IBM WebSphere Application Server for z/OS V5.0.1

Getting Started

GA22-7957-00

Note

Before using this information, be sure to read the general information under “Notices” on page 175.

Compilation date: October 1, 2003

Contents

How to send your comments v

Summary of Changes vii
The Java 2 platform, Enterprise Edition (J2EE) specification vii
Architectural features ix
Product GUIs and tools xi
User roles and activities xii
InfoCenter xiii

Introduction to this book. xix
Task overviews xix

Chapter 1. Installing WebSphere Application Server 1
WebSphere Application Server packages 2
Planning to install an e-business network. 4
  Single server topology 5
  Establishing multimachine environments 7
  Setting up a multinode environment 9
  Running WebSphere Application Server across versions 28
Installing new releases and maintenance levels of WebSphere Application Server for z/OS. 31
  Using a version-specific HFS structure to upgrade WebSphere Application Server for z/OS. 32
  Using an alternate HFS structure to upgrade WebSphere Application Server for z/OS. . . . 34
Installing and customizingthe product 36
  Overview of WebSphere for z/OS configuration 37
  Preparing the base z/OS environment 48
  Preparing for installation and customization . . 56
  Installing the code through SMP/E 57
  Using the Customization Dialog 57
  z/OS port assignments 101
  Automatic restart management 102
  Tips for installing the embedded messaging feature 105
  Migrating and coexisting 106
Installing service to WebSphere Application Server for z/OS 127
  Applying or restoring off a service level . . . 128
Using the installation verification test 131
  Running the Installation Verification Test with a job 131
  Running the Installation Verification Test from a command line 131
Migrating XML applications 132
Creating multiple Version 5 configuration instances 133
  wsinstance command. 135
  Creating servers in coexistence or multiple instance environments 140
  Changing HTTP transport ports 141
Product version and history information 142
Installation: Resources for learning 164

Chapter 2. Samples Gallery 173

Notices 175

Trademarks and service marks. . . . 177

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.

Summary of Changes

This section describes what is new in WebSphere Application Server for z/OS in V5.0.1. It also provides a general overview of V5.

The Java 2 platform, Enterprise Edition (J2EE) specification

The Java 2 platform, Enterprise Edition (J2EE) specification is the standard for developing, deploying, and running enterprise applications. Read this topic for a brief overview of key J2EE concepts, including the parts of the J2EE run-time environment and the J2EE application packaging and deployment. The ultimate source of J2EE information is the specification, available from the Sun Microsystems Web site (java.sun.com). For a list of the specification levels that comprise the J2EE 1.3 specification (Servlet, EJB, and so on), see http://java.sun.com/j2ee/1.3/docs/#specs.

IBM WebSphere Application Server, Version 5 has completed the full J2EE certification test suite. The product supports all of the J2EE 1.3 APIs, and exceeds many with its extensions. You can check the list of J2EE-compatible products posted by Sun Microsystems at http://java.sun.com/j2ee/compatibility.html.

Java 2 Platform, Enterprise Edition defines a standard that applies to all aspects of designing, developing, and deploying multitier, server-based applications. The standard architecture defined by the J2EE specification is composed of the following elements:

- Standard application model for developing multitier applications.
- Standard platform for hosting applications.
- Compatibility test suite for verifying that J2EE platform products comply with the J2EE platform standard.
- Reference implementation providing an operational definition of the J2EE platform.

The J2EE platform specification describes the run-time environment for a J2EE application. This environment includes application components, containers, and resource manager drivers. The elements of this environment communicate with a specified set of standard services.

For more information, see Three-tier architectures.

J2EE platform roles

The J2EE platform also defines a number of distinct roles performed during the application development and deployment life cycle:

- The product provider designs and offers the J2EE platform, APIs, and other features defined in the J2EE specification for purchase.
- The tool provider provides tools used for the development and packaging of application components.
- The application component provider creates Web components, enterprise beans, applets, or application clients to use in J2EE applications.
- The application assembler takes a set of components developed by component providers and assembles them in the form of an enterprise archive (EAR) file.
- The deployer is responsible for deploying an enterprise application into a specific operational environment.

- The system administrator is responsible for the operational environment in which the application runs.

Product providers and tool providers have a product focus. Application component providers and application assemblers focus on the application. Deployers and system administrators focus on the run time.

These roles help identify the tasks and people involved. Understanding this separation of roles is important, because it helps to determine the approach to take when developing and deploying J2EE applications.

For information about the relationship of IBM WebSphere Application Server roles to J2EE roles, see "User roles and activities" on page xii.

J2EE benefits

The J2EE specification provides customers a standard by which to compare J2EE offerings from vendors and develop applications that run on any J2EE compliant platform. Comprehensive, independent Compatibility Test Suites ensure vendor compliance with J2EE standards.

Some benefits of deploying to a J2EE compliant architecture include:
- A simplified architecture based on standard components, services and clients, that takes advantage of the write-once, run-anywhere Java technology.
- Services providing integration with existing systems, including Java DataBase Connectivity (JDBC); Java Message Service (JMS); Java Interface Definition Language (Java IDL); the JavaMail API; and Java Transaction API (JTA and JTS) for reliable business transactions.
- Scalability to meet demand, by distributing containers across multiple system and using database connection pooling, for example.
- A better choice of application development tools and components from vendors providing standard solutions.
- A flexible security model that provides single sign-on support, integration with legacy security schemes, and a unified approach to securing application components.

The J2EE specifications are the result of an industry-wide effort that involves a large number of contributors. IBM has contributed in defining more than 80 percent of the J2EE APIs.

Application components and their containers

The J2EE programming model has four types of application components, which reside in four types of containers in the Application Server:
- Enterprise JavaBeans - Executed by the EJB container
- Servlets and JavaServer Pages files - Executed by the Web container
- Application clients - Executed by the application client container
- Applets - Executed by the applet container

For thorough descriptions of components and containers that apply specifically to the IBM WebSphere Application Server, see "Architectural features" on page ix.

J2EE containers provide the run-time support of the application components. There must be one container for each application component type in a J2EE application.

By having a container between the application components and the set of services, the J2EE specification can provide a federated view of the APIs for the application components.

A container provides the APIs to the application components used for accessing the services. It can also handle security, resource pooling, state management, as well as naming and transaction issues.

Standard services

The J2EE platform provides components with a set of standard services that they can use to interact with each other. See the Sun Web page, http://java.sun.com/products/, for descriptions of each standard service:
- HTTP and HTTPS
- Java Transaction API (JTA)
- Remote Method Invocation/Internet Inter-ORB Protocol (RMI/IIOP)
- Java Interface Definition Language (Java IDL)
- Java DataBase Connectivity (JDBC)
- Java Message Service (JMS)
- Java Naming and Directory Interface (JNDI)
- JavaMail API and the JavaBeans Activation Framework (JAF)
- Java Transaction API (JTA and JTS)
- XML
- J2EE Connector Architecture
- Resource managers

J2EE packaging

Perhaps the most significant change introduced by the J2EE specification is how application components are packaged for deployment.

During a process called *assembly*, J2EE components are packaged into modules. Modules are then packaged into applications. Applications can be deployed on the Application Server. Each module and application contains a J2EE deployment descriptor. The deployment descriptor is an XML file providing instructions for deploying the application.

For more information, including the IBM WebSphere Application Server specifics, see Assembling or packaging.

Architectural features

This document examines the major components within IBM WebSphere Application Server.

HTTP server

IBM WebSphere Application Server works with an HTTP server to handle requests for servlets and other dynamic content from Web applications. (The terms HTTP server and Web server are used interchangeably throughout the documentation.)

The HTTP server and Application Server communicate using the WebSphere HTTP plug-in for the HTTP server. The HTTP plug-in uses an easy-to-read XML configuration file to determine whether a request is handled by the Web server or the Application Server. The HTTP plug-in uses the standard HTTP protocol to communicate with the Application Server though you can configure it to use secure HTTPS, if required. The HTTP plug-in is available for popular Web servers.

For more information, see Configuring Web server plug-ins.

Application Server

The WebSphere Application Server collaborates with the Web server by exchanging client requests and application responses. You can define multiple Application Servers, each running in its own Java virtual machine (JVM).

- **EJB container**

  The EJB container provides the run-time services needed to deploy and manage EJB components, known as enterprise beans. It is a server process that handles requests for both session and entity beans.

  The enterprise beans (inside EJB modules) installed in an Application Server do not communicate directly with the server; instead, an EJB container provides an interface between the enterprise beans and the server. Together, the container and the server provide the bean run-time environment.

  The container provides many low-level services, including threading and transaction support. From an administrative viewpoint, the container manages data storage and retrieval for the contained beans. A single container can manage more than one EJB JAR file.

  For more information, see EJB containers.
- **Web container**

  Servlets and JavaServer Pages (JSP) files are server-side components used to process requests from HTTP clients, such as Web browsers. They handle presentation and control of the user interaction with the underlying application data and business logic. They can also generate formatted data, such as XML, for use by other application components.

  The Web container processes servlets, JSP files and other types of server-side includes. Pre-J2EE servlets run in a servlet engine. Each Web container automatically contains a single session manager.

  When handling servlets, the Web container creates a request object and a response object, and then invokes the servlet service method. The Web container invokes the servlet destroy() method when appropriate and unloads the servlet, after which the JVM performs garbage collection.

  For more information, see Web container.
- **Application client container**

  Application clients are Java programs that typically run on a desktop computer with a graphical user interface (GUI). They have access to the full range of J2EE server-side components and services.

  The application client container handles Java application programs that accesses enterprise beans, Java Database Connectivity (JDBC), and message queues that belong to the Java Message Service (JMS). The J2EE application client program runs on client machines. This program follows the same Java programming model as other Java programs; however, the J2EE application client depends on the application client run time to configure its execution environment, and uses the Java Naming and Directory Interface (JNDI) name space to access resources.

  For more information, see Application clients.
- **Applet container**

  An applet is a client Java class that typically executes in a Web browser, but can also run in a variety of other client applications or devices.

Applets are often used in combination with HTML pages to enhance the user experience provided by a Web browser. They can also shift some of the processing workload from the server to the client.

The applet container handles Java applets embedded in HyperText Markup Language (HTML) documents that reside on a client machine that is remote from the application server. With this type of client, the user accesses an enterprise bean in the Application Server through the Java applet in the HTML document.

For more information, see Application clients.

- **Embedded HTTP server**

  The HTTP handling capability embedded within the application server supports a connection from an HTTP client to the Application Server. An HTTP client can connect to a Web server and the HTTP plug-in can forward the request to the Application Server.

  For more information, see Web container.

- **Virtual host**

  A virtual host is a configuration enabling 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.

  Administrators can use virtual hosts to associate Web applications with a particular host configured to run the application.

  For more information, see Virtual hosts.

Product GUIs and tools

IBM WebSphere Application Server provides a variety of graphical and non-graphical user interfaces. The product GUIs are described, as well as some key scripting and command-line tools. For a complete list of available commands, and for more information about commands, see the section, **Command-line syntax** in the **Quick reference** view of the InfoCenter.

Tools for installing, upgrading, and migrating

Installation wizard

A graphical interface that leads you through the process of installing the product.

Migration tools

Command-line tools such as WASPreUpgrade and WASPostUpgrade, are available to help you migrate from a previous product version.

Tools for developing applications

IBM WebSphere Application Server is not an application development tool, although it provides some Application Programming Interfaces (APIs) for improving the applications you deploy on the server. The WebSphere Studio Application Developer product line offers development environments and tools that complement each edition of the Application Server. For more information, see Product family overview.

Tools for assembling applications

As described in Assembling or packaging, assembly is a necessary packaging and configuration step prior to deploying an application onto the server.

Application assembly tool
The application assembly tool (AAT) assembles enterprise applications for deployment. For more information, see Assembling applications with the AAT.

Deployment tool
The AAT calls this command-line tool to generate code for deployment. In specific cases, you might need to use it. See ejbdeploy tool.

Application Client Resource Configuration Tool
Use this tool to configure deployment descriptors that define the resources needed by application clients. For more information, see Deploying application clients on workstation platforms.

clientUpgrade
Use this tool to migrate client Java archive (JAR) files from the J2EE 1.2 specification to the J2EE 1.3 specification.

Text editor
You can use your favorite text editor to edit deployment descriptors, which are XML documents.

Tools for deploying and administering

As described in Deploying, deploying involves putting an application onto a particular server.

Systems administration tools
See Welcome to System Administration for a description of the available systems administration tools, including the graphical WebSphere Administrative Console, the WSAdmin scripting client, and an assortment of special purpose command line tools.

LaunchClient command
This command line tool starts application clients. See launchClient tool.

XML-SOAP administrative tool
This graphical interface helps you manage deployed Web services.

UDDI user console
This is the graphical interface for interacting with the IBM WebSphere UDDI Registry. See UDDI user console.

NameSpaceDump tool
This command line interface displays the IBM WebSphere Application Server name space for debugging purposes. See dumpNameSpace tool.

Tools for monitoring and tuning

Tools for troubleshooting

Application Server Toolkit
This toolkit provides both local and remote debugging capability for multiple types of application components.

User roles and activities

Review the user role and activity descriptions to understand how someone with your organizational role might use WebSphere Application Server. If you are new to the product or to the Java 2 Platform, Enterprise Edition (J2EE) specification, the simplified model presented here should help you with the basic flow of activities involved in seeing a J2EE application through to deployment.

Setting up and administering a production environment follows roughly these phases.

Using the Application Server in an application development environment follows roughly these phases.

The product tools (described in “Product GUIs and tools” on page xi) and documentation are geared towards helping you with these activities. As you learn more, you will see ways to tailor the flow of activities to your specific needs.

Take advantage of task overviews. Task overviews are special sets of steps in this documentation set. Each outlines a feasible sequence of tasks for working with an area of product functionality, such as security. The tasks typically reflect the main activities, such as Migrating, Developing, Assembling, Deploying, and so on. Use task overviews to gain broad knowledge of the decisions and actions needed to accomplish your goals. From task overviews, you can drill down to more detailed sub-tasks.

For a list of available task overviews, see the **Task overviews** section of the **All topics by activity** view of the online InfoCenter.

Mapping to J2EE roles. The following is the mapping of WebSphere Application Server roles and activities to the roles defined by the J2EE 1.3 specification.

J2EE roles	Product roles	Product activities
Non-applicable	Planner, Installer, IT architect	Planning, Installing product environment
Application Component Provider	Developer / Programmer	Developing
Application Assembler	Developer / Programmer	Assembling
Deployer	Administrator (in production environment)	Deploying, Testing application deployment
System Administrator	Administrator	Administering
Non-applicable	All of the above	Migrating, Tuning, Troubleshooting
Tool Provider, J2EE Product Provider	WebSphere Application Server and product family	Non-applicable

InfoCenter

The InfoCenter displays the documentation for multiple releases of IBM WebSphere Application Server. From theWebSphere software Web page you can select and view information about the different releases of the Application Server, including additional formats for the information contained in the InfoCenter.

Navigational views

The left hand frame, commonly called the navigation frame, contains ″books″, which house the expandable table of contents. The table of contents are revealed by clicking on the ″book″ of choice. The table of contents for each book consist of various navigational views, including:

- **All topics by feature.** This hierarchical view organizes the topics according to the main technologies in the product domain, such as Web services and J2EE resources. This view is similar to the navigation in the Administrative Console.
- **All topics by activity.** This hierarchical view organizes topics according to tasks that might already be familiar to you from working with other software.

  The top-level categories in this view outline the main phases, from application development through deployment and updates. The organization in the next couple of levels of this view reflects that of the **All topics by feature** view.
- **Quick reference.** This fairly flattened view highlights reference information for quick lookup -- such as confirming the syntax for a particular command line argument.

  The Quick reference view makes every reference topic accessible in two clicks, for quick retrieval. For example, you can look up syntax for the addNode command by clicking **Commands**, then clicking the name of the documentation topic -- addNode command. In contrast, the other navigational views require three or more clicks to view many of these topics, which tend to reside deeper in the document hierarchy.

  This view includes sets of links to additional sources of information, outside of the InfoCenter. These links can include IBM Redbooks, Best Practice documents, WebSphere Developer Domain topics, and a variety of others.

Information type indicators

Each document that is displayed in the right-hand frame of the InfoCenter is known as a *topic* or article. Most topics belong primarily to one of the following *information types*:

Concept	Concept topics answer the question, ″What is...?″ They explain what something is and why it behaves the way it does.
Task	Task topics answer the question, ″How do I...?″ Some tasks lead you through a thought process, to arrive at a decision. Other tasks lead you through a detailed series of cursor clicks and keystrokes to produce a tangible result, such as updating and saving a configuration file. Some tasks lend themselves to sequential steps; others do not. When there are *many* valid sequences for performing a set of steps, topics in the InfoCenter typically present *one* of the valid sequences. This simplification helps novice users learn and complete tasks successfully. It is assumed that more advanced users will know or easily discover alternative sequences, adapting the documented sequence to their needs.

Reference	Reference topics provide quick access to facts. For example, Javadoc is a type of reference information from which application developers look up method signatures for a specific Application Programming Interface (API). The Javadoc for this product is organized alphabetically by package name. As with a telephone directory, reference information is not intended to be read in large chunks. Reference topics -- and the entries within reference topics -- are organized for quick lookup, rather than to highlight their interrelationships. For example, two entries in the **Commands** section of the **Quick reference** view might have little in common, except that they both are valid commands that you can type at a command prompt. Based on alphabetization, the two otherwise unrelated items might be listed one directly after the other. It is anticipated that you might look up one of these commands on one occasion, perhaps needing the other one on entirely different occasion.
Help file	These topics are the help files for the graphical user interfaces of the product. They are a mixture of concept, task, and reference information. One copy of these files is installed with the product, enabling you to view these topics by clicking a **Help** link or button. Another copy is included in the InfoCenter. The InfoCenter version of these files can contain updates that have not been included in the installed help files yet.

To use the product documentation effectively, you need **not** understand or rely upon the information types. Their primary purpose is to make the information more predictable and consistent, whether or not you notice their contribution.

Search

A full text document search is available from the banner of the online InfoCenter. In addition, an advanced search option is available. The advanced search options allows for:

Search criteria	Search result
*	any string
?	any single character
""	exact phrase

AND, OR, NOT	boolean operators xx AND yy - hits that contain both xx, yy xx OR yy - hits that contain either xx, yy NOT xx - hits that do NOT contain xx
book selection checkbox	only books with a checkmark will be included in the search

Article IDs

To find the file name for a particular page in the InfoCenter, position your cursor over the title of the article in the navigation frame. Right click on the title of the article and select properties. The filename, along with other file properties will be displayed in resulting window. You can also search on this file name.

