way in which configuration changes are saved. By default if changes are saved to a configuration while changes are being made through a separate process, the second first set of changes will be rolled back. By using this command you can force the save of the configuration over any changes made in the meantime.



wsadmin Operations and Functions – Examples

- **AdminControl Commands**
  - getHost
  - getPort
  - getType
  - reconnect
  - queryNames
  - getMBeanCount
  - getDomainName
  - makeObjectName getDefaultDomain
  - getMBeanInfo
  - isInstanceOf
  - isRegistered
  - getAttribute
  - setAttribute
  - invoke
  - isAlive
  - trace
- **Help commands**
  - attributes
  - operations
  - constructors
  - description
  - notifications
  - classname
  - all
  - help
  - AdminControl
  - AdminConfig



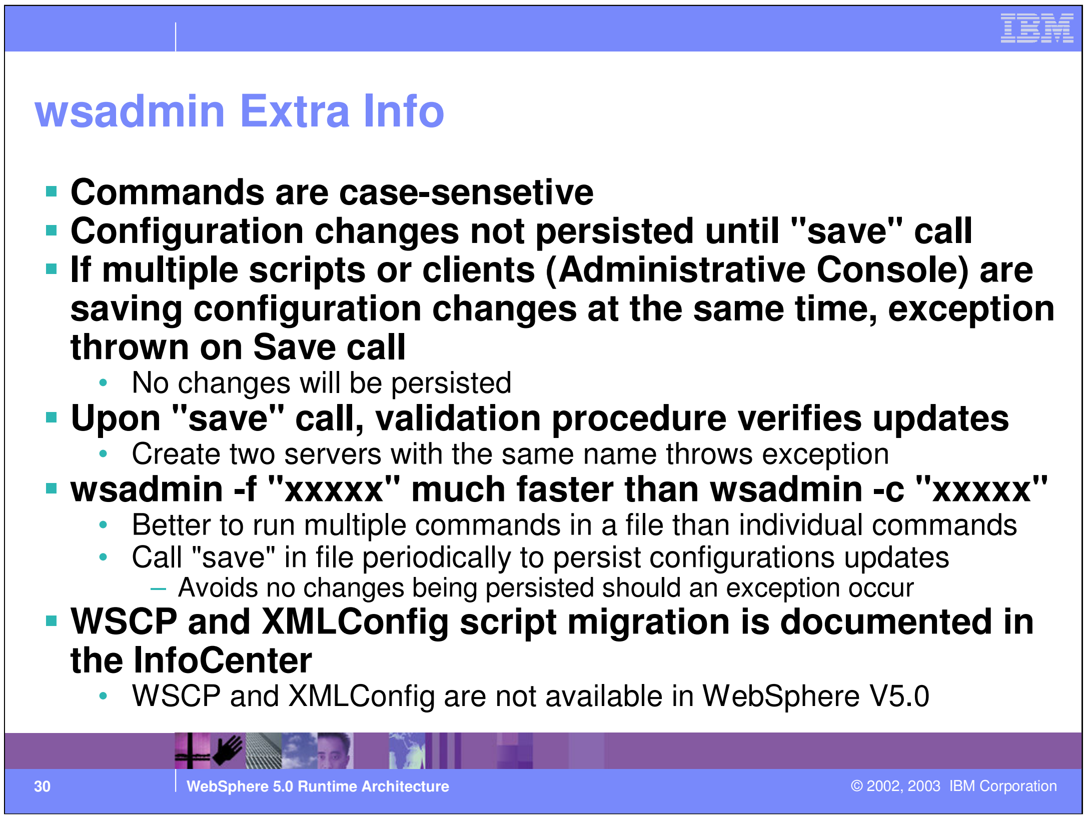
wsadmin Comparison/Contrast Example

- **wscp 4.0**
  - wscp> EnterpriseApp install /Node:mynode/ c:\temp\myapp.ear -defappserver /Node:mynode/ApplicationServer:Default_Server/
- **wsadmin 5.0**
  - wsadmin>$AdminApp Install c:/temp/myapp.ear -conntype NONE
- **Multiple ways to execute the command**
  - wsadmin -c "$AdminApp Install c:/temp/myapp.ear -conntype NONE"
- **wsadmin -lang jacl -f mycommand.jacl**

29 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

Here is an comparison and a contrast between the commands to install an enterprise application using wscp in WebSphere v4.0 and wsadmin in WebSphere V5.0. The syntax of the two commands is fairly similar with the wsadmin command actually being more straight forward.

As stated earlier, wsadmin commands can be run in multiple ways. Either one at a time using the -c option or multiple commands can be run from within a file using the -f option.



wsadmin Extra Info

- **Commands are case-sensetive**
- **Configuration changes not persisted until "save" call**
- **If multiple scripts or clients (Administrative Console) are saving configuration changes at the same time, exception thrown on Save call**
  - No changes will be persisted
- **Upon "save" call, validation procedure verifies updates**
  - Create two servers with the same name throws exception
- **wsadmin -f "xxxxx" much faster than wsadmin -c "xxxxx"**
  - Better to run multiple commands in a file than individual commands
  - Call "save" in file periodically to persist configurations updates
    - Avoids no changes being persisted should an exception occur
- **WSCP and XMLConfig script migration is documented in the InfoCenter**
  - WSCP and XMLConfig are not available in WebSphere V5.0

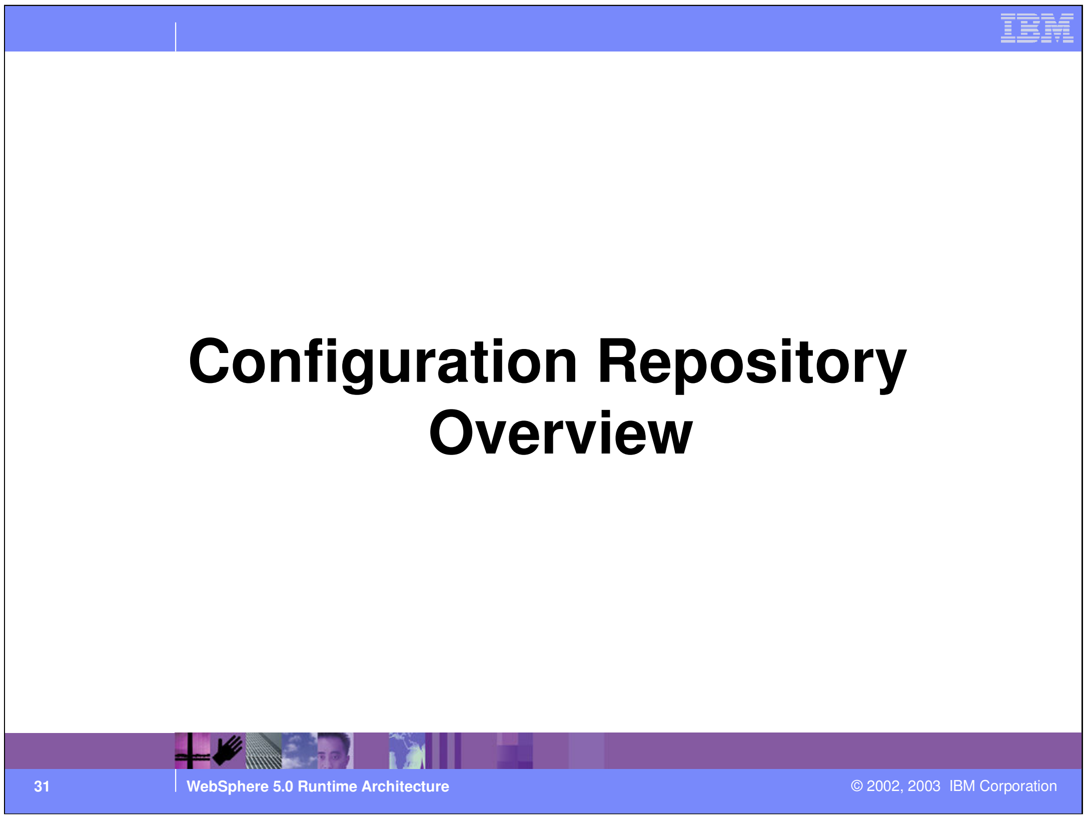
30 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

Changes to the configuration made through wsadmin occurs in a two step process. The first part of saving the changes validates the changes, while the second part actually performs the save, throwing an exception if changes have been made.

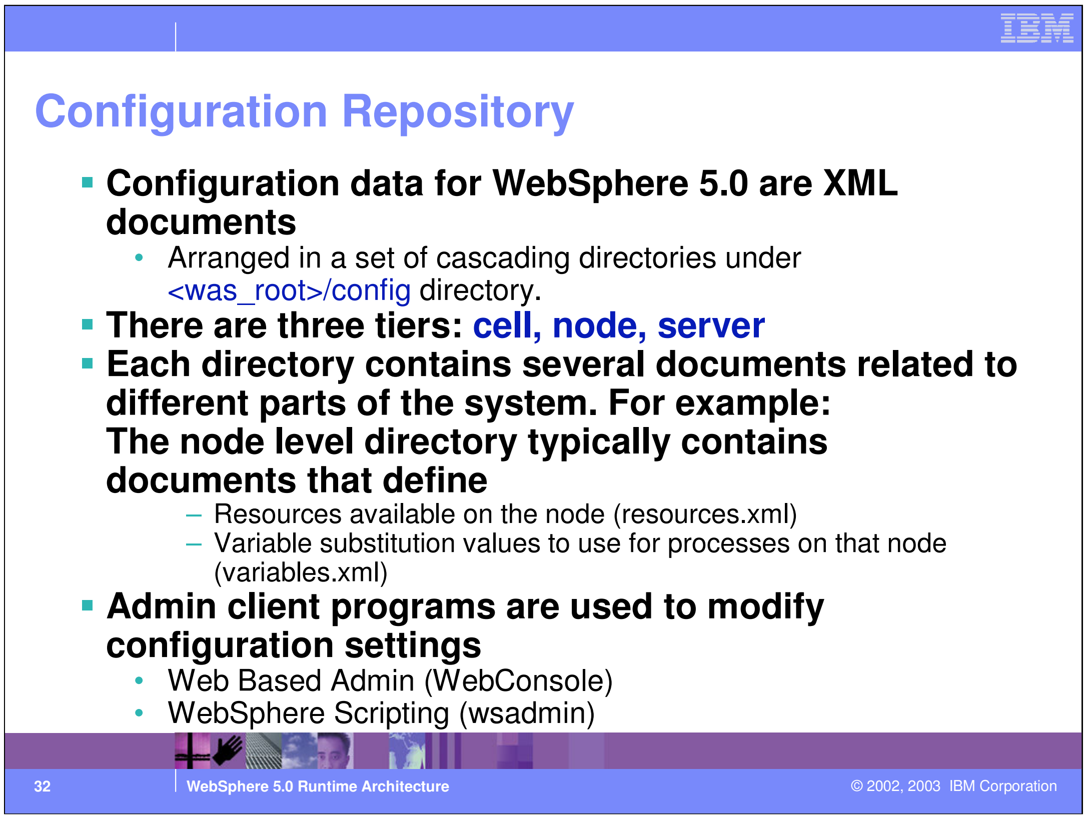
You will have better performance if you run multiple commands from a script file rather than running individual commands as the scripting environment will have to initialize each time.

When running multiple commands in a script, you have the potential to have conflicts and roll back the commands. To avoid having to run a large number of commands over, try calling save on configuration changes periodically.

WSCP and XMLConfig, available in WebSphere v4.0 AE are not included in WebSphere V5. Migration options are documented for moving to wsadmin - search the InfoCenter for 'wscp'



Configuration Repository Overview

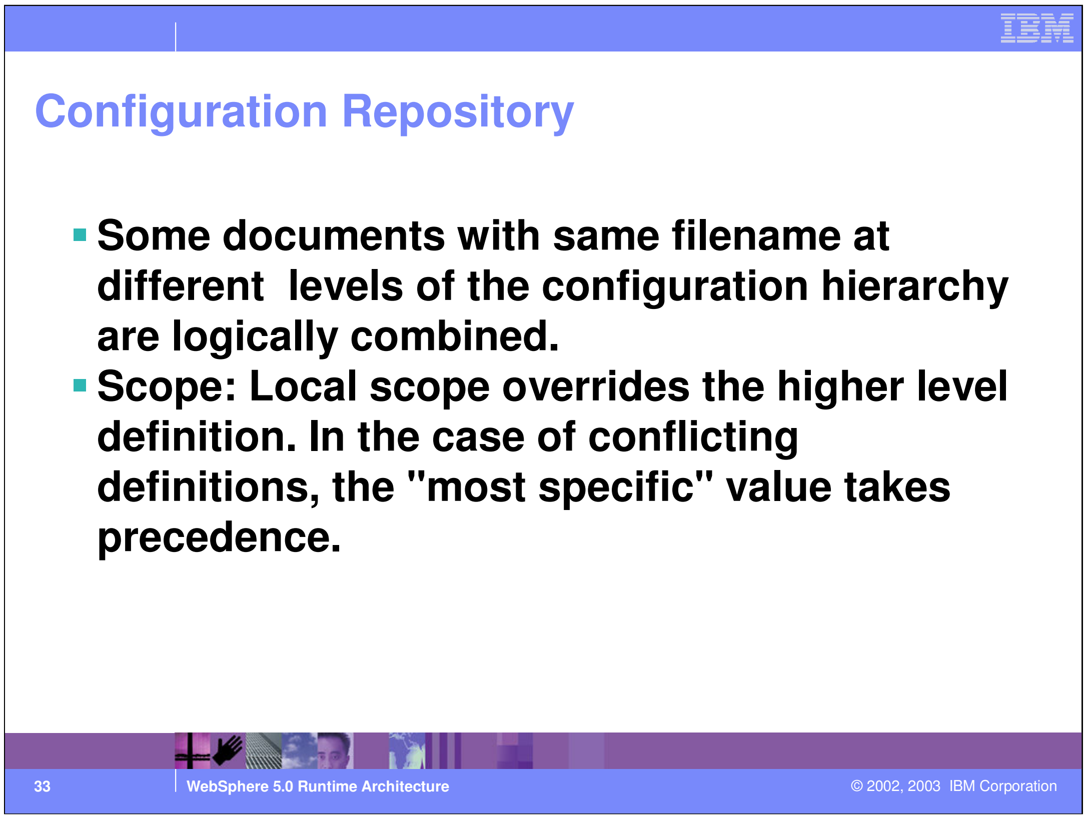


Configuration Repository

- **Configuration data for WebSphere 5.0 are XML documents**
  - Arranged in a set of cascading directories under <was_root>/config directory.
- **There are three tiers: cell, node, server**
- **Each directory contains several documents related to different parts of the system. For example: The node level directory typically contains documents that define**
  - Resources available on the node (resources.xml)
  - Variable substitution values to use for processes on that node (variables.xml)
- **Admin client programs are used to modify configuration settings**
  - Web Based Admin (WebConsole)
  - WebSphere Scripting (wsadmin)

32 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

WebSphere Application Server stores configuration data for in several documents in a cascading hierarchy of directories. Most configuration documents have XML content. 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 any changes that you make to configurations.



Configuration Repository

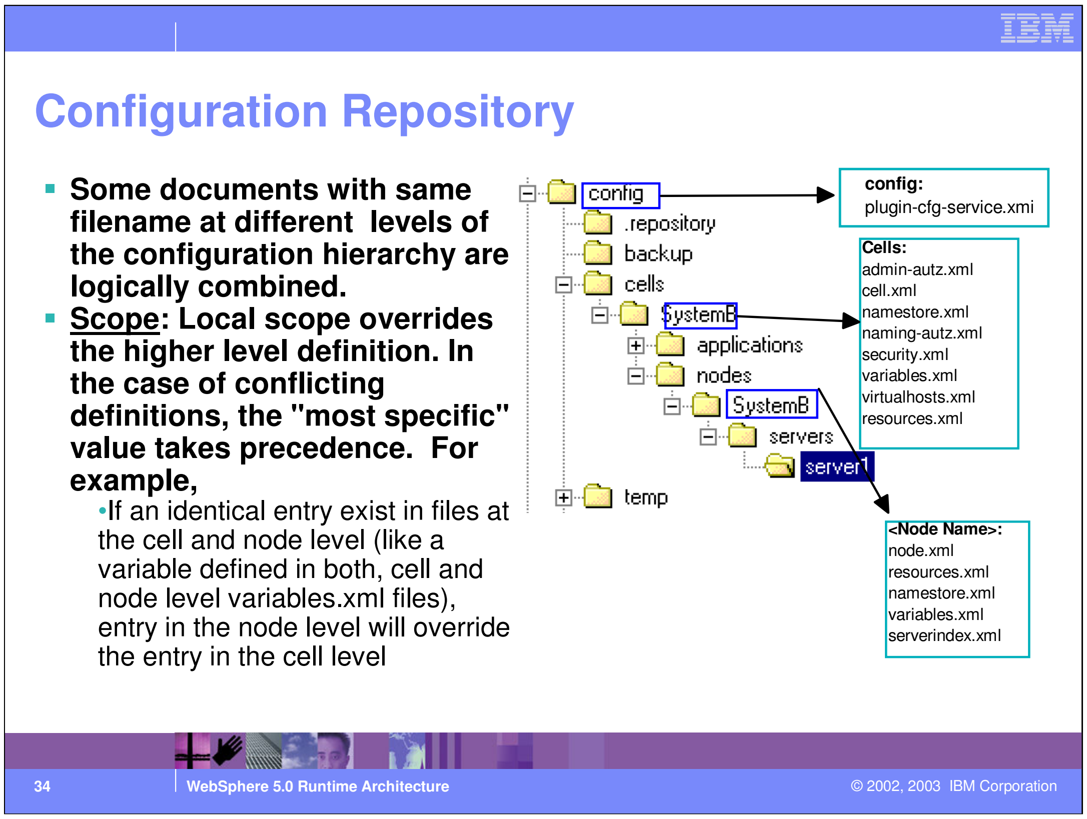
- Some documents with same filename at different levels of the configuration hierarchy are logically combined.
- Scope: Local scope overrides the higher level definition. In the case of conflicting definitions, the "most specific" value takes precedence.

33 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

Under the individual node directories there is a node.xml document that contains config data specific to that node. There are also several node level documents that contain configuration data that is common to all processes running on that node. Example node level documents are: namestore.xml, variables.xml, and resources.xml.

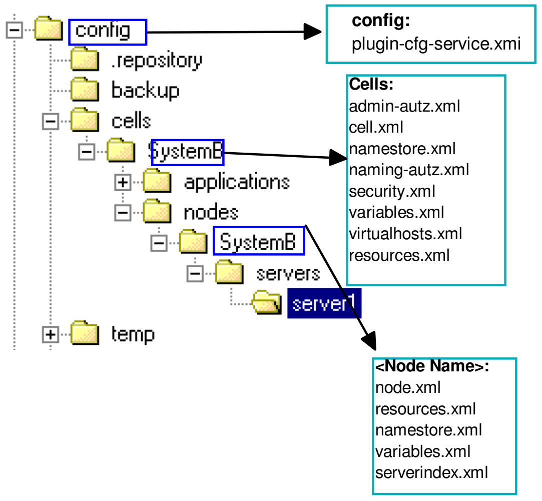
You will notice that these same named documents exist at the cell level. Identically named documents that exist at differing levels of the configuration hierarchy are termed "variable scoped" documents. One reason for this is that the configuration data contained in a document at one level is logically combined with data from documents at other levels of the configuration hierarchy. In the case of conflicting definitions, the "most specific" value takes precedence. For example, if a variable is defined in both cell and node level variables.xml files, the definition in the node document will be used by all processes on that node and the value from the cell document will be ignored by processes on the node.