For example, if you pull up the properties window for the Launching scripting clients topic, and find the file name is `txml_launchscript`. To help someone else to find that topic, you can send the person the file name, which the person can enter into the InfoCenter search window to locate the topic.

Bookmarks

Take advantage of your Web browser capabilities to bookmark (or add to ″favorites″) a topic that is displayed in the right-hand frame of the InfoCenter.

1. Right-click anywhere in the right-hand frame displaying the InfoCenter topic you want to bookmark.
2. Select **Add Bookmark** if you are using Netscape Navigator. Select **Add to Favorites** if you are using Microsoft Internet Explorer.

Font size adjustment

A cascading style sheet (CSS) governs the appearance of the InfoCenter. However, the text size remains flexible. The text is displayed in the size indicated in your Web browser settings.

The instructions for changing the text or font size vary between browser brands and versions. Consult the browser product documentation for instructions. In Microsoft Internet Explorer, look for the menu choice, **View > Text Size**. In Netscape browsers, look for the menu choice, **View > Increase Font Size**.

Browser support

The InfoCenter supports Netscape Navigator 6.0 and Microsoft Internet Explorer 6.0 and higher. In addition, JavaScript must be enabled in the browser. If you use a different or earlier version of a browser, your pages may format differently and some functions may not work correctly.

Printing

The PDF versions (available separately from InfoCenter) are recommended if you would like to print multiple topics with the intent of reading the topics in sequence, like a book.

Browser capabilities may be used to print single topics from the online InfoCenter. To print a topic displayed in the right-hand frame of the InfoCenter, first click anywhere in the right-hand frame to register that frame as the selected frame. Then use your browser to print the selected, right-hand frame.

The online version is subject to change frequently. Timestamps at the bottom of each topic help you keep track of when you printed the information.

Introduction to this book

All of the documentation for WebSphere Application Server for z/OS is organized by task.

Task overviews

Task overviews are special sets of steps in this documentation. Each outlines a feasible sequence of tasks for working with an area of product functionality, such as security. Use task overviews to gain broad knowledge of the decisions and actions needed to accomplish your goals. From task overviews, you can drill down to more detailed sub-tasks.

The tasks in a task overview typically reflect the main activities, such as Migrating, Developing, Assembling, Deploying, and so on.

For learning purposes, task overviews outline one feasible sequence of tasks. In reality, several sequences might work. As you learn more, vary the sequence for your needs.

This book describes the planning, migrating, and installing tasks.

Chapter 1. Installing WebSphere Application Server

Perform the following tasks to install the product on your machine.

1. Plan to install an e-business network.

   This task helps you plan for installing an e-business network. It also describes interoperability considerations.
2. Install a new release of WebSphere Application Server for z/OS.

   This task helps you maintain availability of sysplex systems when installing a new release or maintenance level of WebSphere Application Server for z/OS.
3. Install the product.

   This task helps you prepare your machine for installation and explains the different types of installation available to you. It includes information on:
   - Using the Customization Dialogues.
   - Deciding whether to migrate applications and the configuration from a previous version
   - Deciding whether to coexist with a previous version
4. Verify the WebSphere Application Server installation.
   a. Run the installation verification test (IVT).

      Although you can select the installation wizard from the First Steps panel that appears at the end of installation, you can also start the installation verification test (IVT) by running the **BBOWIVT** program.
5. Examine the results of migration, or prepare to perform a manual migration.

   This task describes exactly what is migrated during the automatic migration. It also describes how to perform a manual migration using the migration tools.
6. Set up a multinode environment.

   This task describes how to set up multiple nodes into a group, or cell, of Application Servers, with centralized configuration under the control of the deployment manager node and its distributed node agents.
7. Set up V4.0.x and V5 coexistence.

   This task describes running WebSphere Application Server V4.0.1 and V5 on the same machine.
8. Automatically restart WebSphere Application Server processes.

   This task describes how to set up WebSphere Application Server processes for the operating system to monitor and restart.
9. Create multiple Version 5 instances on one machine.

   This task describes creating multiple configuration instances from one installation. It also describes creating multiple servers in a coexistence or multiple instance environment, and changing port settings for HTTP transport to avoid conflicts.
10. Install service to WebSphere Application Server for z/OS.

    This task helps you apply maintenance to WebSphere Application Server for z/OS.

The WebSphere Application Server product is installed.

Tuning performance: For best performance on any platform, see Tuning performance .

WebSphere Application Server packages

The WebSphere Application Server family of interoperable products provides a next-generation application server on an industry-standard foundation. The IBM WebSphere Application Server family is divided into five packages for Version 5, to better address requirements you might have. Each edition addresses a distinct set of scenarios and needs. WebSphere Application Server includes:

- **WebSphere Application Server Express** (not pictured below)

  This edition is a lightweight server for static content, servlets, and JSP pages, but does not support enterprise beans.
- **WebSphere Application Server**

  This edition addresses the basic programming and execution needs of desktop developers and single-server production scenarios. The execution environment for this edition addresses standards-based programming for Web and component-based programming, as well as Web services.

  The administration model for this edition presumes a single-server environment - no clustering for failover or workload balancing, nor centralized administration of multiple server instances. However, you can add a stand-alone node to a centrally administered network (the cell) at any time after installing the next product, which controls the cell.
- **WebSphere Application Server Network Deployment**

  This edition addresses Application Server execution in departmental computing scenarios. It provides centralized administration of multiple server instances, as well as basic clustering and caching support.
- **WebSphere Application Server Enterprise**

  This edition addresses large information technology (I/T) production scenarios for applications that are designed according to the Java 2 Platform, Enterprise Edition (J2EE) programming model and Web services. The J2EE specification supports large-scale clustering, caching, content distribution, and dynamic workload management to help gain efficient utilization of shared resources in the I/T computing center. The J2EE specification also addresses complex programming requirements for integrating applications and lines of business by introducing super-standard programming functions for business process management, integration adapters, rules management, and message transformations.
- **WebSphere Application Server for z/OS** (not pictured)

  This edition integrates the WebSphere Application Server product and the Network Deployment product into a single package that runs on a z/OS platform. It addresses application server execution in departmental computing scenarios, and provides centralized administration of multiple server instances, as well as basic clustering and caching support.

The WebSphere software platform for e-business starts with a foundation formed from Web application serving and integration. The WebSphere Application Server family lets you quickly, reliably and flexibly enable your business for the Web. It provides the core software to deploy, integrate and manage your e-business applications. WebSphere Application Server supports custom-built applications, based on integrated WebSphere software platform products, or on other third-party products. Such applications can range from dynamic Web presentations to sophisticated transaction processing systems.

IBM WebSphere Application Server, WebSphere Application Server Network Deployment, and WebSphere Application Server Enterprise are interoperable building blocks. The base product CD-ROMs are included in the Network Deployment package. The Network Deployment package CD-ROMs are included in the Enterprise package.

- The **Clients** installation image contains support for Java (J2EE) Application Client 2, CORBA and ActiveX client run times.
- The **WebSphere Application Server** product installation image contains the core application server runtime, a native JMS provider (the embedded messaging feature), IBM HTTP Server, IBM Developer Kit, IBM Cloudscape, XML and XSL parsers, the application assembly tool (AAT), the deployment tool, the node agent for communicating to the deployment manager when it is part of a cell, and the external adapter library for proxy caching enablement.
- The **Deployment Manager** product installation image contains the deployment manager configured for use in departmental production computing scenarios. It includes the embedded messaging client feature, the Version 2 compliant universal description, discovery, and identification (UDDI) registry feature, and the Web services gateway feature. Although the Network Deployment package includes the base Application Server CD-ROM, installing the Network Deployment product does not install the base Application Server product. You must use the base product CD-ROM to install the base product.
- The **Edge components** installation image contains IBM HTTP Server, and edge of network support for the Load Balancer (Dispatcher) and Caching Proxy (edge caching), as well as support for network authentication and single sign-on.
- The **Enterprise** installation image contains programming model extensions to the core Application Server, such as Business Rule Beans and Business Policy Management, and deployment manager extensions for administering functions included in the programming model extensions. The Enterprise product supports an *umbrella installation*, which automatically installs the base product in the same installation procedure.

  Although the Enterprise package includes the Network Deployment product CD-ROM, installing the Enterprise product does not install the Network Deployment product. You must use the Network Deployment product CD-ROM to install the Network Deployment product.

  If you install the Enterprise product on a machine with an installed Network Deployment product, the Enterprise product extends the Network Deployment administrative console.

  Although the Enterprise package includes a separate base WebSphere Application Server product CD-ROM, installing the Enterprise product on a clean machine automatically installs the base Application Server product in an umbrella installation.

  If you install the Enterprise product on a machine with an installed base Application Server product, the Enterprise product extends the base Application Server product, while automatically installing any base product features that the Enterprise product requires.
- The **WebSphere MQ and WebSphere MQ Event Broker** installation images provide the non-embedded full-function WebSphere MQ Queue Manager for reliable, dynamically load balanced asynchronous messaging for more than 35 platforms.

Included with all packages are the Data Direct Technologies JDBC Drivers and the IBM WebSphere Application Server Toolkit (ASTK) CD-ROMs.

Also included with the base product and the Network Deployment product are the WebSphere Application Server, Version 5 for iSeries CD-ROMs. The Enterprise product is not supported on iSeries servers at this time.

In addition, the WebSphere Application Server Network Deployment and Enterprise packages include the IBM Directory Server and the DB2 Universal Database Enterprise Edition products.

Planning to install an e-business network

Version 5 of the WebSphere Application Server family provides flexible configurations and deployment options for hosting Java 2 Platform, Enterprise Edition (J2EE) applications. The WebSphere Application Server package provides all required components:

- The WebSphere Application Server
- A Web server
- A J2EE client

The WebSphere Application Server for z/OS package provides a smooth transition to deploying and managing applications in a distributed network environment. It includes productivity and scalability enabling components (Edge Components). The deployment manager server lets you easily federate single nodes to its group, which it manages as a single image or *cell*, by using the nodeagent server process on each node.

Tasks in the installation planning process appear below.

1. Plan the general scope of your network.

   Review single machine topologies and how to establish a multimachine environment to learn about WebSphere Application Server, Version 5 product family scalability. With one command you can move an Application Server node from a stand-alone environment into a managed group of server processes, all part of a single-image cell, under the centralized control of the Version 5 deployment manager. Or, you can move an Application Server back to a stand-alone environment just as easily. You can start small and *scale up* to larger integrated topologies easily, with the seamless architecture and packaging of the WebSphere Application Server, Version 5 product family.
2. Review common multinode topologies.

   Determine which scalable and reliable e-business environment is a best fit for your requirements as you learn where to install the typical components of a WebSphere Application Server environment. For example, decide whether to install WebSphere Application Server on the same server as the Web server, or whether to create a cluster of application servers on one or several machines.

   Decide which WebSphere Application Server package you need. Understanding scalability factors and picking the right design can help you start with the correct package, and can help you achieve your business model objectives for e-business in a timely and cost-efficient manner.
3. Review common topologies for WebSphere Application Server Edge Components, as described in the InfoCenter for Edge Components.

   After planning how to install the WebSphere Application Server product, you can plan the installation of Edge Components, which are included in the Network Deployment package and therefore, in the Enterprise package. Edge Components include the Caching Proxy and the Load Balancer component set, which includes components, such as the Dispatcher.

   You can find a link to the Edge Components InfoCenter Web site in the *Installation: Resources for learning* topic.

4. Prepare the base z/OS environment. The base z/OS environment and all of the subsystems associated with z/OS must be able to support your WebSphere installation.
5. Plan for interoperability and coexistence.

   Plan to have WebSphere Application Server interoperate with your other e-business systems, including other versions of WebSphere Application Server. *Interoperability* provides a communication mechanism for WebSphere Application Server nodes that are at different versions. *Coexistence* is another term in common use in this InfoCenter. It describes multiple versions or instances running on the same machine, at the same time.

   Interoperability support enhances migration scenarios with more configuration options. It often is convenient or practical to interoperate during the migration of a configuration from an earlier Application Server version to a later one when some machines are at the earlier version and some at the later one. The mixed environment of machines and application components at different software version levels requires interoperability and coexistence.

   It is often impractical, or even physically impossible, to migrate all the machines and applications within an enterprise at the same time. Understanding multiversion interoperability and coexistence is therefore an essential part of a migration between version levels.

   a. Plan to run WebSphere Application Server across platforms.

      Support of multiple versions is provided on all operating system platforms supported by WebSphere Application Server, Version 5.

   b. Plan to run WebSphere Application Server across versions.

      WebSphere Application Server, Version 5 is generally interoperable with WebSphere Application Server Versions 3.5.x and 4.0.x, although each version has specific requirements. However, the ability to run different versions of an Application Server in a configuration does not let you include Version 5 Application Servers in an existing administrative domain, or let you include Version 3.5.x or Version 4.0.x Application Servers in a Version 5 cell.

Reading the information referred to in this topic provides you with a good approach for implementing your WebSphere Application Server environment.

You are now ready for pre-installation planning and installation. Go to Installing WebSphere Application Server to continue.

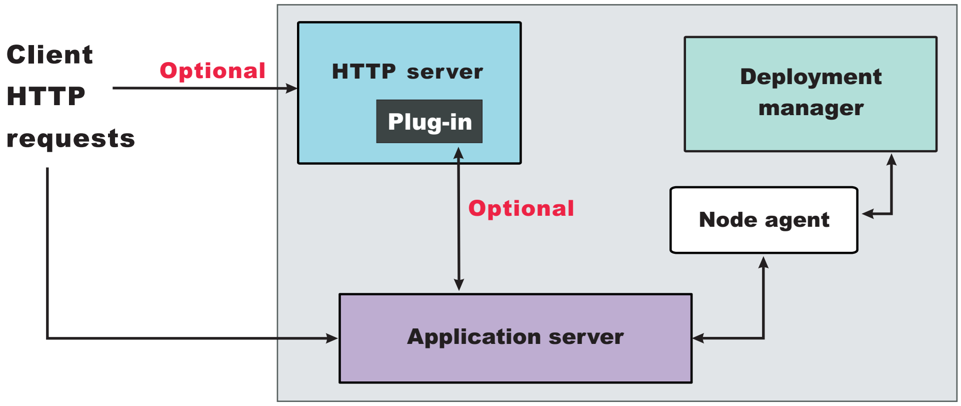
Single server topology

The following illustrations show examples of single server topologies. Each WebSphere Application Server product can run in a single server environment. The most common topology is a stand-alone base WebSphere Application Server product.

IBM WebSphere Application Server for z/OS integrates the WebSphere Application Server product and the Network Deployment product into a single package. In a case such as this, the term product really equates to function.

The WebSphere Application Server base product runs on a single machine. You can install the product in a stand-alone configuration or as part of a cell in a multimachine configuration. The stand-alone configuration is typically for developer desktops or stand-alone production computing, which involve a single Application Server instance operating independently of any other applications.

The Network Deployment function can also run on a stand-alone machine or in a multimachine configuration, as described in other topology topics. The following illustration shows a typical developer environment for Network Deployment function in a stand-alone configuration.

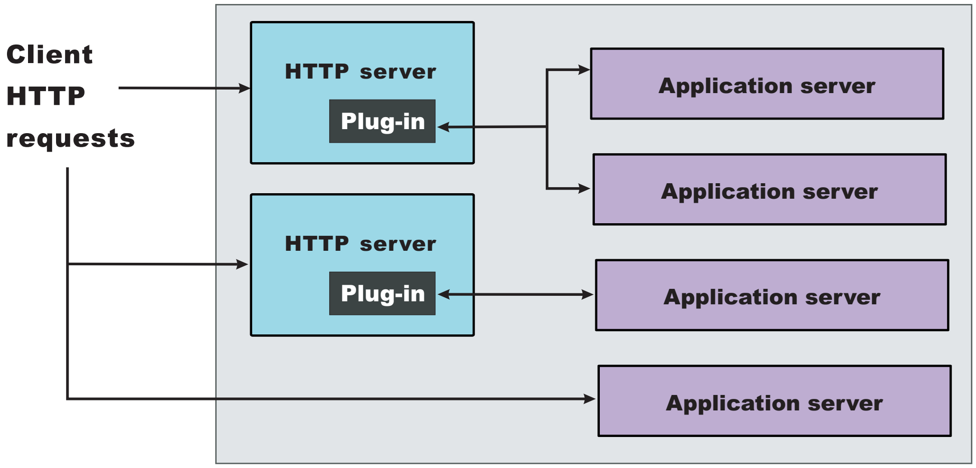


Multiple instances on one machine

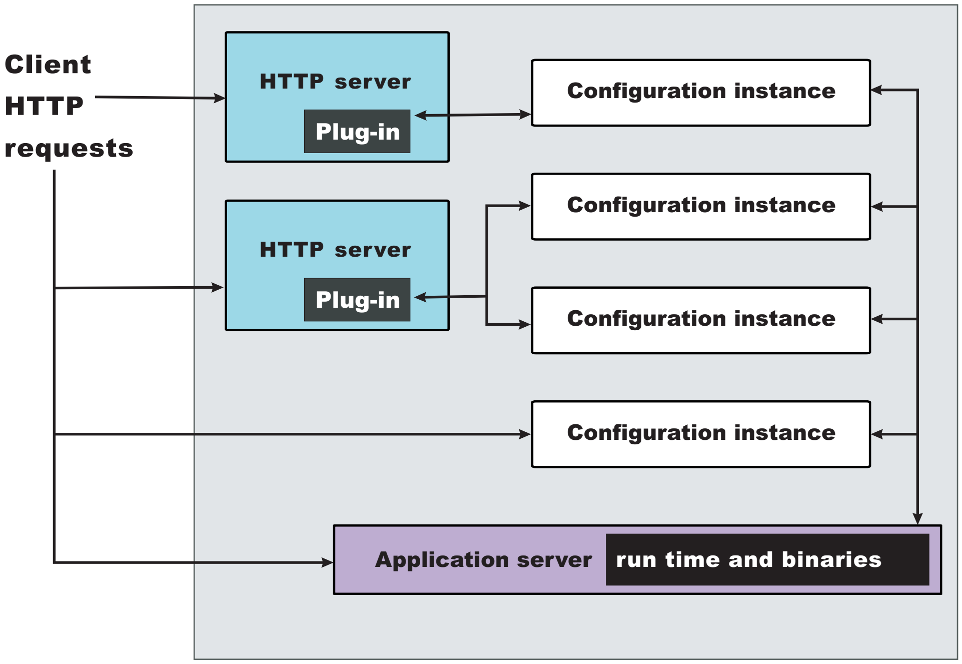
There are three main topologies for the base WebSphere Application Server product:

- A single installation, as described above
- Multiple installations in a coexistence environment
- A single installation with multiple configuration instances

You can install the base WebSphere Application Server product multiple times in separate directories. Each resulting installation instance is a fully functional application server. The following illustration shows an example of such a configuration.