Another reason for variable scoped documents is that they represent data that is not merged but is scoped to different levels of the topology. For example, the namestore.xml document at the cell level contains the cell persistent portion of the name space while the one at the node level contains the node persistent portion of the name space for this node.



Configuration Repository

- **Some documents with same filename at different levels of the configuration hierarchy are logically combined.**
- **<u>Scope</u>: Local scope overrides the higher level definition. In the case of conflicting definitions, the "most specific" value takes precedence. For example,**
  - If an identical entry exist in files at the cell and node level (like a variable defined in both, cell and node level variables.xml files), entry in the node level will override the entry in the cell level



config:
plugin-cfg-service.xmi

Cells:
admin-autz.xml
cell.xml
namestore.xml
naming-autz.xml
security.xml
variables.xml
virtualhosts.xml
resources.xml

<Node Name>:
node.xml
resources.xml
namestore.xml
variables.xml
serverindex.xml

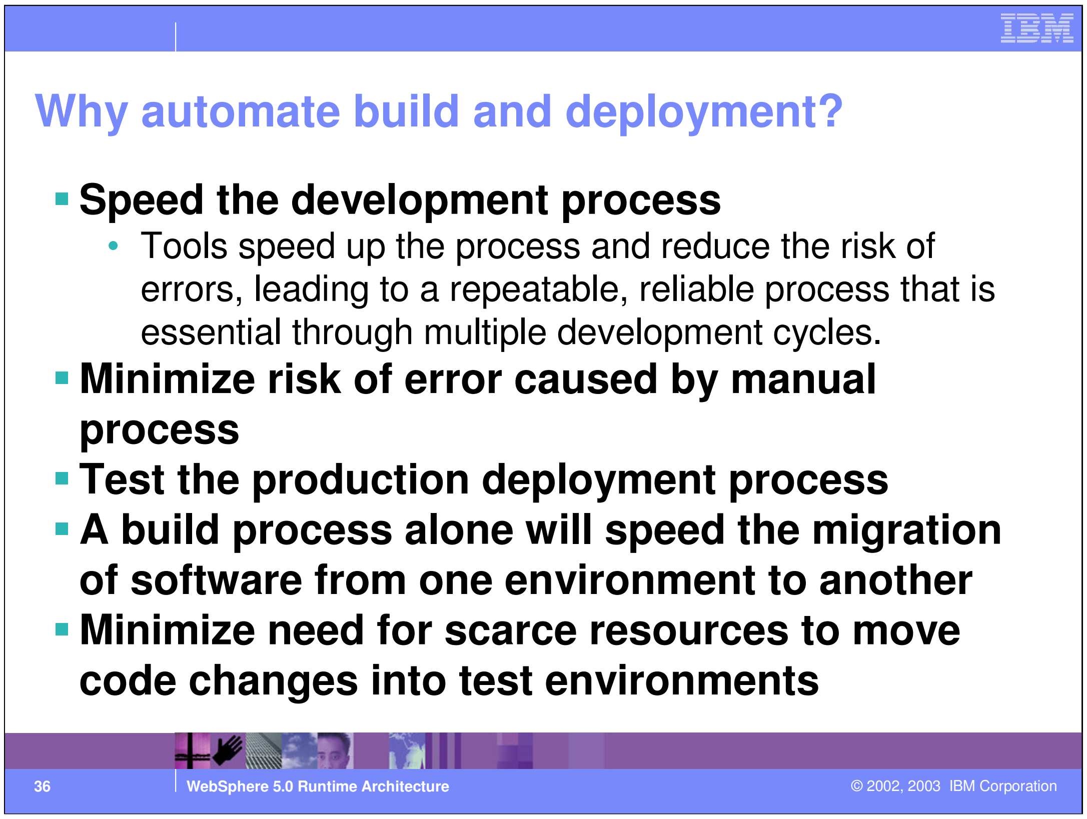
34 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

Under the individual node directories there is a node.xml document that contains config data specific to that node. There are also several node level documents that contain configuration data that is common to all processes running on that node. Example node level documents are: namestore.xml, variables.xml, and resources.xml.

You will notice that these same named documents exist at the cell level. Identically named documents that exist at differing levels of the configuration hierarchy are termed "variable scoped" documents. One reason for this is that the configuration data contained in a document at one level is logically combined with data from documents at other levels of the configuration hierarchy. In the case of conflicting definitions, the "most specific" value takes precedence. For example, if a variable is defined in both cell and node level variables.xml files, the definition in the node document will be used by all processes on that node and the value from the cell document will be ignored by processes on the node.

Another reason for variable scoped documents is that they represent data that is not merged but is scoped to different levels of the topology. For example, the namestore.xml document at the cell level contains the cell persistent portion of the name space while the one at the node level contains the node persistent portion of the name space for this node.

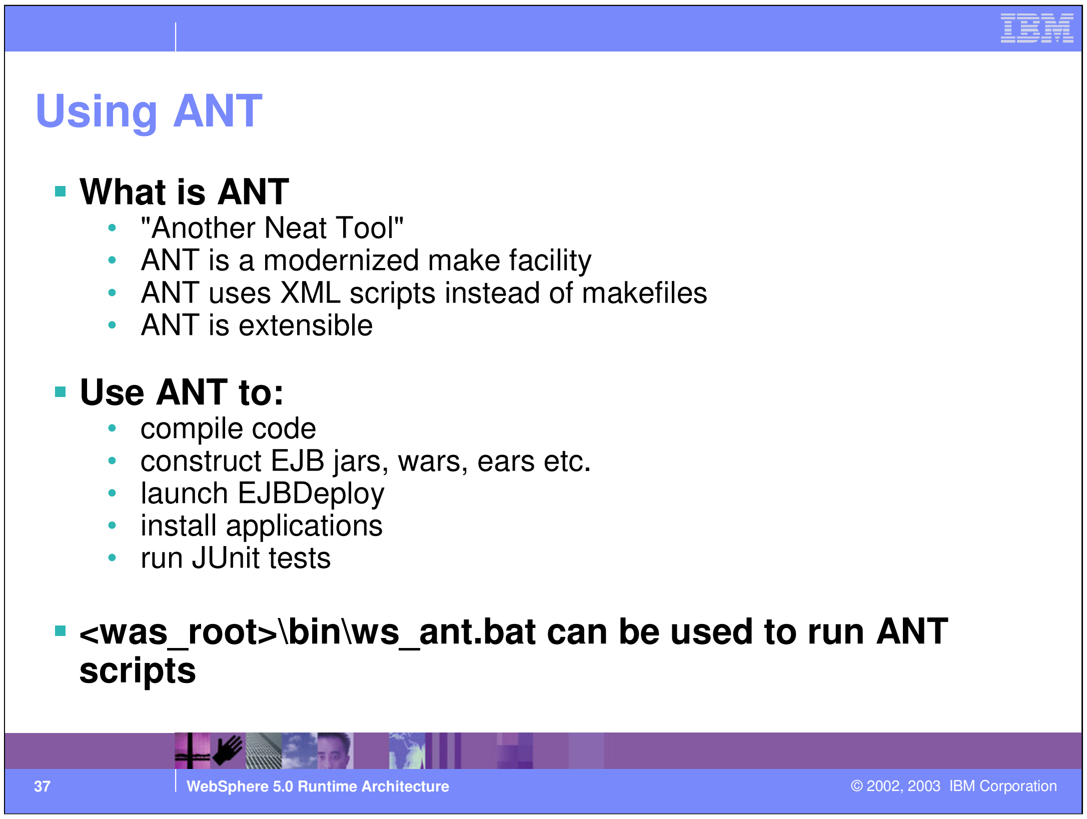
Automating the Build Process using ANT



Why automate build and deployment?

- **Speed the development process**
  - Tools speed up the process and reduce the risk of errors, leading to a repeatable, reliable process that is essential through multiple development cycles.
- **Minimize risk of error caused by manual process**
- **Test the production deployment process**
- **A build process alone will speed the migration of software from one environment to another**
- **Minimize need for scarce resources to move code changes into test environments**

A defined process is one of the most necessary but often least-used tools in software development. It is by nature an overhead task that accompanies a development effort. A defined build process ensures that the software in your development project is built in the exact same manner each time a build is executed. As the build process becomes more complex -- for example, with EJB builds or additional tasks -- it becomes more necessary to achieve such standardization. You should establish, document, and automate the exact series of steps as much as possible. A defined build process is an essential part of any development cycle because it helps close the gap between the development, integration, test, and production environments. A build process alone will speed the migration of software from one environment to another. It also removes many issues related to compilation, classpath, or properties that cost many projects time and money.