You can also install the base WebSphere Application Server product one time and use the **wsinstance** command to create multiple configuration instances. Configuration instances are fully functional Application Servers that share the run time and command binaries of the initial product installation. The following illustration shows an example of such a configuration.



Establishing multimachine environments

Setting up a Network Deployment environment involves several steps performed on each of the computers that comprise the cell.

1. Install and customize the base WebSphere Application Server function on each system to create an Application Server node on the system.

   Install and customize the base WebSphere Application Server function on a system for it to become a node in the cell. The installation and customization procedure is the same for a node to be federated into a cell as it is for a stand-alone Application Server. You can install and customize the base WebSphere Application Server function more than once on a single machine.

   Coexistence is supported.

   You can install and customize the base WebSphere Application Server function once and create multiple configuration instances on the machine, using the **wsinstance** command. You can also install the Network Deployment function once and create multiple configuration instances of the deployment manager. Each deployment manager configuration instance can federate stand-alone base WebSphere Application Server product installation instances, but a deployment manager cannot federate base product configuration instances.

   Migrate applications, security settings, and the remaining configuration from WebSphere Application Server, Version 4.0.x and later. You can also choose to coexist with WebSphere Application Server, Versions 4.0.x and later. Both migration and coexistence are described in the installation and customization procedure for the base WebSphere Application Server function, which is available from the InfoCenter for the WebSphere Application Server for z/OS.
2. Customize the WebSphere Application Server Network Deployment function on a machine. Only one system hosts the deployment manager.

As the deployment manager federates base WebSphere Application Server nodes, it expands the cell that it manages. The deployment manager is the central administrative manager.

Migrate applications, security settings, and the remaining configuration from WebSphere Application Server, Version 4.0.x and later. You can also choose to coexist with WebSphere Application Server, Versions 4.0.x and later. Migration and coexistence are described in the installation procedure for the WebSphere Application Server Network Deployment product.

3. Start the deployment manager process.

   There are two ways to start the deployment manager:
   - As a monitored process, which restarts automatically if a failure occurs
   - As an unmonitored process, which is what the **startManager** command does

   Run the **startManager** command from the `/bin` directory of the installation root of the deployment manager.

   For production systems, running the deployment manager as a monitored process is recommended.
4. Run the `addNode.sh` (or `addNode.bat`) script on every node that you plan to federate into the cell.

   The `addNode.sh` (or `addNode.bat`) script incorporates a base WebSphere Application Server product node into a deployment manager cell. You must run this tool on every system that you plan to make part of a Network Deployment cell. There are several parameters for the **addNode** command, but the most important are `includeapps`, the host name of the deployment manager node, the JMX connector type, and the JMX port of the deployment manager node.

   For example:

   ```
   addnode wasdoct 8889 -includeapps
   ```

   The example adds the base node on which the command runs to the cell managed by the `wasdoct` deployment manager node, using the default SOAP JMX connector type at port 8889. (Port 8889 is a coexistence value for the SOAP port. On the `wasdoct` development machine, the base node uses the default SOAP port.) The command includes all applications on the base node into the cell.

   Alternatively, you can use the administrative console of the deployment manager to add running Application Server nodes to the cell.
5. Enable the appropriate level of security after the installation is complete.
6. Develop and unit test application components.

   Load existing application components and modules into your development environment and debug them.
7. Assemble code into a main application module or enterprise archive (EAR) file.
8. Start all servers in the test environment.
9. Deploy your applications in the test environment.
10. Test all applications thoroughly.

    Follow normal test procedures as you move the test environment into production. Review the information in the Migrating topic to understand what you must look for. In particular, review the table at the end of the topic that links you to specific recommendations and practices.

You must configure your migrating applications to ensure that they migrate in the way you want.

11. Prepare and monitor the environment into which you deploy applications.
12. Adjust application code, configurations, and system settings to improve performance.
13. Fix any known problems.
14. Set up your production system by configuring all server processes for monitoring by their operating systems.

You can create a working WebSphere Application Server cell.

Use the administrative console or other administrative tools to observe and control the incorporated nodes, and the resources on these nodes. The console provides a central location for configuring, monitoring, and controlling all Application Servers on all nodes within the cell.

Setting up a multinode environment

A multinode environment places WebSphere Application Server processes on separate physical machines, under the central management of the deployment manager process, which groups Application Servers into its managed *cell*. This example topology shows how you can have a tier of Application Servers within the cell in addition to traditional tiers that can contain the Web server, databases, enterprise information systems, and other types of persistent storage.

The following illustration shows an example of multiple nodes and multiple tiers.

In this example, Web container cluster members (Machine C) are closer (in a network sense) to the HTTP server (Machine B), which improves their response to client requests. Application server processes that run enterprise beans (Machine D) are closer in network terms to application data, which is represented in an application by entity beans and stored in a database on the data tier.

Clustered Application Servers on Machines C and D help maximize resource use on each machine. Although the example shows two cluster members in each node, you can have as many as a machine can support. Cluster members in a multitiered topology provide process redundancy and use memory more efficiently than in similar topologies that host only single instances of Application Servers. Additional resources on the machines can improve application throughput and performance.

Introducing firewalls between each pair of tiers can provide the same level of security for entity beans as for application data.

A multitiered topology within the cell eliminates local Java virtual machine (JVM) optimizations that occur when the same cluster member runs both the Web container and the EJB container. This topology also introduces network latency, which tends to slow system performance. Although multitiered topologies provide more redundancy for Application Server processes, they also introduce more possible points of failure. The level of redundancy can also complicate maintenance.

The deployment manager on Machine A manages the configuration of cluster members and other Application Server processes on Machines C and D. The deployment manager coordinates all Application Server processes through node agent processes, each of which runs as the `nodeagent` server on a node.

Setting up environments such as those shown in the illustrations involves performing several steps on each computer comprising the deployment manager cell.

1. Customize the Network Deployment function on Machine A, to make it the deployment manager node, using the WebSphere Application Server for z/OS Customization Dialog and, where applicable, the administrative console. Machine A is the system for the deployment manager server (dmgr), which provides a centralized administrative console for the entire group, or cell, of application servers that it controls. Customizing the Network Deployment function does not install or customize the base Application Servers. The WebSphere Application Server for z/OS Customization Dialog provides the basis for configuring the base Application Server, the integral JMS providers, and the Network Deployment function and its associated administrative programs.
2. Install and customize the base WebSphere Application Server function on Machines C and D, to make them Application Server nodes. Aside from the one machine where you configured the deployment manager node, you must customize the base WebSphere Application Server function on other computers that are to comprise the cell. The installation and customization process is the same for an Application Server (server1) that you federate into a cell as it is for a stand-alone Application Server.
3. Start the deployment manager process on Machine A.

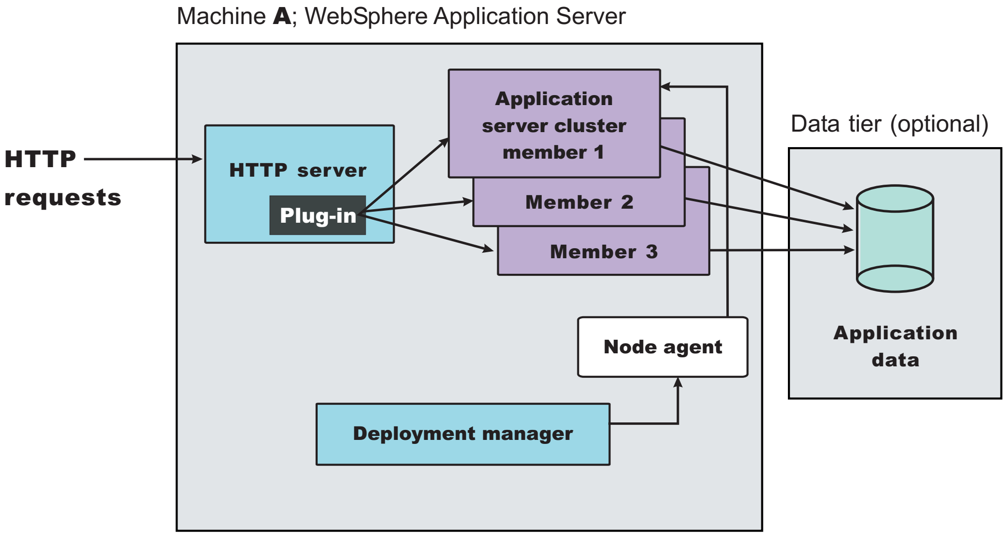
   The **startManager** command line tool is one way.
4. Run the **addNode** command line tool on every node that you intend to incorporate into the cell, such as Machine C and Machine D.

Use the administrative console or other administrative tools to observe and control the federated nodes and the resources on those nodes. The console provides a central location for configuring, monitoring, and controlling all Application Servers on all nodes within the cell.

Return to Planning to install an e-business network to continue.

Vertical scaling topology

Vertical scaling refers to setting up multiple application servers on one machine, usually by creating cluster members.



This topology illustrates a simple vertical scaling example, with multiple cluster members of an Application Server on Machine A. You can also implement vertical scaling on more than one machine in a configuration. Combine vertical scaling with other topologies to boost performance and throughput.

Typical use

Vertical scaling offers the following advantages:
- **Increased processing power efficiency.** An instance of an application server runs in a single Java virtual machine (JVM) process. However, the inherent concurrency limitations of a JVM process prevent it from fully utilizing the processing power of a machine. Creating additional JVM processes provides multiple thread pools, each corresponding to the JVM process associated with each Application Server process. This correspondence avoids concurrency limitations and lets the Application Server use the full processing power of the machine.
- **Load balancing.** Vertical scaling topologies can use the WebSphere Application Server workload management facility.
- **Process failover.** A vertical scaling topology also provides failover support among Application Server cluster members. If one Application Server instance goes offline, the other instances on the machine continue to process client requests.

Single machine vertical scaling topologies have the drawback of introducing the host machine as a single point of failure in the system. Vertical scaling on multiple machines avoids the single point of failure.

Instructions

To set up a vertical scaling topology, use the administrative console to configure a set of Application Server cluster members that reside on the same machine.

It is recommended that you plan vertical scaling configurations ahead of time. However, because vertical scaling does not require any special installation steps, you can implement vertical scaling whenever it is needed.

While you are deciding how many cluster members to create on a machine, take these factors into account:

- **The design of the application.** Applications that use more components require more memory, limiting the number of cluster members you can run on a machine.
- **The hardware environment.** Vertical scaling works best with plenty of memory and processing power. Eventually there is a point of diminishing returns on any machine, where the overhead of running more cluster members cancels out the benefits of adding them.

The best way to ensure good performance in a vertical scaling configuration is to tune a single instance of an Application Server for throughput and performance, then incrementally add cluster members. Test performance and throughput as you add each cluster member. Always monitor memory use when you are configuring a vertical scaling topology, so you do not exceed available physical memory on the machine.

Multimachine topology concepts

Multiple machine environments extend basic single machine WebSphere Application Server configurations by distributing the Application Server over multiple machines, increasing the overall processing power from one machine to contributions from all machines in the configuration.

The flow of data in a WebSphere Application Server environment starts with a Web server receiving requests and routing them to the Application Server for processing. A WebSphere Application Server node stores administrative configuration data in XML files. A database can hold application data for applications that require a place to store data, such as user session information. There are also one or more administrative clients, such as the administrative console, for manipulating configuration data.

Some of the reasons for creating WebSphere Application Server applications that run on multiple machine systems include:

- **Scalability**. Adding more machines increases processing power, which scales up the system to handle a higher client load than that provided by a basic, single machine configuration. Scalability pertains to the capability of a system to adapt readily to a greater or lesser intensity of use, volume, or demand. For example, a scalable system can efficiently adapt to work with larger or smaller networks performing tasks of varying complexity. Ideally, it is possible to handle any given load by adding more servers and machines, assuming each additional machine processes its fair share of client requests. Each machine should process a share of the total system load that is proportional to the processing power of the machine.

  Consider these primary scalability factors when adopting a WebSphere Application Server topology:

  - **Security**. Address certain security concerns by physically separating the Web server from the Application Server, by using firewalls.
  - **Performance**. Maximize performance by ensuring the response time for transactions is as short as possible. You can use two general topologies to improve transaction performance:
    - *Vertical scaling*, in which you create additional Application Server processes on a single physical machine. The *Vertical scaling sample topology* topic describes a physical implementation for vertical scaling.
    - *Horizontal scaling*, in which you create additional application server processes on multiple physical machines to take advantage of the additional processing power available on each machine. You can also use WebSphere Application Server Edge Components, such as the Caching

Proxy Edge component, and the Load Balancer component set (which includes the Dispatcher component), to implement horizontal scaling. The *Horizontal scaling with cluster members sample topology* and *Horizontal scaling with Network Dispatcher sample topology* topics describe physical implementations for horizontal scaling.

- **Throughput**. Add Application Server clusters to scale vertically or horizontally. Application server clusters can increase the number of concurrent transactions that the application can perform, to help process as many transactions as possible within a given time period.
- **Availability and failover support**. Avoid a single point of failure and maximize system availability by ensuring that the topology has some degree of process redundancy. High-availability topologies typically involve horizontal scaling across multiple machines. Vertical scaling can improve availability by creating multiple processes, but the machine becomes a point of failure.

  A Dispatcher server performs intelligent load balancing to determine where to send a TCP/IP request. It can direct client HTTP requests to available Web servers, bypassing any that are offline. Another server can back up the Dispatcher server, to eliminate it as a single point of failure. Workload management of Application Servers and administrative servers also improves availability and failover support.

  Failover support distributes client requests to the remaining servers, which ensures continued client access without significant interruptions. (In practice, failover is not entirely transparent to clients.)

  WebSphere Application Server supports these methods of ensuring availability with multiple machines and applications:
  - **Multiple tiers** The components of an application (the Web server, Application Servers, databases, and so forth) are physically separated on different machines.
  - **HTTP server separation** The Web (HTTP) server is located on a different physical machine than the Application Server. You can redirect requests to Application Servers through a variety of methods.
  - **Demilitarized zone (DMZ)** Firewalls create demilitarized zone machines, which are isolated from both the public Internet and other machines in the configuration. The DMZ scales security processing, which improves security and throughput in the application environment.
  - **Multiple cells** Multiple WebSphere Application Server cells provide failover for clustering Application Servers and deploying applications.
  - **Multiple applications** Multiple application instances can be on the same physical machine, or on more than one machine in the cell.
- **Maintainability**. Understand that the topology affects the ease with which you can update system hardware and software. For instance, using multiple WebSphere Application Server deployment manager cells or horizontal scaling can make a system easier to maintain because you can take individual machines offline without interrupting other machines, running the application.

  Maintainability sometimes conflicts with other topology considerations. For example, limiting the number of Application Server instances makes the application easier to maintain but can have a negative effect on throughput, availability, and performance.
- **Maintaining session state between client HTTP requests**. Consider that session state is important for stateful applications, or for applications that run on multiple machines or Application Server instances. You can share a session between multiple Application Server processes (cluster members) by saving the session state to a database. In addition, configuring a network dispatcher

affects how the session state is maintained. This consideration does not apply if your application runs on a single Application Server instance or is completely stateless.

- **Shared data access**. Placing backend resources, such as databases, on different machines provides ease of use when sharing these resources.
- **Fault isolation**. Providing more robust failover support through a configuration that includes a degree of fault isolation, reduces the potential for failure of one server to affect other servers. Configurations that provide simple failover support are concerned only with individual server failures that have no effect on the performance of other servers. However, in some situations, a malfunctioning server can create problems for other servers that are otherwise functioning normally. For example, a failing server can consume more than its share of system and database resources, preventing other servers from gaining adequate access to these resources. You can configure WebSphere Application Server to provide fault isolation between different parts of a system.
- **Dynamic changes to configurations**. Modifying the system configuration without interrupting its operation enhances the manageability and flexibility of the system. For instance, administrators can add or remove cluster members to handle variations in the client load, change server characteristics and propagate the changes to its cluster members, temporarily stop servers for maintenance, and so forth.
- **Mixed Application Server versions**. Migrating a few machines and applications at one time is possible on certain multiple machine configurations. In addition to Version 5 Application Servers, your server configuration can include earlier version Application Servers, where you currently deploy all applications you intend to migrate. You can migrate applications to Version 5 and deploy them in stages. You can also easily upgrade system hardware and software. When combined with the ability to make dynamic changes to the configuration, you can use a configuration of mixed Application Server versions to upgrade an application or machine without any interruption of service.

  The ability to run different versions of an Application Server in a configuration does not let you include Version 5 Application Servers in an existing administrative domain, nor does it let you include Version 3.5.x or Version 4.0.x application servers in a Version 5 cell.

You can configure multiple machines to add processing power, improve security, maximize availability, and balance workloads. WebSphere Application Server Network Deployment and WebSphere Application Server Edge Components provide clusters, workload management, and the Dispatcher to implement configurations that address these issues. These scaling techniques are generally combined to maximize benefits and minimize problems associated with multiple machine systems.

There are various ways to scale up the basic, single-machine WebSphere Application Server system to meet the needs of your organization. Some scaling techniques include:

- **Clusters**. Clusters are identical multiple Application Server copies. A cluster member is one Application Server in a cluster. The configuration of each cluster member is based upon the Application Server that you copied to create the cluster member. You can create all cluster members on the same physical machine, or on different machines. Using cluster members can improve the performance of a server, simplify its administration, and enable the use of workload management.
- **Workload management (WLM)**. WLM enables both load balancing and failover, improving the reliability and scalability of applications deployed on the

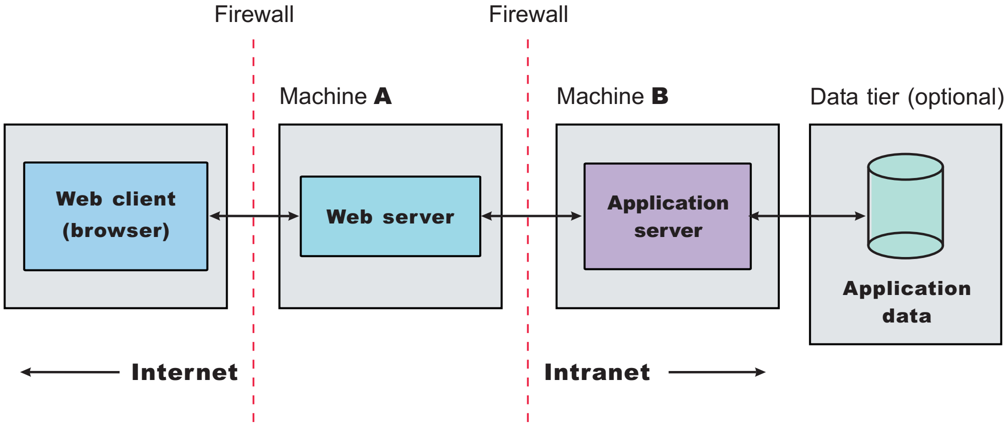
WebSphere Application Server server. Incoming processing requests from clients are transparently distributed among the cluster members of an Application Server.