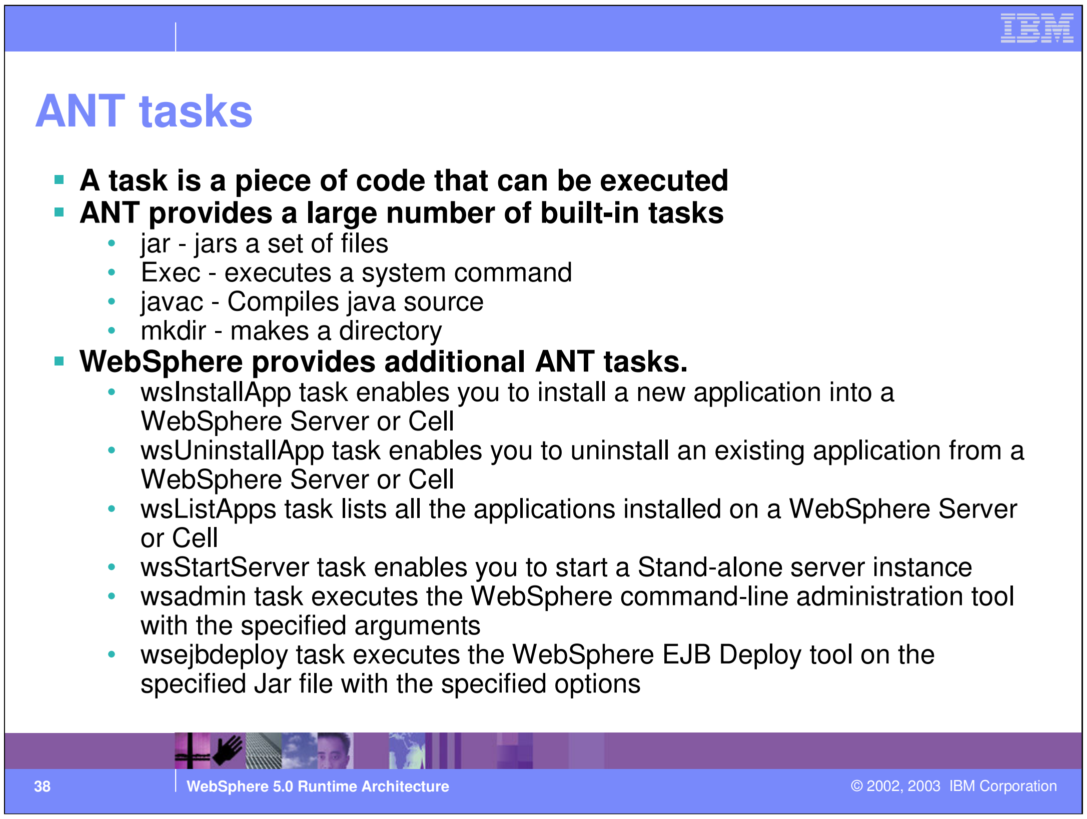
Using ANT

- **What is ANT**
  - "Another Neat Tool"
  - ANT is a modernized make facility
  - ANT uses XML scripts instead of makefiles
  - ANT is extensible
- **Use ANT to:**
  - compile code
  - construct EJB jars, wars, ears etc.
  - launch EJBDeploy
  - install applications
  - run JUnit tests
- **<was_root>\bin\ws_ant.bat can be used to run ANT scripts**

37 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

Ant is a scripting tool that lets you construct your build scripts in much the same fashion as the "make" tool in C or C++. Ant is a subproject of the Apache Jakarta project, part of the Apache Software Foundation. You can use a large number of built-in tasks in Ant without any customization. Because Ant is Java-based, it is platform-independent. It is well suited for building Java applications, but can be used for other build tasks as well. One of its important features is that you can use Java to write new Ant tasks to extend production build capabilities.

ANT tasks can be used to compile code, create EJB JAR files, WAR files and EAR files. ANT can also be used to perform tasks such as generating deployed code for your EJBs , installing an application , launching a client application and running JUnit tests.



ANT tasks

- **A task is a piece of code that can be executed**
- **ANT provides a large number of built-in tasks**
  - jar - jars a set of files
  - Exec - executes a system command
  - javac - Compiles java source
  - mkdir - makes a directory
- **WebSphere provides additional ANT tasks.**
  - wsInstallApp task enables you to install a new application into a WebSphere Server or Cell
  - wsUninstallApp task enables you to uninstall an existing application from a WebSphere Server or Cell
  - wsListApps task lists all the applications installed on a WebSphere Server or Cell
  - wsStartServer task enables you to start a Stand-alone server instance
  - wsadmin task executes the WebSphere command-line administration tool with the specified arguments
  - wsejbdeploy task executes the WebSphere EJB Deploy tool on the specified Jar file with the specified options

38 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

A task is a piece of code that can be executed. There is a set of built-in tasks, but it is also very easy to write your own. ANT provides built in tasks for compiling java source code, creating jar and war files etc. For documentation on the full set of ANT tasks, refer the ANT user's manual http://jakarta.apache.org/ant/manual/index.html

WebSphere provides ANT tasks in addition to the built-in ANT tasks. Examples are wsInstalApp,wsListApps, wsStartServer, wsUninstallApp , wsStopServer and wsadmin.

The wsInstallApp task enables you to install a new application into a WebSphere Server or Cell. This task is a subclass of the wsadmin task and shares many of the

same attributes. This task is a wrapper for the AdminApp.install() command of the wsadmin tool.

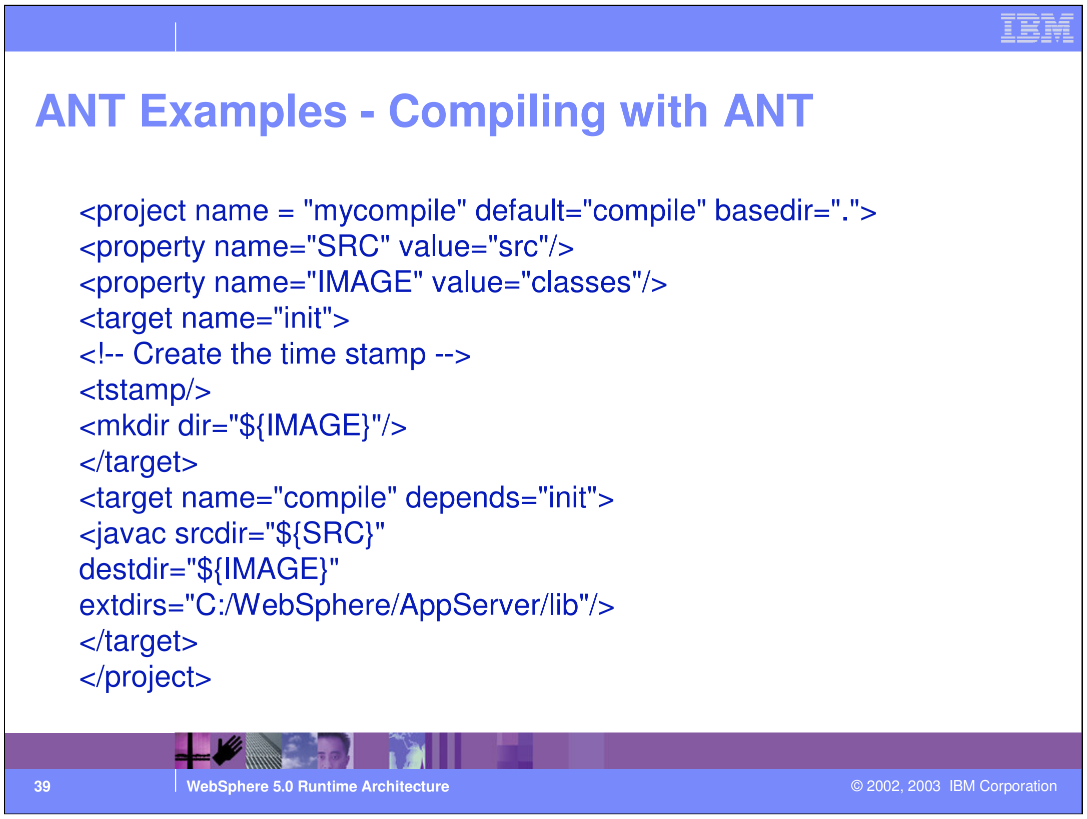
The wsListApps task lists all the applications installed on a WebSphere Server or Cell. This task is a subclass of the wsadmin task and shares many of the same

attributes. This task is a wrapper for the AdminApp.list() command of the wsadmin tool.

The wsStartServer task enables you to start a stand-alone server instance. This is not used to start a server controlled by DeploymentManager. Therefore, this task

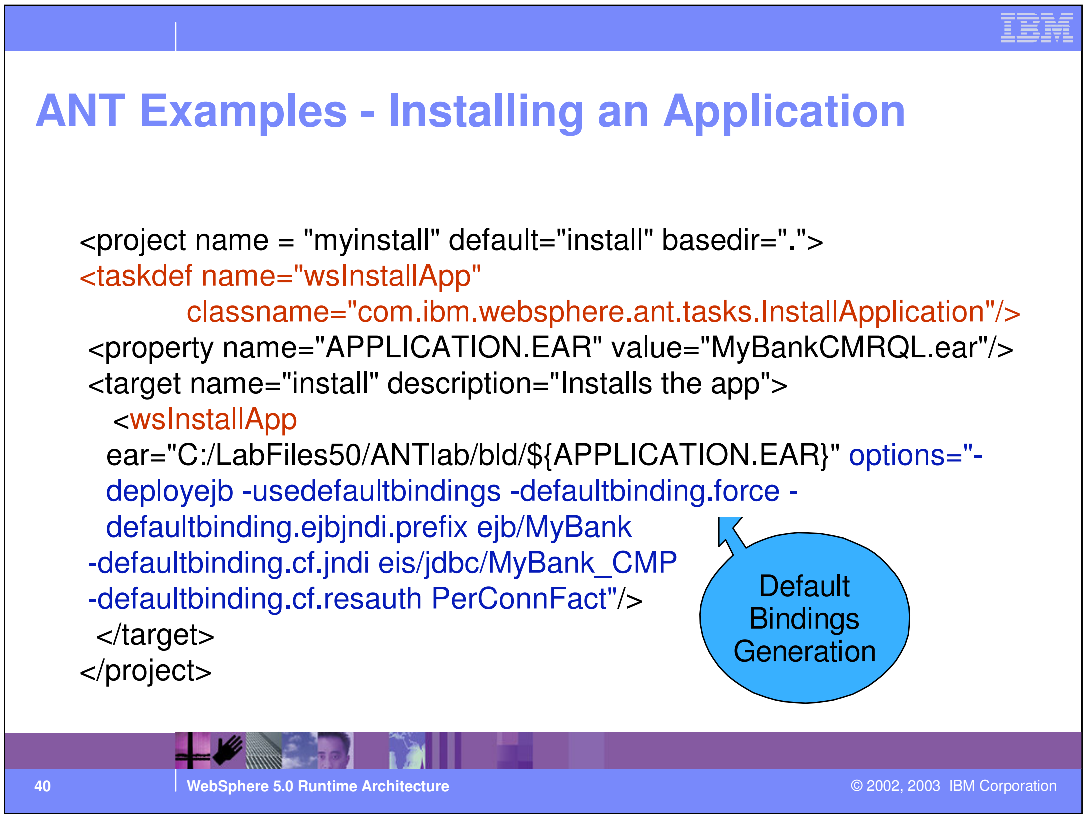
is useful for the Base Application Server, and to start the Node Agent and/or DeploymentManager. If you wish to start a server managed by the Deployment

Manager, use the wsadmin task to execute a scripting command.