- **Dispatcher**. The Dispatcher transparently redirects incoming HTTP requests from Web clients to a set of Web servers. Although the clients behave as if they are communicating directly with a given Web server, the Dispatcher is actually intercepting all requests and distributing them among all the available Web servers in the cluster. The Dispatcher can provide scalability, load balancing, and failover for Web servers.

Keep in mind that the techniques described above are not mutually exclusive. You can combine their basic elements in various ways.

Firewalls and demilitarized zone configurations: *Firewalls* protect backend resources, such as databases in multiple machine systems. You can also use firewalls to protect Application Servers and Web servers from unauthorized outside access.

A *demilitarized zone* (DMZ) configuration involves multiple firewalls that add layers of security between the Internet and critical data and business logic. A wide variety of topologies are appropriate for a DMZ environment. Although WebSphere Application Server provides great flexibility in configuring DMZ topologies, the basic locations of elements in a simple DMZ topology follow:



The main purpose of a DMZ configuration is to protect the business logic and data in the environment from unauthorized access. A typical DMZ configuration includes:

- An outer firewall between the public Internet and the Web server or servers processing the requests originating on the company Web site.
- An inner firewall between the Web server and the Application Servers to which it is forwarding requests. Company data also resides behind the inner firewall.

The area between the two firewalls gives the DMZ configuration its name. Additional firewalls can further safeguard access to databases holding administrative and application data.

Comparison of DMZ configurations

Somehow, requests for applications that WebSphere Application Server manages must get from the Web server to the Application Servers, passing through firewalls. You can implement DMZ configurations for a wide variety of multitiered systems.

WebSphere Application Server offers many configuration choices for accomplishing this goal. The following table summarizes benefits of each DMZ configuration option supported by the product. Criteria for each topology are described after the table.

An **X** represents an advantage.

Benefit (X) or statistic	**Remote HTTP**	**Reverse proxy**
Compatible with product security	**X**	**X**
Avoids data access from DMZ	**X**	**X**
Supports Network Address Translation (NAT)	**X**	**X**
Avoids DMZ protocol switch		**X**
Allows encrypted link between Web server and Application Server	**X**	Depends on Web server
Avoids single point of failure	**X**	
Minimum firewall holes	One per Application Server, plus one if WebSphere Application Server security is used on the Web server machine	One

- **Works with product security.** WebSphere Application Server security protects applications and their components, by enforcing authorization and authentication policies. Configuration options compatible with product security are desirable because they do not necessitate alternative security solutions.
- **Avoids critical business data in the DMZ.** A DMZ configuration protects application logic and data, by creating a buffer between the public Internet Web site and the internal intranet, where Application Servers and the data tier reside. Desirable DMZ topologies do not have databases or application servers with critical business data in the DMZ.
- **Supports Network Address Translation (NAT).** A firewall product that runs NAT receives packets for one IP address, and translates the headers of the packet to send the packet to a second IP address. In environments with firewalls employing NAT, avoid configurations involving complex protocols in which IP addresses are embedded in the body of the IP packet, such as Java Remote Method Invocation (RMI) or Internet Inter-Orb Protocol (IIOP). These IP addresses are not translated, making the packet useless.
- **Avoids the DMZ protocol switch.** The Web server sends HTTP requests to Application Servers behind firewalls. It is simplest to open an HTTP port in the firewall to let the requests through. Configurations that require switching to another protocol, such as IIOP, and opening firewall ports corresponding to the protocol, are less desirable. They are often more complex to set up, and the protocol switching overhead can impact performance.
- **Allows an encrypted link between Web server and Application Server.** Configurations that support encryption of communication between the Web server and application server reduce the risk that attackers are able to obtain secure information by *sniffing* packets sent between the Web server and Application Server. A performance penalty usually accompanies such encryption.
- **Avoids a single point of failure.** A point of failure exists when one process or machine depends on another process or machine. A single point of failure is especially undesirable because if the point fails, the whole system becomes

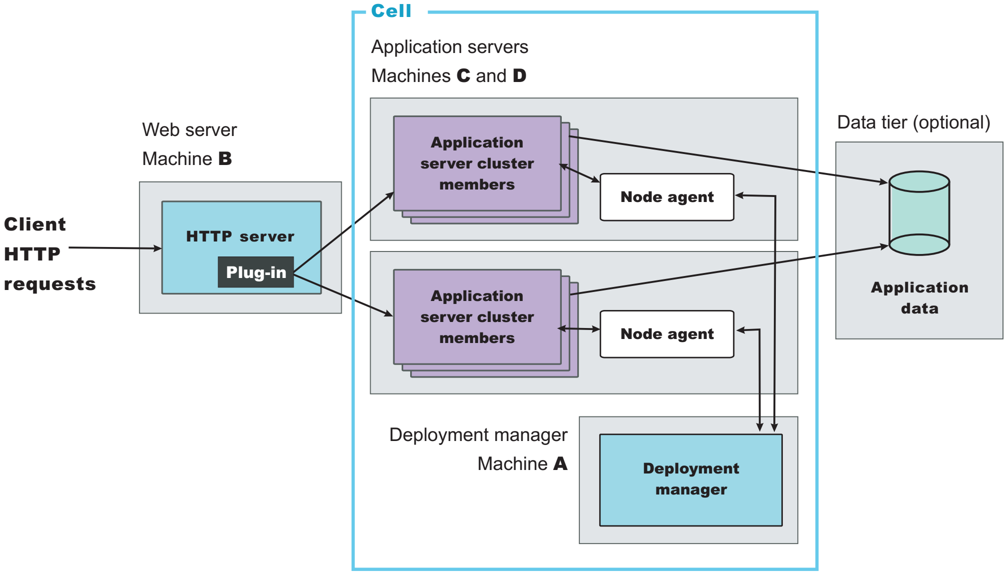
unavailable. When comparing DMZ solutions, a single point of failure refers to a single point of failure between the Web server and Application Server. Various failover configurations can minimize downtime and possibly even prevent a failure. However, these configurations usually require additional hardware and administrative resources.

- **Minimizes the number of firewall holes.** Configurations that minimize the number of firewall ports are desirable because each additional firewall port leaves the firewall more vulnerable to attackers.

Some solutions are faster than others, in terms of the number of client requests they can process per unit of time. Some solutions require little or no maintenance after you establish them, while others require periodic administrative steps, such as stopping a server and starting it again after modifying resources that affect the configuration. To learn about the necessary maintenance for a topology, review the instructions for setting up and maintaining that topology. Of course, if you can automate the necessary administrative steps through command line clients and scripting, this might not concern you.

Horizontal scaling topology

Horizontal scaling exists when there are members of an application server cluster on multiple physical machines. Having cluster members on several machines lets a single application span the machines, yet still present a single system image. The following figure shows an example of horizontal scaling.



In this example, the Web server on Machine B distributes requests to clustered Application Servers on Machines C and D. Cluster members on Machines C and D are created in the same cluster.

You can combine a network dispatcher to distribute client HTTP requests with clustering, to reap the benefits of both types of horizontal scaling. The *Network dispatcher topology* topic describes this system configuration.

Typical use

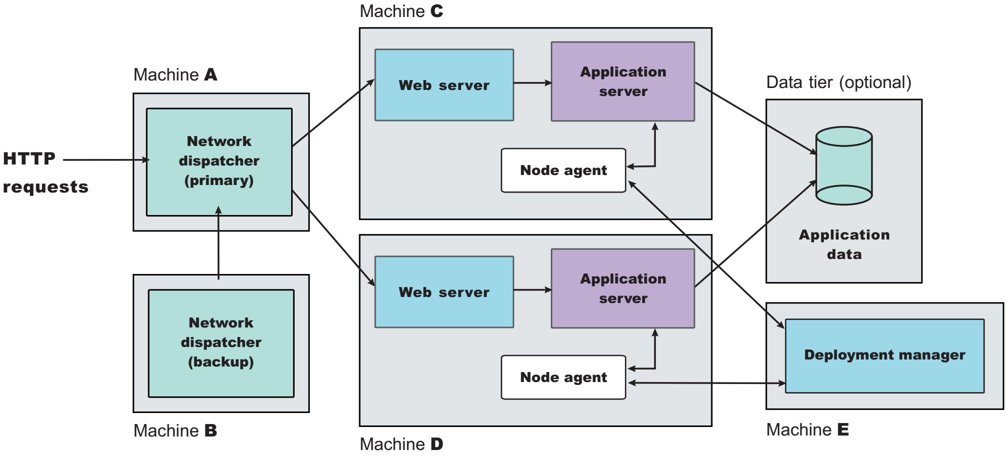
Horizontal scaling provides the increased throughput of vertical scaling topologies but also provides failover support. This topology lets you handle Application Server process failure and hardware failure without significant interruption to client service. You can also use horizontal scaling to optimize the distribution of client requests through mechanisms, such as workload management or remote HTTP transport.

Dispatcher

The Dispatcher component is part of the Load Balancer component set of the IBM WebSphere Application Server Edge Components product. The Edge components are included in WebSphere Application Server Network Deployment. The Dispatcher performs intelligent load balancing by using server availability, capability, workload, and other criteria you can define, to determine where to send a TCP/IP request. You can use the Dispatcher to distribute HTTP requests among Application Server instances that are running on multiple physical machines.

A simple Dispatcher topology

The following figure illustrates a simple horizontal scaling configuration that uses the Dispatcher to distribute requests among Application Servers on different machines.

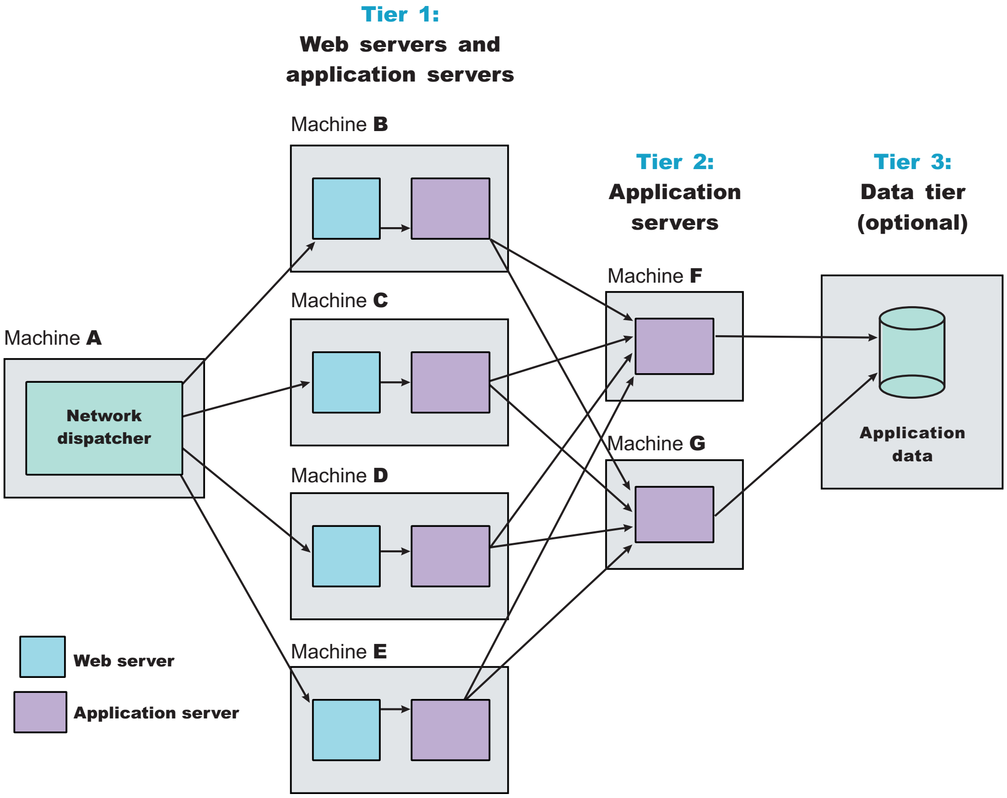


You normally configure a backup node for the Dispatcher machine, to eliminate it as a single point of failure. In this example, you can set up the backup Dispatcher node (Machine B) to take over if the primary Dispatcher node (Machine A) fails.

You can cluster Application Servers in this example from the same model, or configure them independently.

A more complex Dispatcher topology

The next figure shows a Dispatcher that distributes requests among several machines containing clustered Web servers and Application Servers. Backups are not shown.



The first tier Web server machines host servlet-based applications. The second tier Application Servers contain mostly enterprise beans that access application data and execute business logic. This approach lets you employ numerous, less powerful machines on the first tier and fewer but more powerful machines on the second tier.

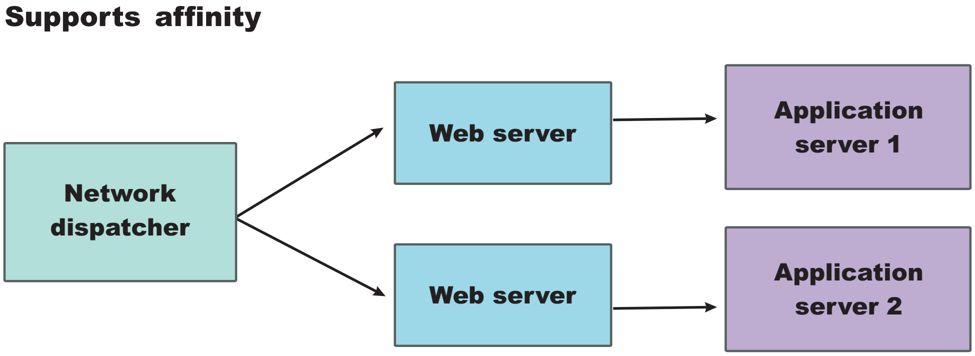
Using the Dispatcher with firewalls

You can also use a load balancing product, such as the Dispatcher with demilitarized zone (DMZ) topologies. The Dispatcher simplifies creating a DMZ topology with two firewalls. One firewall protects the Web server from the public Web site. The second protects backend systems from the Web server in the DMZ, by using proxy services.

The Dispatcher machine is placed between the outside firewall and the cluster of Web servers that it serves. The outside firewall provides filtering to allow only HTTP and HTTPS traffic. The firewall to the backend systems (DBMS, CICS, SAP, and so on) handles non-HTTP protocols, such as IIOP and JDBC. The WebSphere Application Server can reside in the DMZ or on the same side of the firewall as the data tier.

The Dispatcher and session affinity

In a Dispatcher or similarly configured topology, you must associate a Web server with a separate, rather than clustered, Application Server to preserve affinity between the servers.



Discussion

Adding a mechanism for the Dispatcher to distribute HTTP requests provides these advantages:

- Improves server performance, by distributing incoming TCP/IP requests - in this case, HTTP requests, among a cluster of servers.
- Increases the number of connected users.
- Eliminates the Web server as a single point of failure. You can also use this mechanism in combination with WebSphere Application Server workload management, to eliminate the Application Server as a single point of failure.
- Improves throughput, by letting multiple servers and CPUs handle the client workload.

Instructions

Set up machines containing Web servers and Application Servers for the topology you plan to implement.

Place the Dispatcher, or another load balancing product, in front of the Web server machines. See the Edge Component documentation for the Dispatcher, or the other load balancing product documentation. Instructions vary per product.

The load balancing product communicates with the Web server, which in turn communicates with Application Servers. The configuration involves setting up communications between the load balancing product and the Web server.

It does not matter to the Dispatcher whether the Web server is routing requests to an Application Server or processing them itself. Therefore, it is not necessary to perform any special configuration to make the load balancing product and Application Servers aware of one another. This lack of configuration is true with the Dispatcher, based on testing with IBM WebSphere Application Server. Results can vary with other load balancing products.

Web server separation

Web server separation is a topology that physically separates the Web (HTTP) server from the Application Servers, placing the Web server on a different machine in the configuration. Compared to a configuration where the Web server and the Application Servers are located on the same physical server, separating the Web server can improve application performance, provide better fault isolation, and enhance security. These topologies are often used with firewalls to create a secure demilitarized zone (DMZ) surrounding the Web server.

WebSphere Application Server provides these alternatives for physically separating the Web server from the Application Server:

- HTTP transport configurations
- Reverse proxy or IP forwarding configurations

These system topologies are described in detail in other topics.

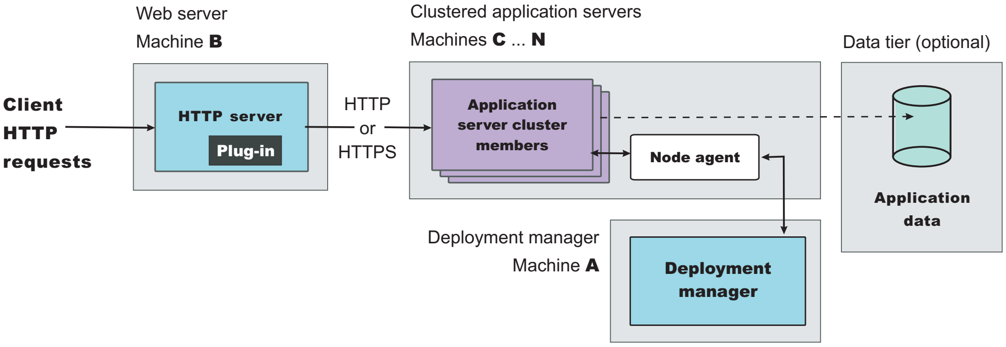
The following table summarizes advantages and disadvantages of each configuration. Criteria are explained after the table.

Topology	Secure Sockets Layer	Database password required?	Workload management	Network Address Translation	Performance	Administration
HTTP server separation	Yes	No	Yes	Yes	High	Manual
Reverse proxy	Yes	No	No	Yes	High	Manual

- **Secure Sockets Layer** Supports Secure Sockets Layer (SSL) security.
- **Database password required?** Requires a stored database user ID and password on the machine used by the database processes.
- **Workload management** Uses the workload management service to balance client workloads.
- **Network Address Translation** Supports Network Address Translation (NAT) firewalls. NAT firewalls receive packets for one IP address, translate the headers of the packets, and send the packets to a second IP address.
- **Performance** Compares the relative performance of each of these configurations.
- **Administration** Specifies whether to administer the configuration manually or through the administrative console.