ANT Examples - Compiling with ANT

```
<project name = "mycompile" default="compile" basedir=".">
<property name="SRC" value="src"/>
<property name="IMAGE" value="classes"/>
<target name="init">
<!-- Create the time stamp -->
<tstamp/>
<mkdir dir="${IMAGE}"/>
</target>
<target name="compile" depends="init">
<javac srcdir="${SRC}"
destdir="${IMAGE}"
extdirs="C:/WebSphere/AppServer/lib"/>
</target>
</project>
```



ANT Examples - Installing an Application

```
<project name = "myinstall" default="install" basedir=".">
<taskdef name="wsInstallApp"
        classname="com.ibm.websphere.ant.tasks.InstallApplication"/>
 <property name="APPLICATION.EAR" value="MyBankCMRQL.ear"/>
 <target name="install" description="Installs the app">
   <wsInstallApp
   ear="C:/LabFiles50/ANTlab/bld/${APPLICATION.EAR}" options="-
   deployejb -usedefaultbindings -defaultbinding.force -
   defaultbinding.ejbjndi.prefix ejb/MyBank
 -defaultbinding.cf.jndi eis/jdbc/MyBank_CMP
 -defaultbinding.cf.resauth PerConnFact"/>
  </target>
</project>
```

40 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

WebSphere Application Server Release 5 includes support for Ant tasks. When you use one of WebSphere's ANT tasks, a taskdef element should be added to the project before using the task. The wsInstallApp task enables you to install a new application into a WebSphere Server or Cell. This task is a subclass of the wsadmin task and shares many of the same attributes. This task is a wrapper for the AdminApp.install() command of the wsadmin tool.

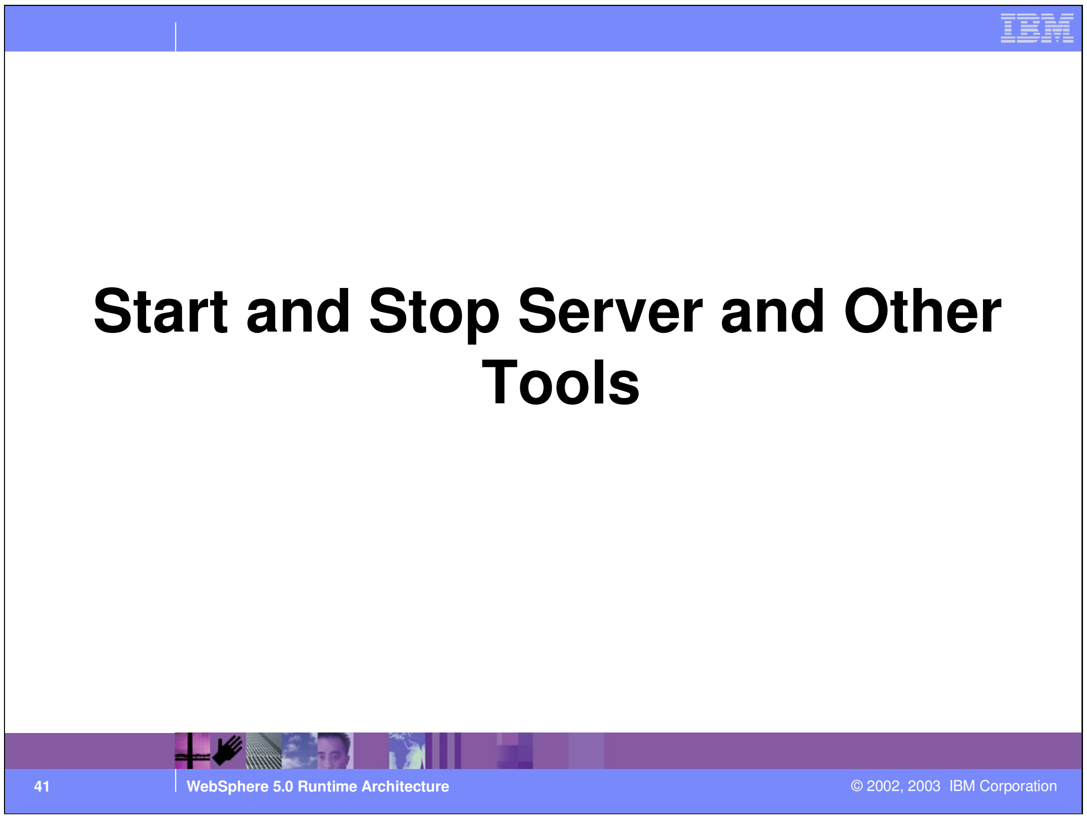
Default Bindings Generation :

WebSphere Application Server, Release 5 provides support for automatically generating bindings for J2EE applications.

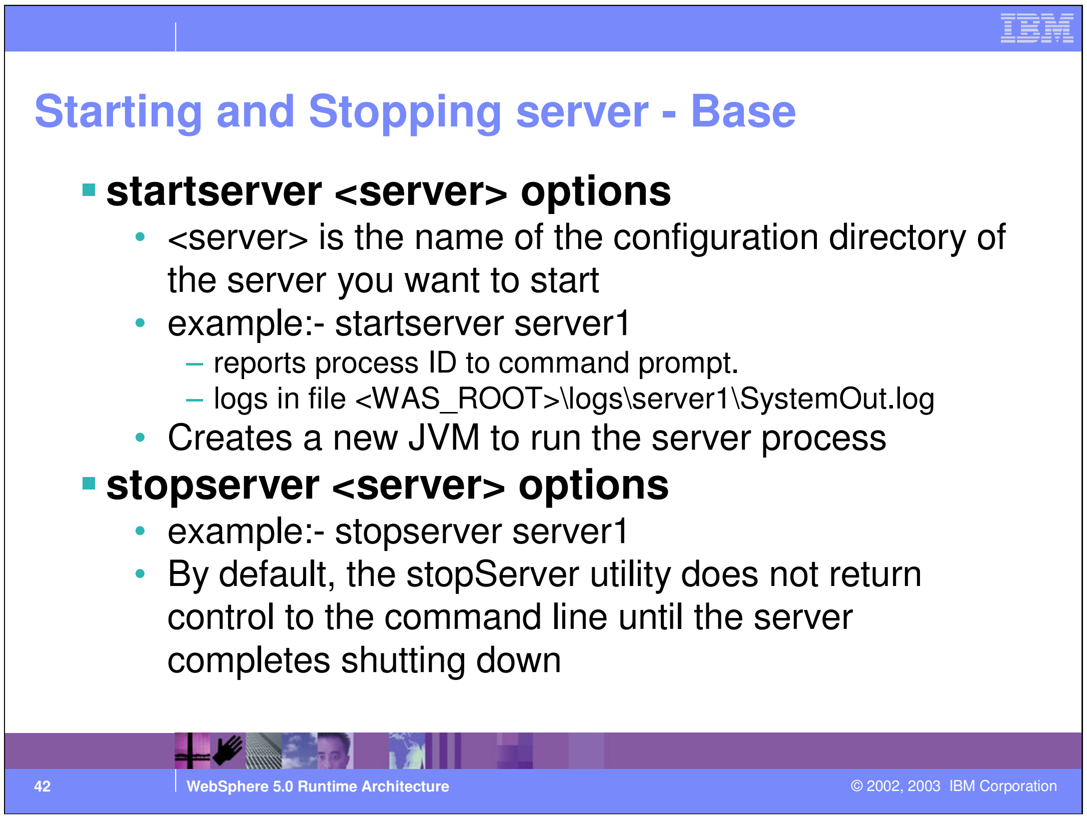
-usedefaultbindings enables binding generation.

-defaultbinding.force forces pre-existing bindings to be overwritten. -defaultbinding.ejbjndi.prefix prefix specifies the JNDI prefix for all global EJB JNDI names.

-defaultbinding.cf.jndi jndi-name and -defaultbinding.cf.resauth { PerConnFact | Container } specify the default connection factory bindings for CMP 2.0 modules.



Start and Stop Server and Other Tools



Starting and Stopping server - Base

- **startserver <server> options**
  - <server> is the name of the configuration directory of the server you want to start
  - example:- startserver server1
    - reports process ID to command prompt.
    - logs in file <WAS_ROOT>\logs\server1\SystemOut.log
  - Creates a new JVM to run the server process
- **stopserver <server> options**
  - example:- stopserver server1
  - By default, the stopServer utility does not return control to the command line until the server completes shutting down

42 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

The options for the startServer command are:

-nowait: Tells the startServer command not to wait for successful initialization of the launched server process.

-binaryData: Generates a binary launch data file for use by the eclipse plugin.

-quiet: Suppresses the progress information that the startServer command prints in normal mode.

-trace: Generates trace information into a file, using the startServer command, for debugging purposes.

-timeout <seconds>: Specifies the waiting time before server initialization times out and returns an error.

-statusport <portnumber>: Allows an administrator to set the port number for server status callback.

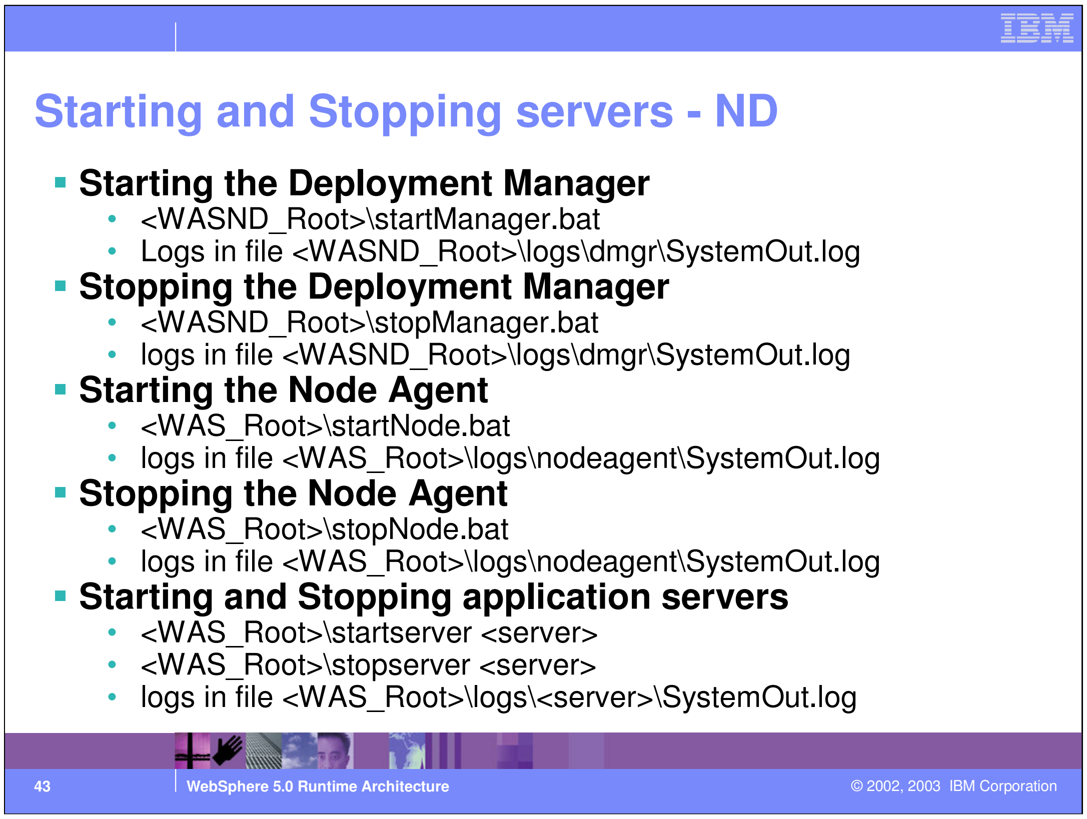
-cell <cellname>: Allows override of the current cell value that is set in setupCmdLine tool.

-node <nodename>: Allows override of the current node value that is set in the setupCmdLine tool.

-script [<script filename>]: Generates a launch script with the startServer tool, instead of launching the server process directly. The launch script name is an optional argument. If you do not supply the launch script name, the default script file name is start_<server> based on the <server> name passed as the first argument to the startServer command.

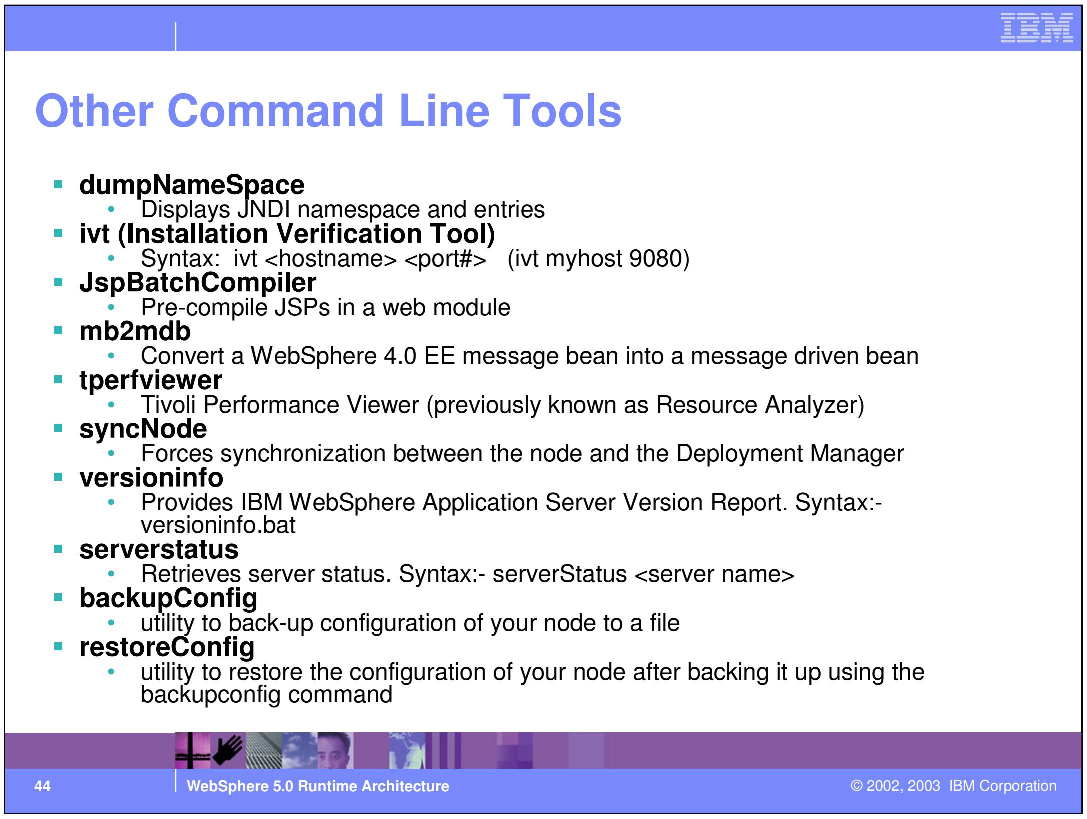
The options for the stopServer command are:

-nowait: Tells the stopServer command not to wait for successful initialization of the launched server process.



Starting and Stopping servers - ND

- **Starting the Deployment Manager**
  - <WASND_Root>\startManager.bat
  - Logs in file <WASND_Root>\logs\dmgr\SystemOut.log
- **Stopping the Deployment Manager**
  - <WASND_Root>\stopManager.bat
  - logs in file <WASND_Root>\logs\dmgr\SystemOut.log
- **Starting the Node Agent**
  - <WAS_Root>\startNode.bat
  - logs in file <WAS_Root>\logs\nodeagent\SystemOut.log
- **Stopping the Node Agent**
  - <WAS_Root>\stopNode.bat
  - logs in file <WAS_Root>\logs\nodeagent\SystemOut.log
- **Starting and Stopping application servers**
  - <WAS_Root>\startserver <server>
  - <WAS_Root>\stopserver <server>
  - logs in file <WAS_Root>\logs\<server>\SystemOut.log



Other Command Line Tools

- **dumpNameSpace**
  - Displays JNDI namespace and entries
- **ivt (Installation Verification Tool)**
  - Syntax: ivt <hostname> <port#> (ivt myhost 9080)
- **JspBatchCompiler**
  - Pre-compile JSPs in a web module
- **mb2mdb**
  - Convert a WebSphere 4.0 EE message bean into a message driven bean
- **tperfviewer**
  - Tivoli Performance Viewer (previously known as Resource Analyzer)
- **syncNode**
  - Forces synchronization between the node and the Deployment Manager
- **versioninfo**
  - Provides IBM WebSphere Application Server Version Report. Syntax:- versioninfo.bat
- **serverstatus**
  - Retrieves server status. Syntax:- serverStatus <server name>
- **backupConfig**
  - utility to back-up configuration of your node to a file
- **restoreConfig**
  - utility to restore the configuration of your node after backing it up using the backupconfig command

44 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

dumpnamespace:

The name space stored by a given name server can be dumped with the dumpNameSpace tool that is shipped with WebSphere Application Server. This tool can be invoked from the command line or from a Java program. The naming service for the WebSphere Application Server host must be active when this tool is invoked

The JspBatchCompiler utility can be used to precompile JSPs in web modules contained in enterprise applications. This utility can be run on applications before they are installed in the application server.