These criteria give you a basis to compare the relative difficulty of administering each configuration.

HTTP transport: WebSphere Application Server can use HTTP to route requests from a Web server to Application Servers on remote machines.



In the diagram, Machine B hosts the Web server and receives HTTP requests from clients. The Web server forwards the requests to the Application Server on Machine C, by using the HTTP or HTTPS protocol.

Variations on this configuration include vertical scaling of the application servers by creating new application servers on the same machine. Alternatively, for a horizontal scaling scenario, add machines (D, E, ...N) running application servers to the environment.

The HTTP transport supports Network Address Translation (NAT) firewalls.

Load balancing support

The HTTP transport is fully integrated with WebSphere Application Server workload management and the clustering facility. HTTP transport balances loads within a cluster.

If you are using a third party load balancer in between the plug-in and the Application Server for HTTP requests that are not subject to an existing affinity, you can override the round robin algorithm that determines where to route a request. To override this algorithm, specify the IP address of your third party load balancer on the `ClusterAddress` element within the ServerCluster tag in plugin-cfg.xml file. Specifying this IP address delegates load balancing for the servers within this cluster to the third party load balancer. (For more information about load balancing in a z/OS environment, see the *Enabling High Availability e-business on IBM eServer zSeries* Redbook.)

- **Load balancing between Application Servers.** You can configure the HTTP transport to forward requests from each URL to a different application server and its cluster members, enabling manual load balancing. For instance, you can forward URLs that generate a large number of requests to application servers on more powerful machines.
- **Load balancing among cluster members.** The HTTP transport automatically distributes requests among members of a cluster that is defined to respond to a single URL. The method for selecting which cluster member handles a particular request combines a round-robin selection policy with server affinity.

  If session persistence is not enabled, which is the default, requests are distributed among all available cluster members using a strict round-robin policy. Each cluster member gets the next request in turn. The only exception is when a cluster member is added or restarted. See the failover support information later in this topic for details.

  If session affinity is enabled, requests are distributed as follows:

  - The HTTP transport distributes the first request of each session and all requests that are not associated with a session, as if session persistence is not enabled. They are distributed using a round-robin policy, except when cluster members are added or restarted.
  - The HTTP transport attempts to distribute all requests associated with a particular session to the same cluster member. Different sessions are assigned to different cluster members of the Application Server.

    Be aware that there is no guarantee that the same cluster member is used for all requests within a session. You cannot always maintain session affinity in situations where the number of available cluster members changes during the lifetime of a session. The Session Manager session clustering facility ensures that session state is not lost if requests are switched to another cluster member during a session. In any case, applications that require available session information across multiple client invocations must store session information in a database.

Failover support

The HTTP transport automatically handles failover and changes in the number of available cluster members.

- If a cluster member is stopped or unexpectedly fails, all subsequent requests are distributed among the remaining cluster members. The unavailable cluster member is skipped.
- If a cluster member is added or restarted, the system automatically begins to distribute requests to it. The next several requests are dispatched to that cluster member before HTTP resumes its normal methods for distributing requests to the cluster members of an Application Server, based on whether session persistence is enabled. See the load balancing support information for details.

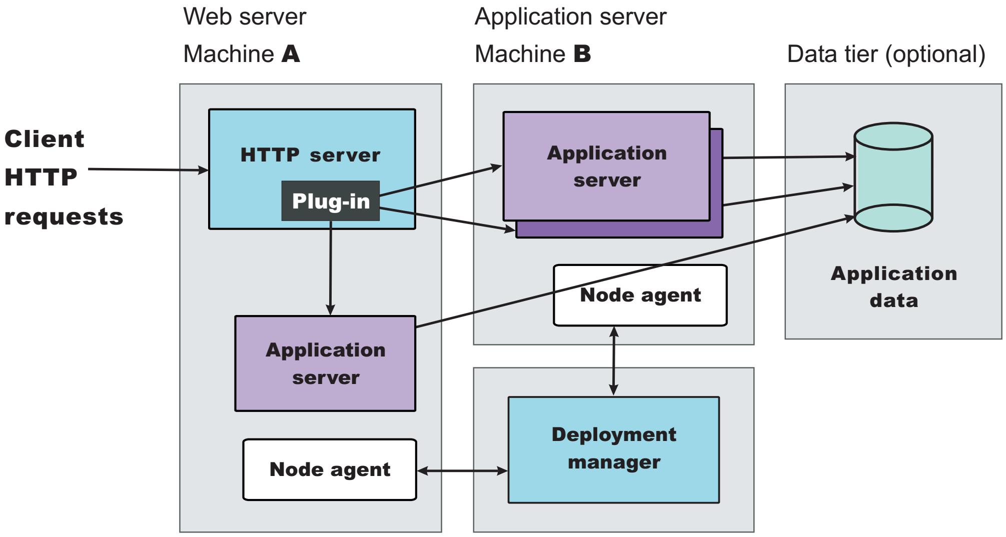
Typical use

HTTP transport has the following advantages:

- Supports load balancing and failover.
- Requires no requirements for database access through the firewall.
- Supports WebSphere Application Server security.
- Supports Secure Sockets Layer (SSL) encryption for communications between the Web server and the Application Server.
- Supports Network Address Translation (NAT) firewalls.
- Supports relatively fast performance.

The HTTP transport has the disadvantage of requiring at least one firewall port, more if multiple Application Server cluster members are configured, or WebSphere Application Server security is used on the machine hosting the Web server.

A variation of the HTTP transport topology occurs when an instance of the Application Server runs on the machine that hosts the Web server. In this configuration, an instance of an Application Server runs on the same machine as the Web server.



Such configurations can direct client requests to additional application server cluster members on other machines. This example redirects client requests to both the Application Server instance running on Machine A and the cluster members running on Machine B. You can administer all Application Server instances from the deployment manager node, which can exist on Machine A, Machine B, or another machine. The deployment manager communicates with the nodeagent server process on each machine, to coordinate configuration changes.

Typical use

This topology is recommended only in situations where hardware limitations prevent you from hosting the Web server on a dedicated machine.

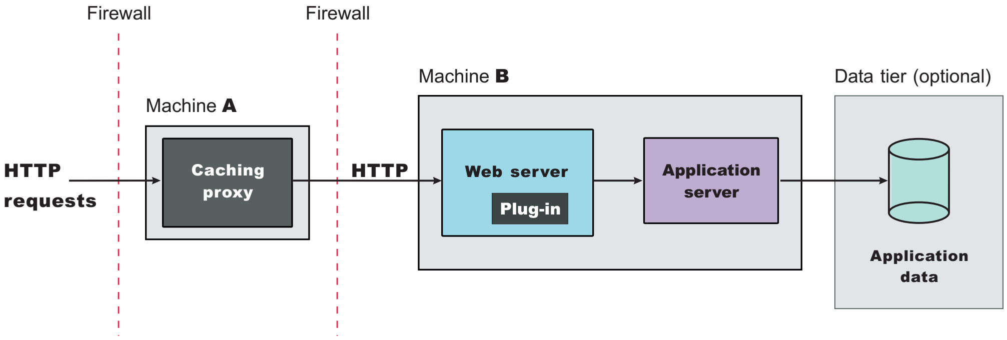
In many production environments, one set of servers is configured to run Web servers and another set of servers is configured to run Application Servers. This configuration lets you add capacity in a production environment. You might also use this topology to more fully replicate a production configuration in a test environment. This topology provides a means of load distribution between a machine hosting both the Web server and Application Server, and machines hosting just the Application Server.

You can also use this topology to distribute the workload in situations where there are a limited number of machines.

Reverse proxy (IP forwarding):

Reverse proxy, or *IP-forwarding* topologies use a reverse proxy server, such as the Caching Proxy in WebSphere Application Server Edge Components, to receive incoming HTTP requests and forward them to a Web server. The Web server forwards the requests to the Application Servers for actual processing. The reverse proxy returns completed requests to the client, hiding the originating Web server.

The following figure shows a simple reverse proxy topology.



In this example, a reverse proxy resides in a demilitarized zone (DMZ) between the outer and inner firewalls. It listens on an HTTP port, typically port 80, for HTTP requests. The reverse proxy then forwards such requests to an HTTP server that resides on the same machine as WebSphere Application Server. After the requests are fulfilled, they are returned through the reverse proxy to the client, hiding the originating Web server.

Typical use

Reverse proxy servers are typically used in DMZ configurations to provide additional security between the public Internet and the Web servers (and application servers) servicing requests. A reverse proxy product used with WebSphere Application Server must support Network Address Translation (NAT) and WebSphere Application Server security.

Reverse proxy configurations support high performance DMZ solutions that require as few open ports in the firewall as possible. The reverse proxy capabilities of the Web server inside the DMZ require as few as one open port in the second firewall, potentially two if using Secure Sockets Layer (SSL) - port 443.

Advantages of using a reverse proxy server in a DMZ configuration include:
- The reverse proxy server does not need database access through the firewall.
- The reverse proxy configuration supports WebSphere Application Server security and NAT firewalls.
- The basic reverse proxy configuration is well known and tested in the industry, resulting in less customer confusion than other DMZ configurations.
- The reverse proxy configuration is reliable and its performance is relatively fast.
- The reverse proxy configuration eliminates protocol switching, by using the HTTP protocol for all forwarded requests.
- The reverse proxy configuration does not affect the configuration and maintenance of an application deployed on WebSphere Application Server.
- The reverse proxy server uses only one HTTP firewall port for requests and responses.

The reverse proxy configuration is also a disadvantage in some environments where security policies prohibit using the same port or protocol for inbound and outbound traffic across a firewall.

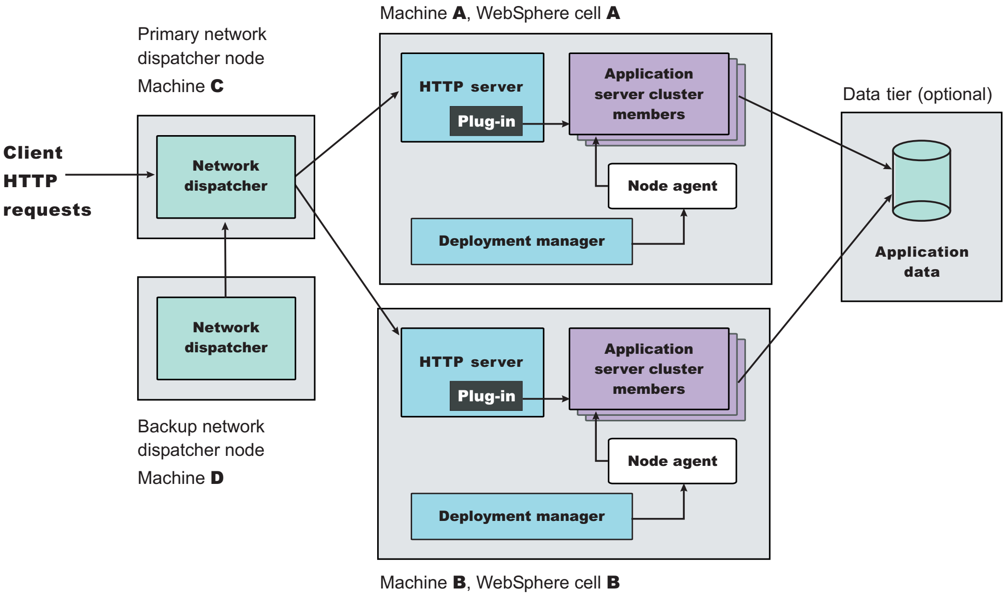
Disadvantages of using a reverse proxy server in a DMZ configuration include the following:
- The presence of a reverse proxy server in a DMZ is not suitable for some environments.
- The reverse proxy configuration requires more hardware and software than similar topologies that do not include a reverse proxy server, which makes it more complicated to configure and maintain.
- The reverse proxy server does not participate in WebSphere Application Server workload management.

Instructions

Implementation specifics are determined by the reverse proxy server. Refer to the documentation for the product you are using. No additional WebSphere Application Server administration is required for the reverse proxy server, although you might need it for other elements of the reverse proxy topology.

Multiple deployment manager cells

The following figure shows an example of how you can implement an application over multiple WebSphere Application Server deployment manager cells.



The example application runs simultaneously in two cells, each hosted on a different physical machine (Machines A and B). Network Dispatcher is used to distribute incoming HTTP requests between the two cells, presenting a single image of the application to clients. A backup Network Dispatcher node provides failover support.

Installing enterprise applications on the cluster in each deployment manager node ensures that identical versions of the application run in each cluster member. However, you administer each cell independently. Each cell has its own set of XML configuration files.

You can also run a different version of the application in each cell cluster. Because the cells are isolated from one another, you can also run different versions of the WebSphere Application Server software in each cell. For example, you can have a Version 5 cell and a Version 4 domain interoperating in your network.

Typical use

Topologies that incorporate more than one cell have the following advantages:

- Isolation of hardware failure. If one cell goes offline due to hardware problems, the others can still process client requests.
- Isolation of software failure. Running an application in two or more cells isolates any problems that occur within a cell, while the other cells continue to handle client requests. This isolation is helpful in a variety of situations:
  - When rolling out a new application or a revision of an existing application.

    You can bring the new application or revision online in one cell, and test it in a live situation while other cells continue to handle client requests with the production version of the application.
  - When deploying a new version of the WebSphere Application Server software.

    You can bring the new version into production, and test it in a live situation without interrupting service.

- – When applying fixes or patches to the WebSphere Application Server software.

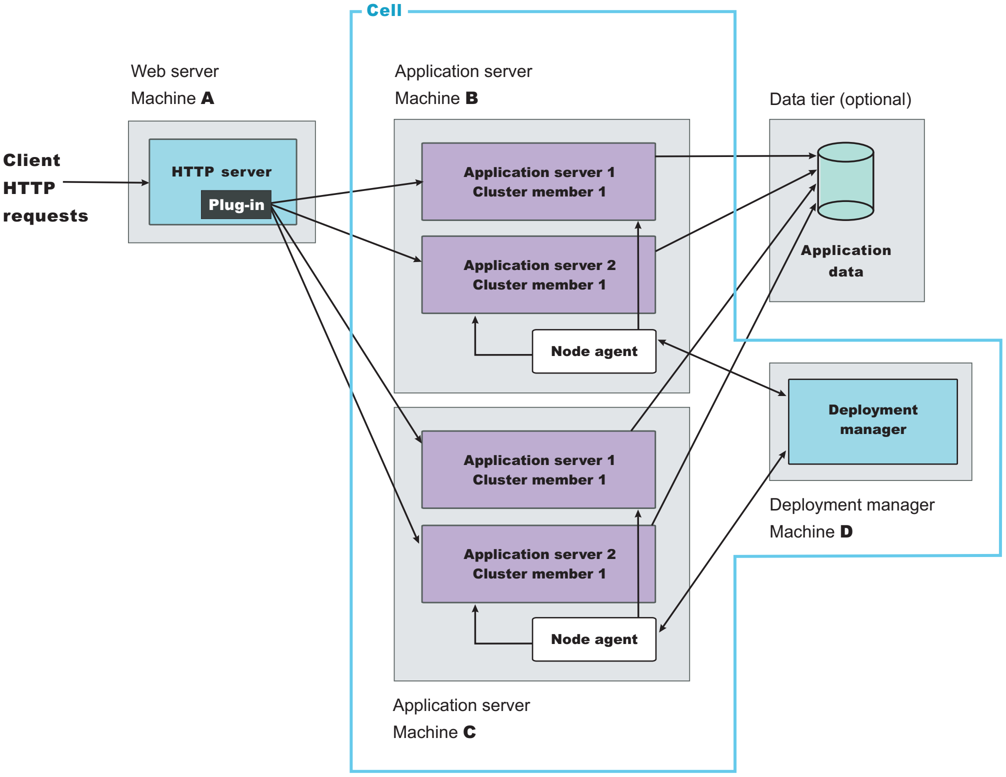
  You can take each cell offline to upgrade it, without interrupting the application.

If an unforeseen problem occurs with the new software, using multiple cells can prevent an outage to an entire site. You can also rollback to a previous software version more quickly. You can handle hardware and software upgrades on a cell-by-cell basis during off-peak hours.

Using multiple cells has several drawbacks:
- Deployment is more complicated than for a single administrative cell.
- Multiple cells require more administration effort because each cell is administered independently. Use scripts to standardize and automate common administrative tasks to reduce this problem.

Multiple Application Servers within a node

The following figure shows a topology in which cluster members of more than one Application Server are hosted on a physical node.



The example topology is a variation of the basic horizontal scaling topology. Cluster members are not hosted on just a single machine, but are distributed throughout all of the machines in the system. In this example, a cluster member is hosted on Machines B and C. Machine A serves as the Web server for the application and distributes client requests to the Application Server cluster members on each node.

Typical use

Hosting cluster members of multiple Application Servers within a node provides the following benefits:

- Improved throughput. Clustering an Application Server enables it to handle more client requests simultaneously.
- Improved performance. Hosting cluster members on multiple machines lets cluster members have full use of the processing resources on their machines.
- Hardware failover. Hosting cluster members on multiple nodes isolates hardware failures and provides failover support. Redirect client requests to the Application Server cluster members on other nodes if one node goes offline.
- Application server failover. Hosting cluster members on multiple nodes also isolates application software failures and provides failover support if a cluster member stops running. Redirect client requests to cluster members on other nodes.
- Process isolation. If one Application Server process fails, its cluster members on the other nodes are unaffected.

A drawback of this topology is more complex maintenance. You must maintain cluster members of each Application Server on multiple machines.

Running WebSphere Application Server across versions

WebSphere Application Server, Version 5 is generally interoperable with WebSphere Application Server, Versions 3.5.x and 4.0.x. However, there are specific requirements to address for each version. Make the following changes to support interoperability between versions.

1. Apply required fixes.

Table 1. Fixes to apply for Version 3.5.x

Fix	Version 3.5.3	Version 3.5.4	Version 3.5.5	Version 3.5.6
PQ51387	Apply	Apply		
PQ60074	Apply	Apply	Apply	Apply
PQ60335			Apply	
PQ63548	Apply	Apply	Apply	Apply

Table 2. Fixes to apply for Version 4.0.x

Fix	Version 4.0.1	Version 4.0.2	Version 4.0.3
PQ60074	Apply	Apply	
PQ60336	Apply	Apply	
PQ63548	Apply	Apply	Apply

All fixes are available on the IBM WebSphere Application Server Support page.

Fix PQ51387:
A naming client fix that supports Version 3.5.x naming client access to the Version 5 name server.

Fix PQ60074:
An Object Request Broker (ORB) fix that supports Version 5 naming client access to the Version 3.5.x or 4.0.x name server. A down-level client has no problem accessing a Version 5 name server, even using corbaloc.

Fix PQ60335:
An ORB fix to reconcile `java.math.BigDecimal` and other class differences in IBM Software Development Kits 122 and 131.

Note: This fix does not apply to IBM Software Development Kits on the Solaris Operating Environment.

Fix PQ60336:
An ORB fix to reconcile `java.math.BigDecimal` and other class differences in IBM Software Development Kits 130 and 131.

Note: This fix does not apply to IBM Software Development Kits on Solaris platforms.

Fix PQ63548:
A fix to correct problems that might occur when passing embedded valueTypes between WebSphere Application Server releases.

The best solution is to upgrade all your installations to the latest release and PTF levels, such as Versions 3.5.7 or 4.0.4, which do not require this fix. If this solution is not possible, apply the fix to your version.

Symptoms include `org.omg.CORBA.MARSHAL` exceptions when passing embedded valueTypes across the versions. Sometimes, other symptoms might mask `org.omg.CORBA.MARSHAL` exceptions, which makes them difficult to identify.

If symptoms reoccur in spite of the fix, re-export existing IORs.

2. Follow required guidelines.

Table 3. Guidelines to apply for Version 3.5.x and Version 4.0.x

Guideline	Version 3.5.x	Version 4.0.x
1	Apply	
2	Apply	Apply
3	Apply	
4	Apply	
5	Apply	Apply
6	Apply	Apply

Guideline 1:
Use the context of the lowest common denominator when interoperating at the naming level. For example, always use the 3.5.x context com.ibm.ejs.ns.jndi.CNInitialContextFactory when a client or server is at 3.5.x. For later versions, use the current com.ibm.websphere.naming.WsnInitialContextFactory context.

Guideline 2:
Make required naming changes to support Version 3.5.x or 4.0.x client access to Version 5 enterprise beans. This issue is new, introduced by Version 5. There are several ways to make it work, such as:

- Updating the `namebindings.xml` file if you do not use the Version 5 migration tools, but have installed Version 3.5.x or 4.0.x applications on Version 5. To allow Version 3.5.x or 4.0.x client access to the applications, add additional information to the bind information in the Version 5 namespace to make the old JNDI names work. Add the information to the `namebindings.xml` configuration file at the cell level using the administrative console.

  Note: Applications that you migrate to Version 5 during installation, or that you manually migrate using the WASPreUpgrade and WASPostUpgrade migration tools, already have this update.

After federating an application server node into a deployment manager cell, you cannot use the migration tools. To use these tools again, remove the node from the cell, use the tools, and add the node to the cell again.

- Calling Version 5 enterprise beans directly using the naming path that includes the server on which the enterprise beans are running, such as `cell/node/server1/some/ejb/name`, for example.
- Using the Version 4.0.x client `java:/comp` location to find Version 5 enterprise beans. (You cannot use the command from a Version 3.5.x client.)

Guideline 3:

Ensure that programs performing a JNDI lookup of the `UserTransaction` interface, use an InitialContext that resolves to a local implementation of the interface. Also ensure that such programs use a JNDI location appropriate for the enterprise bean version.

Prior to the EJB 1.1 Specification, the JNDI location of the UserTransaction interface was not specified. Earlier versions up to and including Version 3.5.x do not use the EJB 1.1 Specification. They bind the UserTransaction interface to a JNDI location of `jta/usertransaction`.

Version 4, and later releases, bind the UserTransaction interface at the location defined by the EJB 1.1 Specification, which is `java:comp/UserTransaction`.

Version 5 no longer provides the earlier `jta/usertransaction` binding within Web and EJB containers to applications at a J2EE specification level of 1.3 or later, to enforce use of the newer UserTransaction interface. For example, EJB 2.0 applications can use only the `java:comp/UserTransaction` location.

Guideline 4:

Be aware of limitations when calling WorkLoad Management (WLM)-enabled enterprise beans.

Some clients cannot call WLM-enabled enterprise beans on remote clusters when there is a local WLM-enabled enterprise bean of the same name. If there is a cluster local to the client with the same enterprise bean as the remote cluster that the client is trying to call, the client ends up talking to the local cluster. The following table lists supported combinations of clients calling WLM-enabled enterprise beans on remote application servers.

All clients at Version:	Server at Version:	Supported interoperability
3.5.6	5	Yes
4.02, 4.03	5	Yes
5	3.5.x	No
5	4.02, 4.03	Yes

Guideline 5:

Be aware of administrative console limitations.

You cannot use the administrative interfaces for Version 5 to administer a Version 3.5.x or 4.0.x administrative server. Likewise, you cannot use a Version 3.5.x or Version 4.0.x administrative console to administer a

Version 5 environment. If you use the administrative console on a remote machine to administer Version 3.5.x or 4.0.x WebSphere Application Server domains, migrating any of the nodes or domains to Version 5 renders the remote administration console ineffective for administering any Version 5 environment.

Guideline 6:

Use Secure Sockets Layer Version 3 (SSL v3) when interoperating with Version 3.5.x for secure connections. You cannot use Common Secure Interoperability Version 2 (CSIv2) for interoperability, because Versions 3.5.x and 4.0.x do not support CSIv2.

This information is dynamic and might be augmented by information in Technical Articles that are available on the WebSphere Developer Domain. Be sure to check the site for the latest information. You can find a link to the site in the Installation: Resources for Learning topic.

Return to Planning to install an e-business network to continue.

Installing new releases and maintenance levels of WebSphere Application Server for z/OS

This task uses IBM functions and methods to migrate WebSphere Application Server for z/OS in a sysplex from one functional level to the next, with as little disruption as possible. This task does not apply to WebSphere Application Server for z/OS running in a monoplex, or on a single system in a sysplex. Installing a new code level in a monoplex requires you to shut down WebSphere Application Server for z/OS. You have no choice but to disrupt service to clients of the monoplex system. You can also ignore this topic if disrupting service to clients is not a problem when running WebSphere Application Server for z/OS in a sysplex.

IBM functions and methods for migrating WebSphere Application Server for z/OS include the following:

- Documenting types of migration methods.
- Providing a function to offload WebSphere Application Server for z/OS configuration data for reloading later into a new or existing configuration.
- Managing WebSphere Application Server variables in a central system-management database, so that there is no confusion about where to go for authoritative configuration data.
- Supporting differing functional levels of WebSphere Application Server for z/OS within the same network or within the same z/OS sysplex while you perform an orderly migration of the WebSphere Application Server for z/OS run time from one functional level to another. The assumption is that this migration happens over a relatively short period of time, perhaps a few weeks.

You can install new functional levels of WebSphere Application Server for z/OS without disrupting service to your clients, provided you have the proper HFS structure in a sysplex and you use what is defined here as a *rolling upgrade*. The rolling upgrade method upgrades the host cluster of WebSphere Application Server for z/OS by upgrading clustered host instances one at a time. This lets you keep available service to clients while you do the upgrade. Availability of service continues because only one system is removed from the host cluster at any time, while other clustered host instances remain running.

You have the option of using either of two methods for performing the rolling upgrade. The optional methods differ in the HFS structure they use. One method uses a version-specific HFS structure. The other uses an alternate HFS structure. Read through the procedure for using each method to determine which is better for you. The main determinant is whether to use a shared HFS.

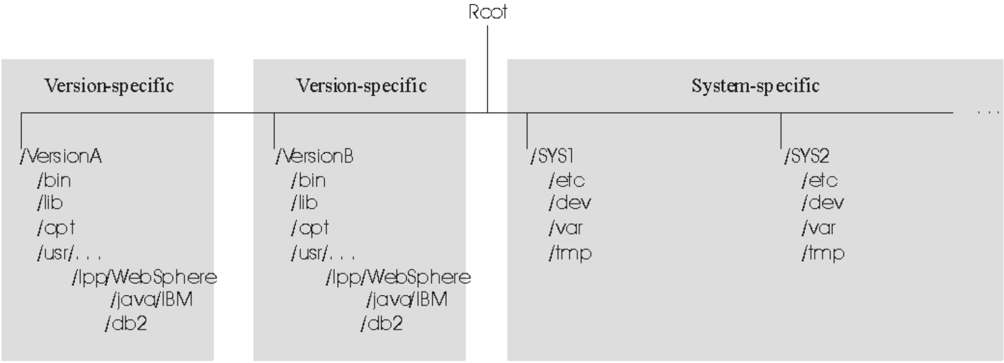
1. Optional: Use a version-specific HFS structure to upgrade WebSphere Application Server for z/OS.
2. Optional: Use an alternate HFS structure to upgrade WebSphere Application Server for z/OS.

You can maintain service to clients when upgrading the host cluster of WebSphere Application Server for z/OS.

Using a version-specific HFS structure to upgrade WebSphere Application Server for z/OS

This task describes one of the two methods for performing a *rolling upgrade* of WebSphere Application Server for z/OS in a sysplex that is introduced by Installing new releases and maintenance levels of WebSphere Application Server for z/OS. This task describes using a version-specific HFS structure to upgrade WebSphere Application Server for z/OS.

Using a version-specific HFS structure in a rolling upgrade means that two different versions of WebSphere Application Server for z/OS run in the sysplex at the same time, each with a separate HFS structure. A conventional sysplex environment has but one HFS structure that all WebSphere Application Server for z/OS systems share. To run a second, different version of WebSphere Application Server for z/OS at the same time requires a second HFS structure that is specifically for the second version as shown in the following illustration:



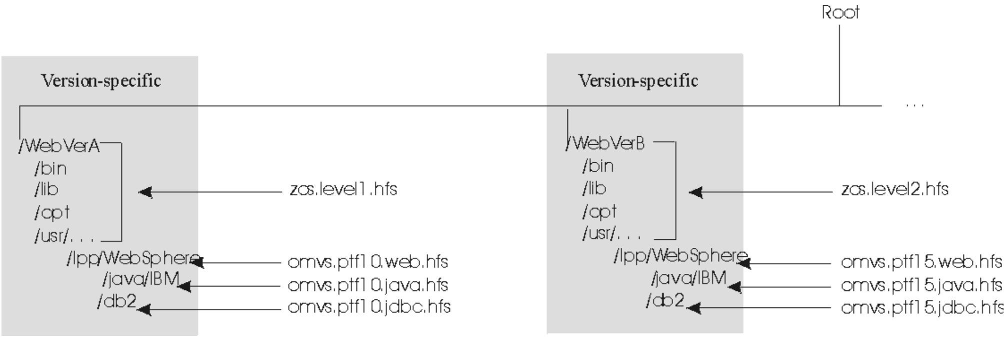
Each version-specific HFS structure requires its own /usr directory (containing a version-specific level of WebSphere Application Server for z/OS and Java) and /db2 directory (containing a version-specific level of JDBC) because of code-level interdependencies between products.

With the dual HFS structure in place, you can mount one code-level version of WebSphere Application Server for z/OS, run the host cluster from that mount point, and upgrade a second mount point where you mounted the other version of WebSphere Application Server for z/OS.

Example: Assume you mount a version-specific HFS for one service level (PTF 10) at /VersionA and another version-specific HFS for another service level (PTF 15) at /VersionB:

```
mount omvs.ptf10.was.hfs  at /VersionA/usr/lpp/WebSphere
mount omvs.ptf10.java.hfs at /VersionA/usr/lpp/java/IBM
mount omvs.ptf15.was.hfs  at /VersionB/usr/lpp/WebSphere
mount omvs.ptf15.java.hfs at /VersionB/usr/lpp/java/IBM
mount omvs.ptf15.jdbc.hfs at /VersionB/usr/lpp/db2
```

The HFS structure appears in the following illustration:



You can use a symbolic link to determine which code version any system in the sysplex addresses:

```
/usr --> $VERSION/usr
```

You use the **SETOMVS** command to control how $VERSION resolves. In this example, set the initial value of $VERSION for each system in the sysplex to `VersionA`. All systems that use the symbolic link to refer to the `/usr` directory resolve to `/VersionA/usr`.

For any system in the sysplex to use the HFS structure associated with PTF15, change the value of $VERSION on that system (and only on that system) to `VersionB`. Accordingly, any references on that system to `/usr` resolve to `/VersionB/usr` through the symbolic link.

Use the following procedure to switch the code level for a clustered host instance:

1. Install the new code, copy it to a new data set, and mount the data set at the VersionB mount point.
2. Shut down all application servers and WebSphere Application Server for z/OS on that clustered host instance
3. Use the **SETOMVS** command to change $VERSION to VersionB
4. Using SET PROG, load the LPA modules from data sets associated with the new level
5. Change the start procedures to address the new code level load libraries
6. Restart WebSphere Application Server for z/OS and the Application Servers.
7. Repeat the prior steps of this process for each clustered host instance, one at a time, to upgrade the code level of WebSphere Application Server for z/OS throughout the sysplex without disrupting service to your clients.

Each code level of WebSphere Application Server for z/OS is designed to tolerate an older code level. Differing levels of WebSphere Application Server for z/OS can coexist compatibly within the sysplex during the upgrade process.

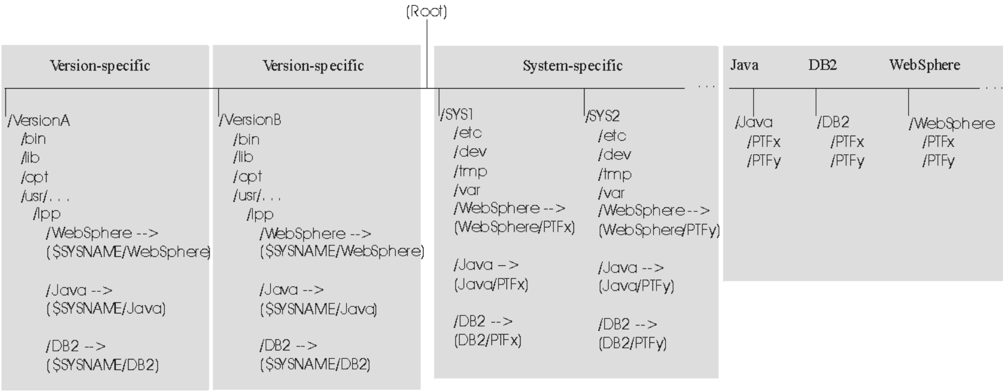
In cases when a new level of WebSphere Application Server for z/OS introduces new functions, all members of the host cluster run in compatibility mode during the upgrade process. When all clustered host instances are at the new code level, restart each instance, one by one, to enable the new function.

Read about using an alternate HFS structure to upgrade WebSphere Application Server for z/OS.

Using an alternate HFS structure to upgrade WebSphere Application Server for z/OS

This task is the second of the two methods for performing a *rolling upgrade* of WebSphere Application Server for z/OS in a sysplex that is introduced by Installing new releases and maintenance levels of WebSphere Application Server for z/OS. Using a version-specific HFS structure to upgrade WebSphere Application Server for z/OS describes the first method. This task describes using an alternate HFS structure to upgrade WebSphere Application Server for z/OS.

The alternate HFS structure accomplishes the same objective as the version-specific HFS structure, but does not mount a product HFS directly off the version-specific subdirectories (referred to by the $VERSION symbolic link). Rather, version-specific subdirectories refer to system-specific subdirectories using symbolic links with the $SYSNAME symbol. In turn, system-specific subdirectories refer to program product subdirectories through symbolic links. The alternate HFS structure is depicted in the following illustration:



The alternate HFS structure has the following characteristics:

- **Version-specific subdirectories that allow systems in the sysplex to refer to differing versions of system code.**

  The WebSphere Application Server for z/OS, Java, and DB2 for OS/390 (JDBC) subdirectories do not contain product code but instead, contain symbolic links to system-specific subdirectories through the use of the $SYSNAME symbol. As far as WebSphere Application Server for z/OS is concerned, you need not change these symbolic links. However, you should plan for creating version-specific structures for future system upgrades.
- **System-specific subdirectories that contain symbolic links to WebSphere Application Server for z/OS, Java, and DB2 for OS/390 (JDBC) subdirectories in the sysplex root.**

  The symbolic links point to specific code levels (for example, `WebSphere/PTFx`). To change the code level that a system uses, you change these symbolic links.
- **Individual subdirectories for WebSphere Application Server for z/OS, Java, and DB2 for OS/390 (JDBC) components.**

Each of these subdirectories can have one or more subdirectories for a specific code level.
- **Shared subdirectories, such as the WebSphere390 subdirectory.**

With the alternate HFS structure in place, you can mount one or more code levels of WebSphere Application Server for z/OS, Java, or DB2 for OS/390 (JDBC) under their individual component subdirectories. Each system-specific subdirectory uses symbolic links to component code levels and can refer to new code levels by changing those symbolic links.

There are certain advantages to the alternate HFS structure:
- You can stage product upgrades and service in a sysplex environment with minimal impact to availability. You can stage product upgrades or service without applying the upgrade or service to all products at the same time.
- You can upgrade one product without duplicating another version-specific ($VERSION) structure by placing the level of control at the system-specific subdirectories and linking to those subdirectories through the $SYSNAME symbol. However, it is beneficial to plan for a second version-specific structure so you are prepared for future system upgrades.
- You can help performance by using version-specific subdirectories that are read-only. Changes are in the system-specific ($SYSNAME) subdirectory, which is read/write.
- You can save storage space because you need not duplicate version-specific HFS structures for program product upgrades.