Syntax: dumpnamespace <options>

The mb2mdb is utility to convert message beans created for WebSphere v4 Enterprise Edition to messaged driven beans for WebSphere V5. Syntax is :-

mb2mdb <inputMB.jar/ear> <jmsListenerConfig.xml> <workingDirectory> <outputMDB.jar/ear>

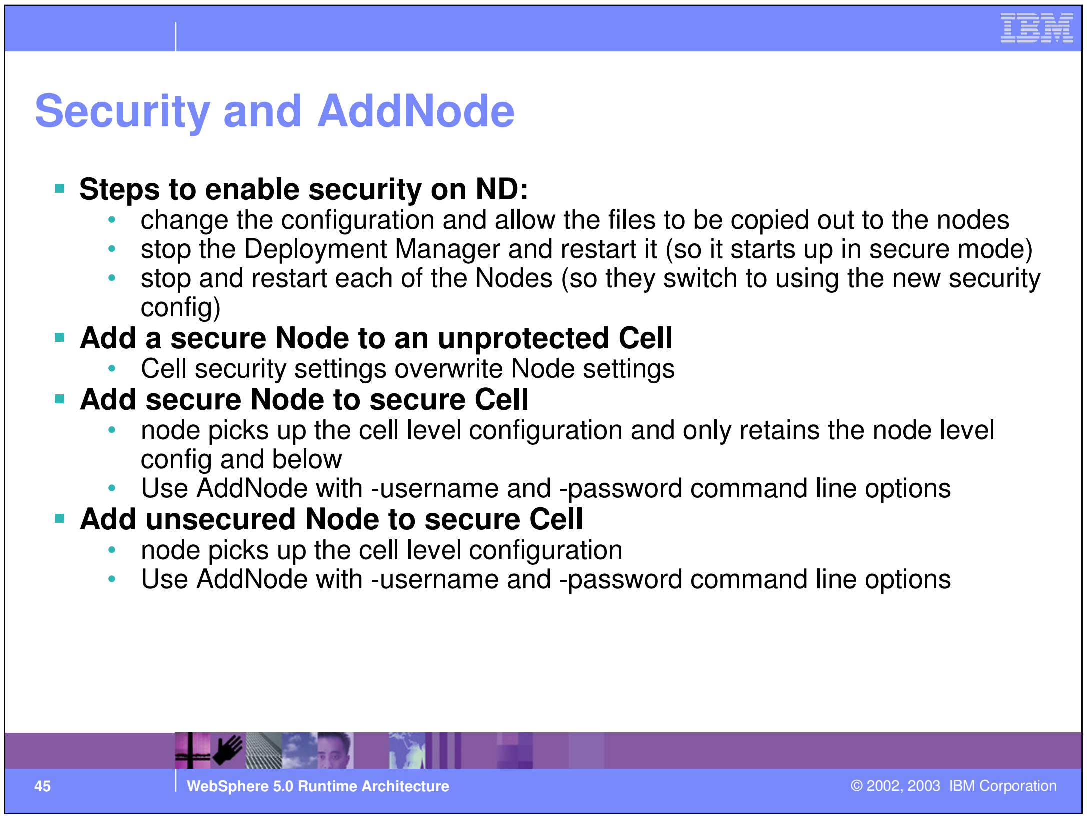
tperfviewer: To start performance monitoring from the command line.

Go to the product_installation_directory/bin directory and run the tperfviewer script.

You can specify the host and port in Windows NT and 2000 environments as: tperfviewer.bat host name port_number

JSPBatchCompiler: As an IBM enhancement to JSP support, IBM WebSphere Application Server provides a batch JSP compiler. Use this function to batch compile your JSP files and thereby enable faster responses to the initial client requests for the JSP files on your production Web server.

Batch compiling makes the first request for a JSP file much faster because the JSP file is translated and compiled into a servlet. Batch compiling is also useful as a fast way to resynchronize all of the JSP files for an application.



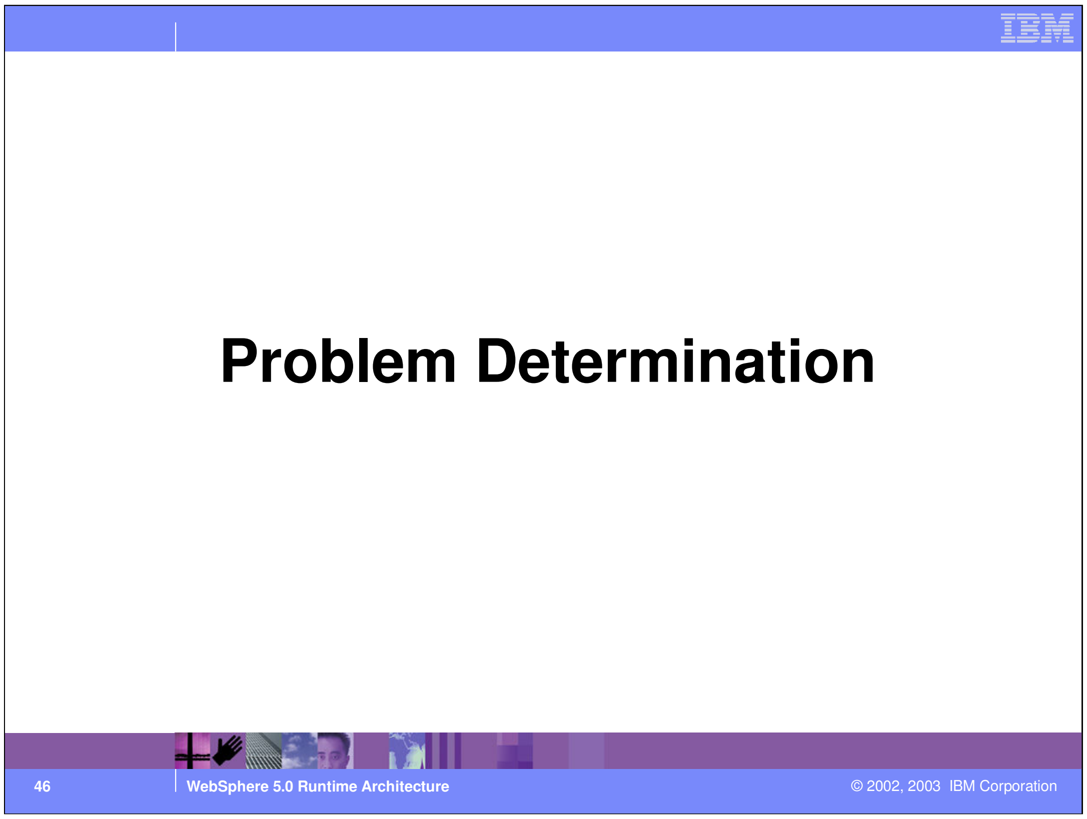
Security and AddNode

- **Steps to enable security on ND:**
  - change the configuration and allow the files to be copied out to the nodes
  - stop the Deployment Manager and restart it (so it starts up in secure mode)
  - stop and restart each of the Nodes (so they switch to using the new security config)
- **Add a secure Node to an unprotected Cell**
  - Cell security settings overwrite Node settings
- **Add secure Node to secure Cell**
  - node picks up the cell level configuration and only retains the node level config and below
  - Use AddNode with -username and -password command line options
- **Add unsecured Node to secure Cell**
  - node picks up the cell level configuration
  - Use AddNode with -username and -password command line options

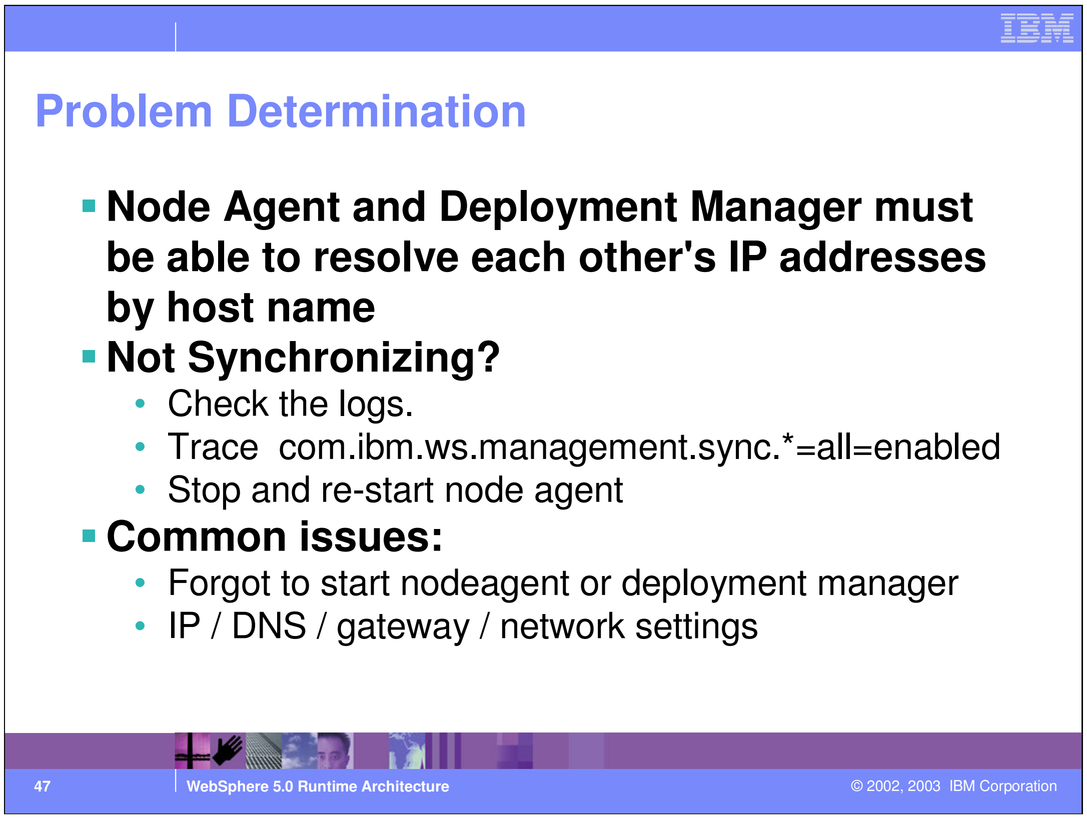
45 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

When you federate a node into a cell, the node picks up the cell level configuration and only retains the node level config and below. So cell level items such as cell security, virtualhosts, and cell level resources defined on the stand-alone node are not merged in with the cell. Instead, whatever settings there are in force at the ND cell are replicated down to the node during AddNode.