Example: Assume you have an individual component directory for WebSphere Application Server for z/OS (/WebSphere), Java (/Java), and DB2 for OS/390 (JDBC) (/DB2). Assume that each directory contains two subdirectories, one for PTFx (/PTFx) and one for PTFy (PTFy). Also assume that the code for each component update is in its own HFS data set, such as OMVS.PTFX.WEB.HFS, OMVS.PTFX.JAVA.HFS, and OMVS.PTFX.JDBC.HFS, for example. The mount commands are:

```
MOUNT FILESYSTEM('OMVS.PTFX.WEB.HFS')   MOUNTPOINT('/WebSphere/PTFx') TYPE(HFS) MODE(RDWR)
MOUNT FILESYSTEM('OMVS.PTFX.JAVA.HFS')  MOUNTPOINT('/Java/PTFx')      TYPE(HFS) MODE(RDWR)
MOUNT FILESYSTEM('OMVS.PTFX.JDBC.HFS')  MOUNTPOINT('/DB2/PTFx')       TYPE(HFS) MODE(RDWR)

MOUNT FILESYSTEM('OMVS.PTFY.WEB.HFS')   MOUNTPOINT('/WebSphere/PTFy') TYPE(HFS) MODE(RDWR)
MOUNT FILESYSTEM('OMVS.PTFY.JAVA.HFS')  MOUNTPOINT('/Java/PTFy')      TYPE(HFS) MODE(RDWR)
MOUNT FILESYSTEM('OMVS.PTFY.JDBC.HFS')  MOUNTPOINT('/DB2/PTFy')       TYPE(HFS) MODE(RDWR)
```

System SYS1 refers to the PTFx levels of code through these symbolic links:

```
/WebSphere --> /WebSphere/PTFx
/Java      --> /Java/PTFx
/DB2       --> /DB2/PTFx
```

If you want system SYS1 in the sysplex to use the HFS structure associated with PTFy, change the symbolic links for /WebSphere, /Java, and /DB2:

```
/WebSphere --> /WebSphere/PTFy
/Java      --> /Java/PTFy
/DB2       --> /DB2/PTFy
```

Use the following procedure to switch the code level for the WebSphere Application Server for z/OS for the clustered host instance on SYS1.

1. Install the new code for WebSphere Application Server for z/OS, Java, and DB2 for OS/390 (JDBC), copy each component to its own data set, and mount the data under its component subdirectory. WebSphere Application Server for

z/OS, Java, and DB2 for OS/390 (JDBC) code levels are usually interdependent, so keep the level of each component coordinated with the others.

2. Shut down all Application Servers and the WebSphere Application Server for z/OS clustered host instance on SYS1.
3. Change the symbolic links for the system-specific subdirectories for SYS1.
4. Load new run-time modules into LPA and update the link list. You can do this dynamically, but IBM recommends you re-IPL the system.
5. Change the start procedures to address the new code level load libraries.
6. Restart WebSphere Application Server for z/OS and the Application Servers.

By repeating this process for each clustered host instance, one at a time, you can upgrade the code level of WebSphere Application Server for z/OS throughout the sysplex without disrupting service to your clients.

Installing and customizing the product

This topic describes the context of installing and customizing IBM WebSphere Application Server for z/OS V5.

To create a complete, customized WebSphere for z/OS installation, you need to prepare the operating system and subsystems, configure native products (for example, WLM, RACF and TCP/IP), physically load the product code through SMP/E, run the Customization Dialog, follow the customized instructions and run the jobs (including the Installation Verification Test) from the dialog, and bring up your server.

1. Ensure the base environment is prepared to support a WebSphere for z/OS code install. This means you need to install prerequisite products and configure z/OS subsystems, such as resource recovery services (RRS) and workload management.
2. Configure your native products, such as WLM, RACF and TCP/IP.
3. Install the WebSphere for z/OS code using the SMP/E install process. The SMP/E install process is documented, including prerequisites, in detail in the WebSphere Application Server for z/OS V5 Program Directory (if you use CBPDO) or *ServerPac: Installing Your Order* (if you use ServerPac).
4. Run the WebSphere for z/OS Customization Dialog. The Customization Dialog is an ISPF Dialog that eliminates the need to hand-tailor sample jobs supplied with the product. Through a series of panels, you choose options and define variables. Using your values, the Dialog tailors the WebSphere for z/OS customization jobs but does not execute them. Rather, the dialog provides a custom set of instructions for you to follow and tailored jobs ready to complete the product customization. The Customization Dialog supports these goals:
   - Configuring the base Application Server.
   - Configuring the JMS integral provider.
   - Configuring the deployment manager.
   - Federating a base Application Server node.
   - Configuring Web service function.
5. Follow the instructions created by the Customization Dialog Make sure you run the Installation Verification Test.
6. With the Administrative Console, create the server definition. This server is used by the Installation Verification Test and is an example of an application server you will create for your own applications.

Note: When creating additional servers, the Administrative Console creates new server HFS directories, XML files, and a was.env file with a new symbolic link that points to it. You need to manually create the new server's RACF STARTED profile (though, if you choose to reuse the JCL, you will need a new STARTED profile for only the servant region and not the controller region), the RACF CBIND profile, JCL (unless you use the existing JCL), and a separate WLM application environment for the new cluster transition group (even if the additional server is under the same cell and node as another).

When you finish the entire installation and customization process, you have WebSphere for z/OS V5 running in a system. As you gain experience, you can roll out WebSphere for z/OS across your sysplex to gain the advantages of z/OS sysplex operations.

Overview of WebSphere for z/OS configuration

WebSphere Application Server for z/OS V5, hereafter referred to as WebSphere for z/OS, builds upon the function of WebSphere Application Server V4.0.1 for z/OS and OS/390 and WebSphere Application Server Advanced Edition.

This section covers planning, installing, and customizing tasks for WebSphere for z/OS.

Planning for, installing, and customizing WebSphere for z/OS includes those tasks you must perform prior to installing business applications, such as planning your system configuration and installing the WebSphere for z/OS run-time environment. This chapter:

- Gives a general overview of the tasks you must do to initially install and customize WebSphere for z/OS.
- Provides pictures and descriptions of your run-time environment after the initial installation and customization of the base and network deployment cells. The initial base installation and customization is performed on a monoplex or a single system in a sysplex.
- Provides a checklist of items you should consider for your initial installation of WebSphere for z/OS, your application development and client systems, and advanced system configurations.

To install the run-time environment initially, you must perform tasks in two general areas:

1. The base z/OS system. You must prepare various z/OS elements, products, and your network prior to setting up WebSphere for z/OS. For instance, you must perform such tasks as updating your TCP/IP network, setting up security controls, and defining workload management (WLM) workloads. See Preparing the base z/OS environment for details.
2. The WebSphere for z/OS run-time environment itself. This includes loading the code, changing parmlib members, creating environment files, and running configuration jobs. See Overview of installing and customizing WebSphere for z/OS for details.

If you already have a prior version of WebSphere for z/OS installed and customized, you can configure WebSphere Application Server for z/OS V5 to coexist with it. For more information, see the migration information.

After installation and customization, you can install application development environments for your application developers and client environments for your business applications. For more information about this, see the assembling applications information in the z/OS view of the WebSphere Application Server InfoCenter, which you can access via the WebSphere for z/OS library Web site.

When you have stabilized WebSphere for z/OS on the first system, you can enable WebSphere for z/OS in a sysplex. You may also implement other advanced system configurations, such as multi-node cells or connecting your business applications to an IMS or CICS database. These and other topics are in Performing advanced tasks.

Steps for creating your implementation plan

To get started, plan to build all WebSphere Application Server for z/OS run-time servers on one system, then replicate them on other systems as you expand into a cell. This procedure guides you through initial planning and implementation of WebSphere Application Server for z/OS on a monoplex. Then it guides you through setting up your application development and client environments. Finally, the procedure guides you through planning for optional advanced system configurations.

Before you begin: We assume you have a z/OS system on which you will implement WebSphere Application Server for z/OS.

Perform the following steps to implement your plan:

1. Plan WebSphere Application Server for z/OS on a monoplex or a single system in a multi-system sysplex. Check off each item as you complete it:

Check off	Item	For more information, see . . .
	Determine the skills you need.	Determining your skill needs
	Determine WebSphere Application Server for z/OS system requirements.	Determining WebSphere Application Server for z/OS system requirements
	Understand and plan for customization changes you will need to do for your TCP/IP network.	Updating your TCP/IP network
	Understand security options and prepare for securing your system.	Setting up security
	Set up workload management environments for WebSphere Application Server for z/OS run-time servers.	Setting up workload management (WLM)
	Customize resource recovery services for use by WebSphere Application Server for z/OS.	Recommendations for resource recovery services
	Plan for your performance and monitoring systems.	Guideline for RMF and other monitoring systems
	Follow recommendations for memory utilization.	Recommendations for using memory
	Plan and define your problem diagnosis procedures.	Planning for problem diagnosis

Check off	Item	For more information, see . . .
	Consider automatic restart management before you install WebSphere Application Server for z/OS.	Tip on automatic restart management (ARM)

2. Install and customize WebSphere Application Server for z/OS.

Check off	Item	For more information, see . . .
	Install and customize a running WebSphere Application Server for z/OS V5 base configuration.	Installing the code through SMP/E and Using the Customization Dialog

3. Perform various post-installation tasks.

Check off	Item	For more information, see . . .
	Plan and define your system backup procedures.	Guidelines for backup of the WebSphere Application Server for z/OS system
	Plan and define your software service procedures.	Overview of product service
	Implement automation controls and set up automatic restart management for WebSphere Application Server for z/OS, if desired.	Setting up automation and automatic restart management

4. Plan for your application development and client environments.

Check off	Item	For more information, see . . .
	Review WebSphere Application Server for z/OS requirements for application development and client environments.	.dita

5. (Optional) Plan and implement advanced system configurations.

Check off	Item	For more information, see . . .
	Implement the Integrated JMS Provider, using WebSphere MQSeries.	.dita
	Configure a WebSphere Application Server for z/OS network deployment cell.	Using the Customization Dialog

Check off	Item	For more information, see . . .
	Expand the WebSphere Application Server for z/OS network deployment cell to include additional systems in the same sysplex.	Setting up WebSphere Application Server for z/OS on multiple systems in a sysplex
	Implement additional application servers in the cell.	Using the Customization Dialog
	Plan to have multiple TCP/IP stacks, use connection optimization, use an IBM Network Dispatcher, or use bind-specific support.	.dita
	Implement advanced security controls such as SSL and Kerberos.	.dita
	Set up RACF protection for DB2, if desired. Tune system performance.	.dita .dita
	Access DB2 resources (if applicable)	.dita
	Access IMS resources: 1. Use the IMS Connector for Java. 2. Use the IMS JDBC Connector.	.dita
	Access CICS resources with the CICS Transaction Gateway ECI connector.	.dita
	Plan for testing and production systems.	Configuring your systems for test and production

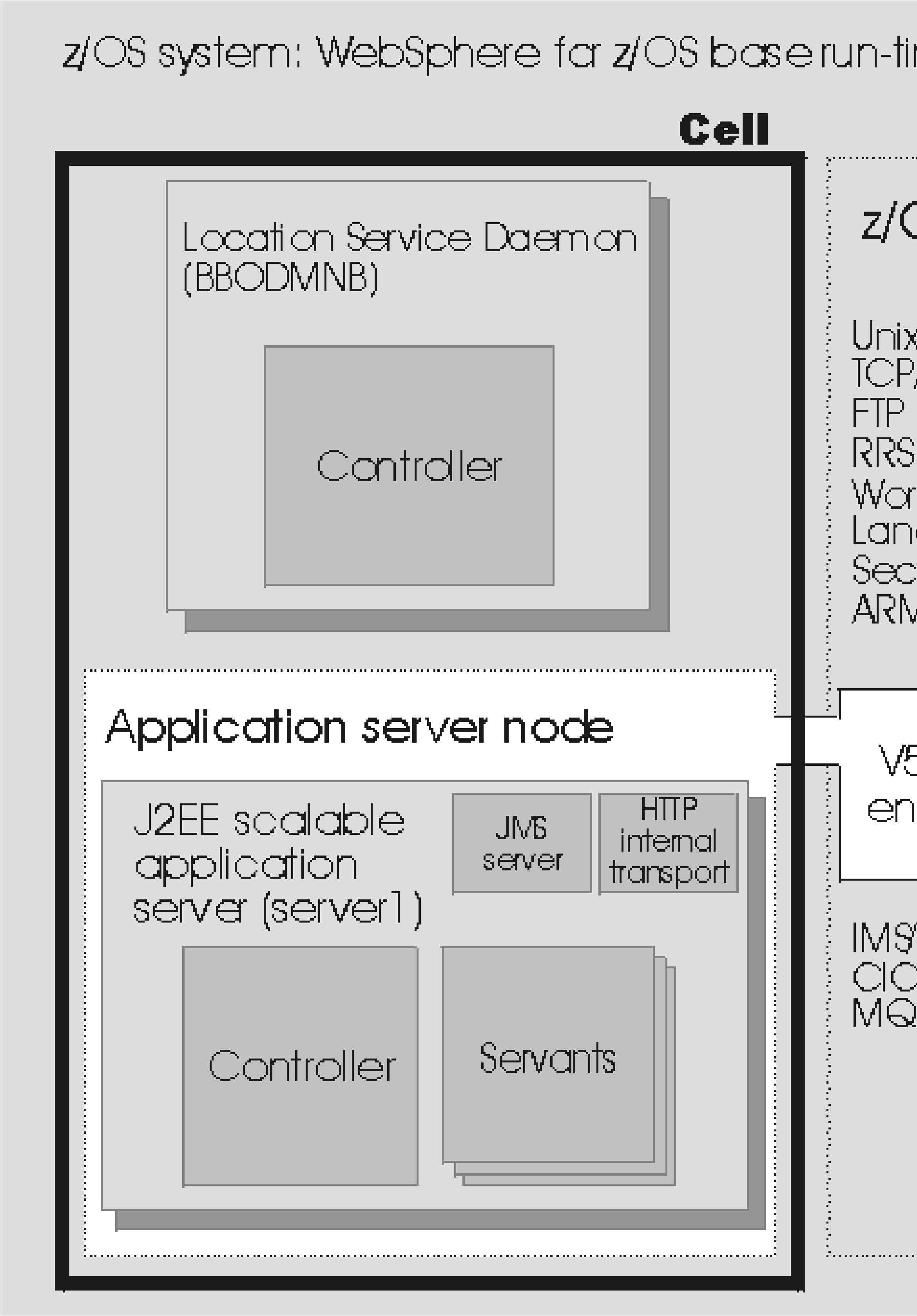
6. Plan and implement release and maintenance upgrades.

Check off	Item	For more information, see . . .
	Review code upgrade methods.	Installing new releases and maintenance levels of WebSphere Application Server for z/OS

You are done when you have checked all the applicable items.

WebSphere for z/OS base cell

A typical WebSphere for z/OS base run-time 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.


z/OS system: WebSphere for z/OS base run-ti
Cell
Location Service Daemon (BBODMNB)
Controller
Application server node
J2EE scalable application server (server1)
JMS server
HTTP internal transport
Controller
Servants
z/C
Unix
TCP/
FTP
RRS
Wor
Lan
Sec
ARM
V5
en
IMS
CIC
MQ

The run-time servers use other z/OS functions, such as z/OS UNIX and TCP/IP. Part of installing WebSphere for z/OS includes configuring these functions for use by the run-time.

J2EE servers contain at least one Web container and one EJB container. The Web container manages Web applications (servlets and JavaServer Pages), while the EJB container manages enterprise beans.

The HTTP internal transport, which is part of the J2EE Application Server, is a functional component that acts as an HTTP protocol catcher for Web applications. The HTTP internal transport is depicted in the server1 J2EE Application Server.

The JMS server, which is also part of the J2EE Application Server, hosts the WebSphere for z/OS JMS function. The JMS server is depicted in the server1 J2EE Application Server.

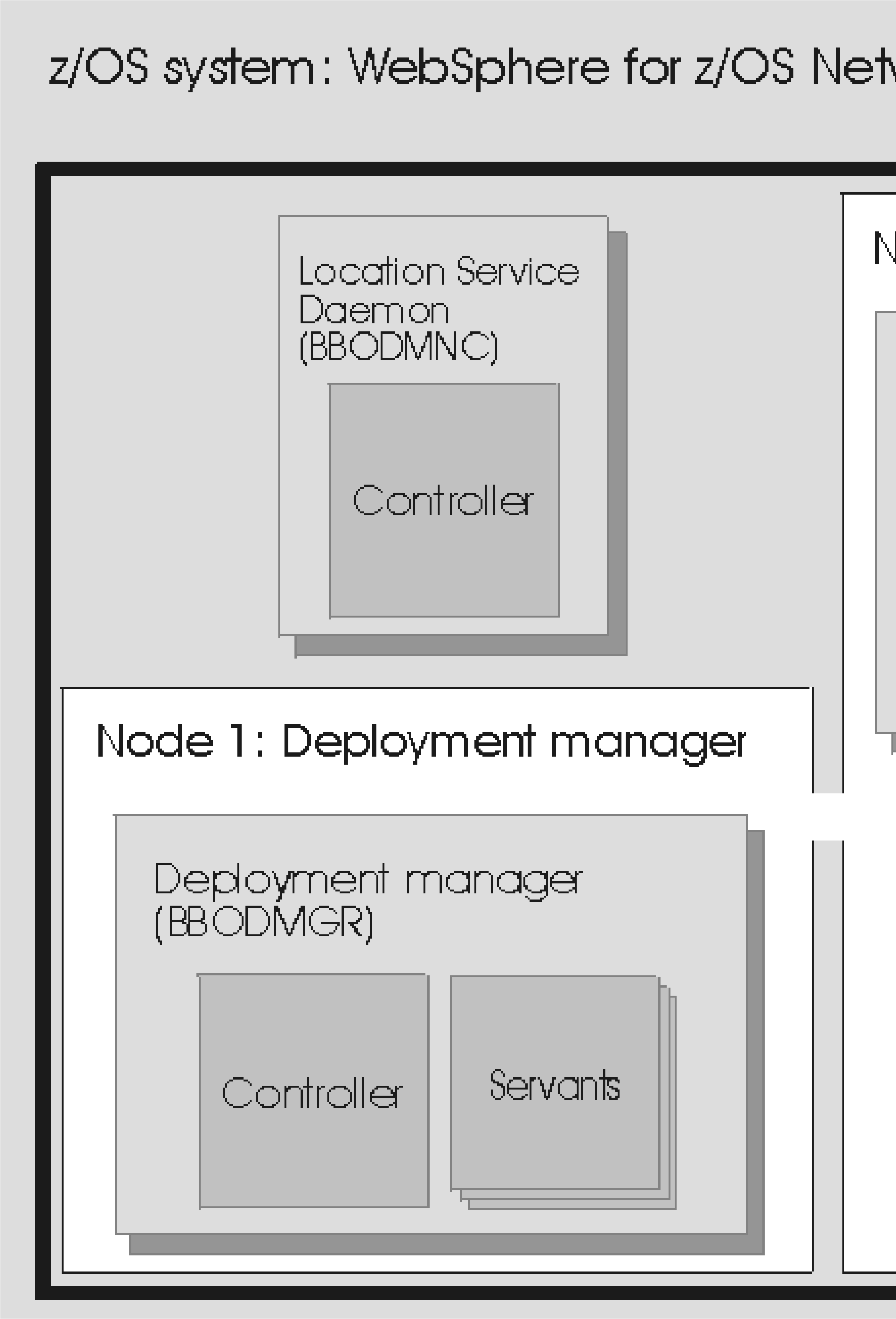
WebSphere for z/OS network deployment cell

Once you have your WebSphere for z/OS base cell up and running, there are many reasons why you should continue on and configure a network deployment cell. The network deployment cell provides a more robust operating environment. A network deployment cell has the unique ability to:

- Span multiple systems in a sysplex
- Fully utilize clustering
- Start and stop servers from the administrative console

Reference the following table for specifics on the differences between a WebSphere for z/OS base cell and a network deployment cell.