If you want a secure node after AddNode, then you can setup security through the ND dmgr before running AddNode. Be sure to remember to use the -username and -password command line options for AddNode in that case, or else it won't be able to connect to the ND dmgr. An alternative to the command line options is to set properties in the sas.client.props file for AddNode to pick up.



Problem Determination



Problem Determination

- **Node Agent and Deployment Manager must be able to resolve each other's IP addresses by host name**
- **Not Synchronizing?**
  - Check the logs.
  - Trace com.ibm.ws.management.sync.*=all=enabled
  - Stop and re-start node agent
- **Common issues:**
  - Forgot to start nodeagent or deployment manager
  - IP / DNS / gateway / network settings

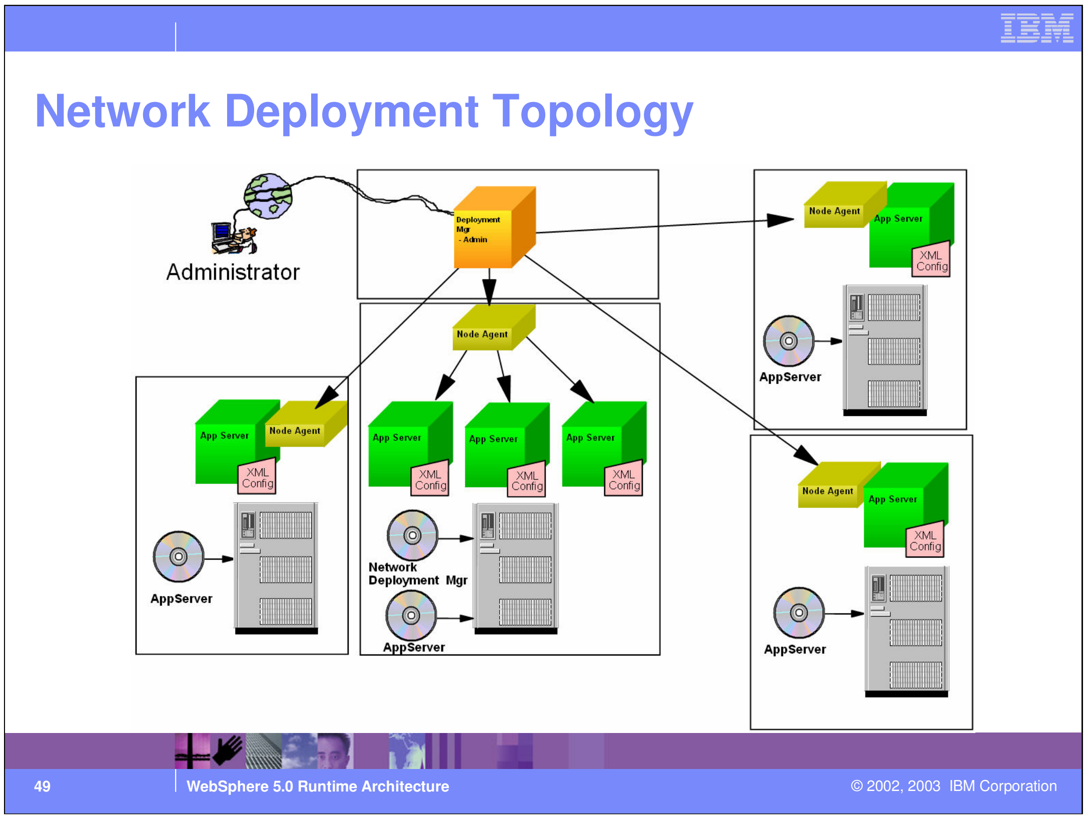
When you do an AddNode, the two machines need to be able to resolve each other by IP AND by the HostName each was configured with, unless the IP address was given AS the host name at installation time.

File synchronization is triggered explicitly in the code in several places. There is also a button in the Admin Console to force a synchronization if needed. The servers do not use file timestamps to determine what files need to be synchronized.

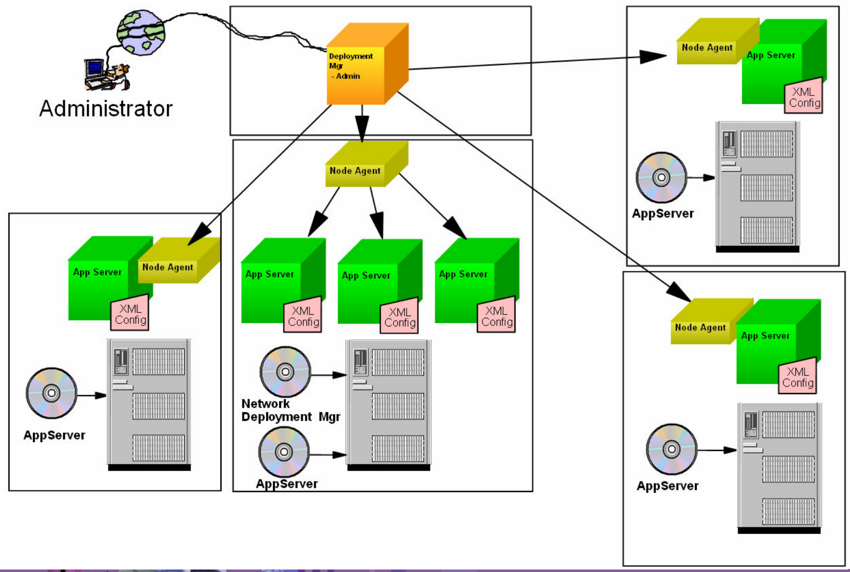
Cluster members are created by the synchronization mechanism copying configuration files to the node agent. After a 'Save', the files exist on the Deployment manager. If cluster members are not being created on the node, troubleshoot synchronization issues.

One frustrating thing is forgetting to start the node agent! When the node agent cannot communicate with the Deployment Manager for any reason (Network issues, not started, etc.), the symptom is that application server status is listed as 'unavailable' in the administration console.

Summary



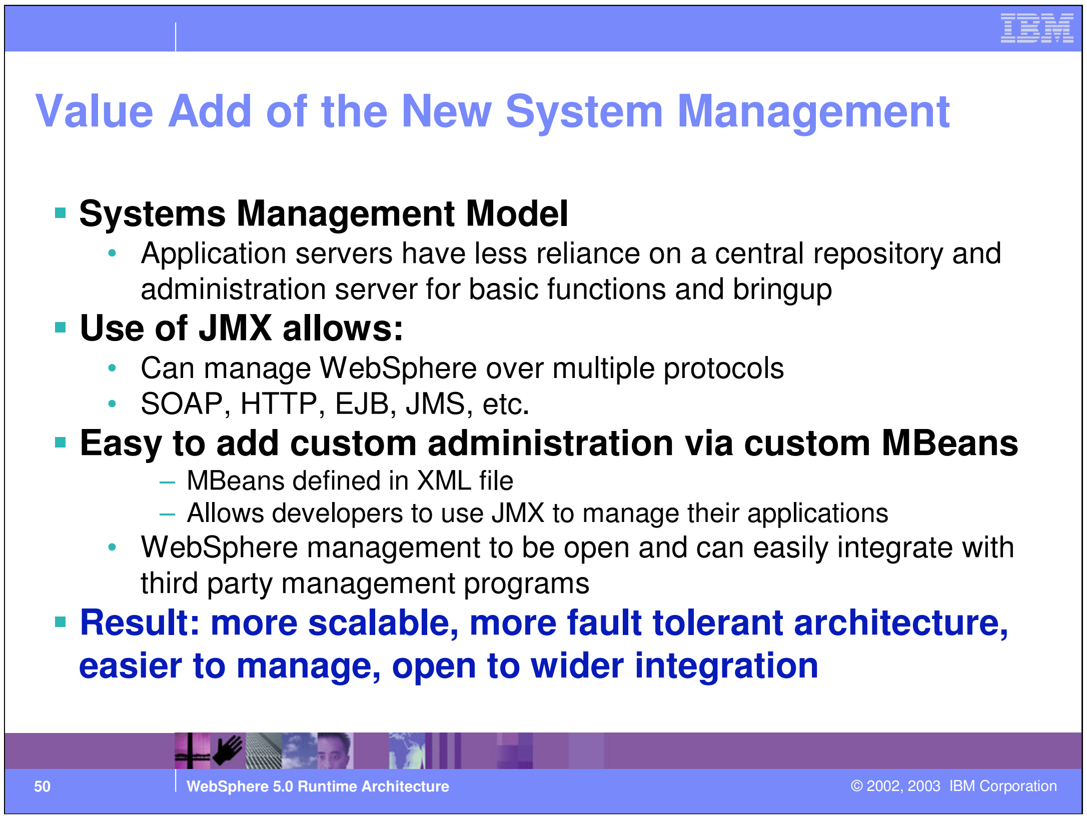
Network Deployment Topology



49 WebSphere 5.0 Runtime Architecture © 2002, 2003 IBM Corporation

Here is a sample view of one of a number of possible scenarios using Network Deployment. The Deployment Manager is responsible for the configuration of six installs of WebSphere - represented by green boxes. The four Node Agents serve to communicate the configuration information to the application servers, where it is written in the XML repository files - the pinkish things...

Note that on the box near the center of the page, both App Servers and a Network Deployment have been installed. This is the box that serves as the Deployment Manager for the topology.



Value Add of the New System Management

- **Systems Management Model**
  - Application servers have less reliance on a central repository and administration server for basic functions and bringup
- **Use of JMX allows:**
  - Can manage WebSphere over multiple protocols
  - SOAP, HTTP, EJB, JMS, etc.
- **Easy to add custom administration via custom MBeans**
    - MBeans defined in XML file
    - Allows developers to use JMX to manage their applications
  - WebSphere management to be open and can easily integrate with third party management programs
- **Result: more scalable, more fault tolerant architecture, easier to manage, open to wider integration**

Trademarks and Disclaimers