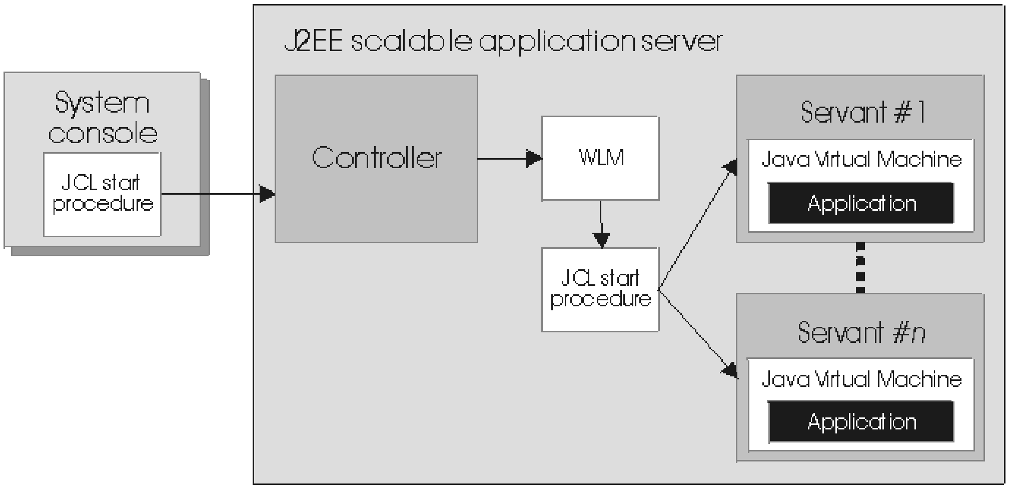
	Base cell	**Network deployment cell**
Configuration:	Set up each base server node through the ISPF Customization Dialog. Set up additional servers within the node with the Administrative Console.	Set up each deployment manager node through the ISPF Customization Dialog. Add base server nodes to the deployment manager cell through the ISPF Customization Dialog.
Address spaces:	Min: 3 (location service daemon, controller, servant)	Min: 6 (location service daemon, Application Server controller, Application Server servant, Deployment Manager controller, Deployment Manager servant, node agent)
	Max: Limited only by resources.	Max: Limited only by resources.
Administrative isolation:	Each base server node is a separate administrative domain.	All nodes in the cell are in the same administrative domain.
Operational isolation:	You may start and stop servers independently. Each server has an independent, unshared JNDI namespace.	You may start and stop servers independently. The JNDI namespace is shared among all servers in the cell.
Application servers allowed to have multiple servants?	Only if the Administration application is not present	Yes
Clustering available?	No	Yes


z/OS system: WebSphere for z/OS Netw
Location Service Daemon (BBODMNC)
Controller
Node 1: Deployment manager
Deployment manager (BBODMGR)
Controller
Servants
N

The picture above depicts a typical WebSphere for z/OS network depolyment run-time includes a location service daemon (BBODMNC and two nodes (both housed in a cell, which can span systems)-one for the Deployment Manager and one for the Application Server. The Deployment Manager node includes a Deployment Manager (BBODMGR) with a controller and any number of servants. The Application Server node includes a node agent (BBON001), a JMS server (BBOJ001), and an Application Server (server1) with a controller and any number of servants.

WebSphere Application Server for z/OS terminology and system configuration breakdown

In WebSphere Application Server for z/OS, the functional component on which applications run is called a *server*. Servers comprise address spaces that actually run code.



Within each server are two kinds of address spaces: controllers and servants. A *controller* runs system authorized programs and manages tasks, such as communication, for the server. Each server has one controller that you start with a JCL start procedure when you enter the appropriate start command on the MVS console.

Note: When a Deployment Manager cell is configured, you can also start controllers through the Administrative Console or with one of the following shell scripts:

- startServer.sh starts the Application Server
- startNode.sh starts the node agent
- startManager.sh starts the Deployment Manager server

Only an administrator with SAF OPERCMDS class authorization can issue these shell scripts.

A *servant* is the address space in which the JVM resides. It runs unauthorized programs such as business applications. Depending on the workload, a server has one or more servants running at a time. When work builds up, WLM dynamically starts additional servants to meet the demand.

Note: The location service daemon, node agent and JMS Server are specialized servers and have no servants.

Here is a quick breakdown of the different server types on your system:

Unmanaged (base) application server
The application server set up during base configuration that hosts your J2EE applications.

Managed (network deployment) application server
The application server set up during network deployment configuration that hosts your J2EE applications.

Location service daemon
A server which is the initial point of contact for client requests in either configuration.

JMS server
Hosts the WebSphere Application Server for z/OS JMS function, which controls the MQ broker and queue manager in either configuration.

Deployment Manager
A specialized Application Server that hosts the Administrative Console application (it hosts only administrative applications) and provides cell-level administrative function in a network deployment configuration. The Administrative Console application administers servers (grouped into nodes) on many different systems. The Deployment Manager is the sole occupant of its own node structure which doesn't need a node agent because there are no application servers in the node, and a cell may have only one Deployment Manager.

Note: The version of the Administrative Console application that runs in the Deployment Manager is designed to manage multi-node environments, whereas the version in the base Application Server is for single node environments only.

Node agent
Provides node-level administrative function in a network deployment configuration.

Note: Every element of the configuration (servers, clusters, nodes and cells) has both a long and short name:

- The ″Server name″ is the server long name used in the HFS path and the principal name by which the server is known to WebSphere Application Server for z/OS. It is used to identify the server through the Administrative Console and scripting. It is a mixed case name and greater than 8 characters in length.
- The ″Server short name″ is the platform-specific native alias and the principal name by which the server is known to z/OS. It is used to identify the server to underlying z/OS facilities, such as the Security Server, JES, WLM and ARM. For example, the server short name is used as the MVS JOBNAME.
- The ″Cluster short name″ is used as the WLM application environment name.

A *cluster* is a *logical grouping* of like-configured servers. Clusters exist to promote scalability and availability; workload balancing occurs across the servers in a cluster. Clusters allow you to partition workloads into separate servers while still referring to them as a single unit. Clustering is typically applied to a multi-node cell, where each node is configured on a separate system and the cluster has a member (server) on each node. Client requests are distributed among the cluster members based on workload manager decisions.

A node contains servers which may be part of a cluster. The cluster may span nodes as long as all involved nodes are in the same cell.

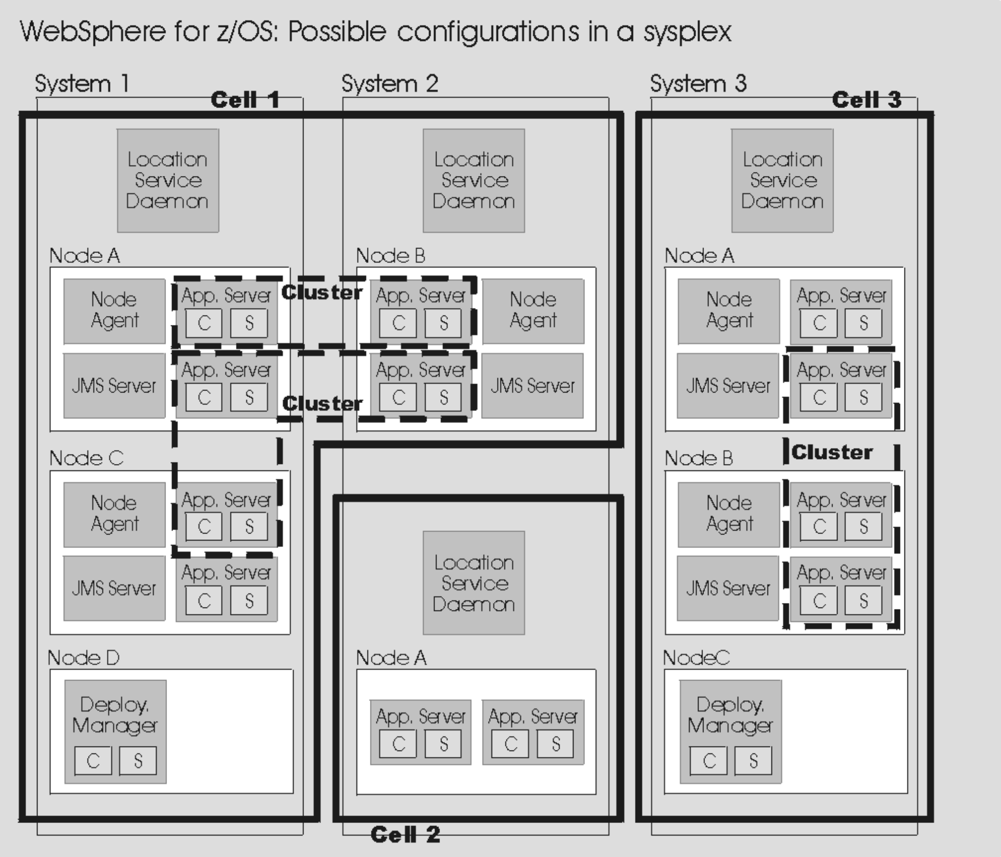
Here is a quick breakdown of clusters, nodes and cells:

cluster
A logical collection of like-configured servers. A cluster can span nodes and systems within the same cell.

node A logical collection of managed servers on a particular system in the cell. A node can contain servers that are part of clusters that span other nodes, but the node itself is confined to a single system and a single cell. You may configure multiple nodes on the same system--either in the same cell or in different cells. Nodes exist mainly for the purposes of administration, and each application server node managed by a Deployment Manager must have a node agent.

cell A logical collection of nodes and the boundary of the administrative domain. The nodes that comprise a cell must be configured on systems in the same sysplex. You may configure multiple cells in the same sysplex. Different cells may have nodes on the same systems, though a given node can be a member of only one cell. The cell is the largest unit of organization.

To help you understand the interaction between servers, clusters, nodes and cells, here is a diagram depicting various configurations you can set up in your network deployment sysplex:



Cells 1 and 3 in the illustration depict Network Deployment configuration cells. Cell 2 is a base configuration cell.

See Example of the controller JCL for an example of the structure of the JCL for the servant procedure.

The WebSphere for z/OS HTTP internal transport

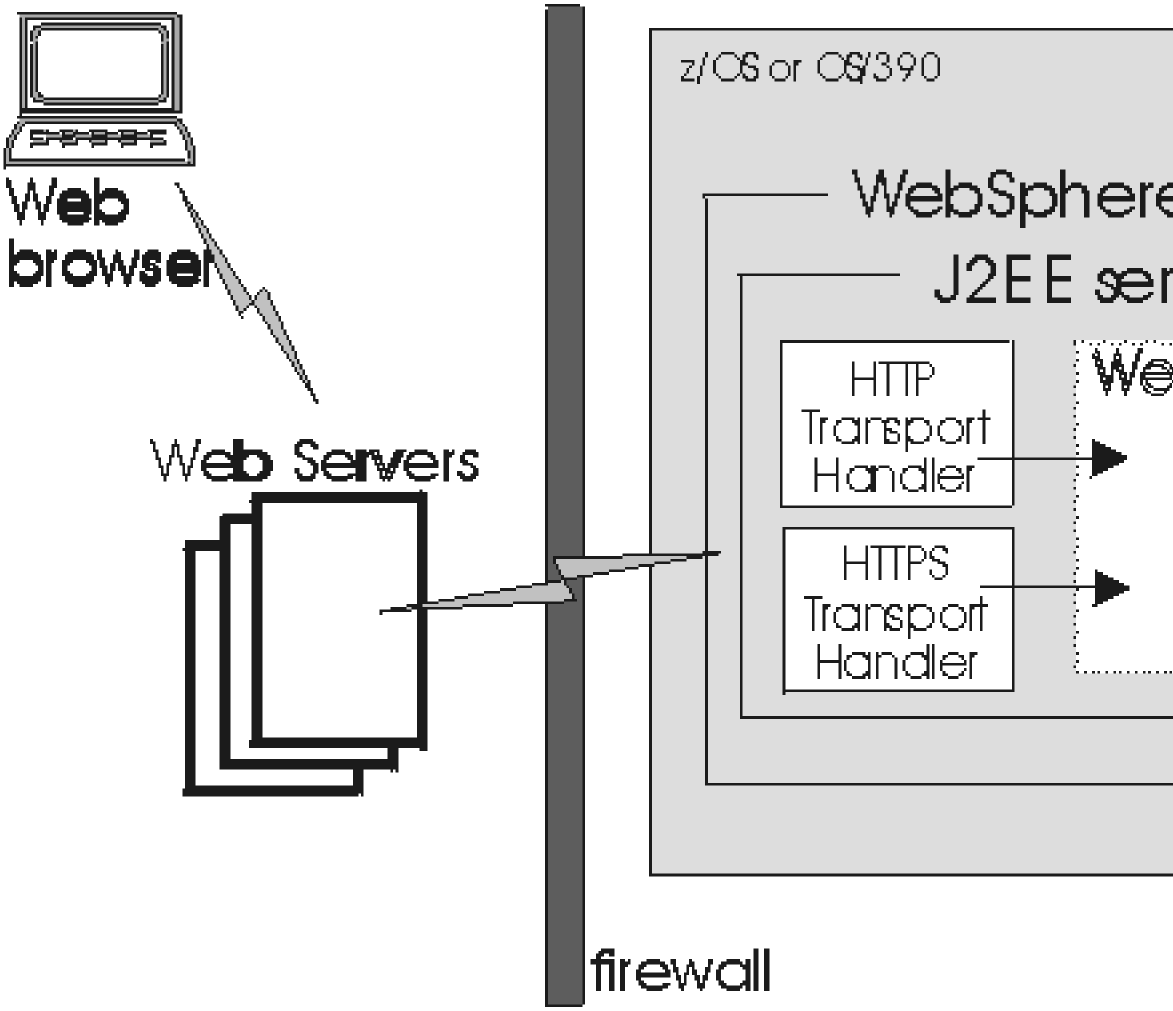
Web components, which are known as Web applications, may consist of any combination of the following parts:

- One or more Java servlets
- Any other Java classes that act as utility classes in support of the servlets
- Static files such as HTML pages and GIF or JPEG images
- JavaServer Pages (JSPs) that format dynamic output

To enable Web applications for use, your Web-serving environment requires an HTTP transport (to receive HTTP requests from a network of browsers using the HTTP access protocol) and an execution environment (to interpret the inbound request and run the appropriate servlet, based on the contents of the inbound request). The WebSphere for z/OS J2EE server includes a choice of two HTTP transports and execution environments:

1. The HTTP internal transport and/or HTTPS internal transport in combination with the Web container in the J2EE server, or
2. The IBM HTTP server for z/OS in combination with the WebSphere HTTP Plug-in for z/OS shipped with the WebSphere for z/OS product, and/or Web container in the J2EE server.

Recommendation: Because the application server is not designed to directly host static content (for example, HTML pages), the recommended configuration places a Web server with a plug-in in front of the application server. See Figure.



Web applications running in the Web container have direct access to resources on z/OS or OS/390, or can access them through Enterprise beans running in any WebSphere for z/OS J2EE server. Web applications use the RMI/IIOP protocol to access Enterprise beans running in J2EE servers on the same or different z/OS or OS/390 images.

For more information about deploying Web applications, see the assembling applications information in the z/OS view of the WebSphere Application Server InfoCenter, which you can access via the WebSphere for z/OS library Web site.

Preparing the base z/OS environment

You must ensure you have an adequate skill level to support the base z/OS environment.

Some z/OS function customization steps you need to do for WebSphere for z/OS can be done before you install and customize WebSphere for z/OS itself. We have put those tasks into this chapter, allowing you to segment your work.

1. Determine your WebSphere for z/OS system requirements.
2. Update your TCP/IP network, as needed.
3. Ensure you understand and set up the relevant security considerations.
4. Set up workload management (WLM).

Once you complete these steps, you need only to verify completeness:

- With the *WebSphere Application Server V5: Program Directory* that all prerequisites have been met. Also verify with the PSP bucket to ensure the latest adjustments have been met.
- With your z/OS system programmers, DB2 (or other DB administrators), and other technical staff to ensure your environment is ready

Determining your skill needs

In assembling your project team, you should consider the skills you need to implement WebSphere for z/OS. We have put together the recommended skillset necessary to support the following configurations:

- Basic configurations
- Advanced configurations
- Production environments

Documentation to support the z/OS skills described in this article can be found at this Web site: z/OS Internet Library

Note: Along with the skills listed below and depending on the needs of the applications you deploy, you may also need skills to configure the resource managers your applications require. Such skills are CICS, DB2 (if applicable), IMS, and MQ.

For basic configurations:

Below are the recommended skills necessary to support a basic configuration:

- z/OS UNIX System Services and the hierarchical file system (HFS) - to set up a functional HFS and UNIX environment
- eNetwork Communications Server (TCP/IP) or equivalent - to configure connectivity for WebSphere for z/OS clients and servers
- Resource recovery services (RRS) - to implement resource recovery services and support two-phase commit transactions

- Security Server (RACF), or the security product you use - to authenticate WebSphere for z/OS clients and servers, and authorize access to resources
- Workload management (WLM)
- SMP/E and JCL
- System logger - to set up logstreams for RRS and the WebSphere for z/OS error log
- Java

For advanced configurations:

When dealing with advanced configurations, you need all the same skills as for basic configurations, plus the following:

- Parallel sysplex - to implement multi-system configurations
- CICS
- IMS
- MQ

For production environments:

As you move your system toward a production environment, you need to have the following system skills available:

- Automatic restart management (ARM)
- System Automation, if you have it installed, or whichever automation you prefer to use
- Sysplex, if you plan to use WebSphere for z/OS in a cell that spans systems
- Secure Sockets Layer (SSL) or Kerberos, if you plan to have security in a distributed network
- Advanced environment SSL if you enable security
- RMF or other performance measurement systems
- Webserver, if you plan to support HTTP clients

Determining WebSphere for z/OS system requirements

The following sections describe system requirements for WebSphere Application Server for z/OS.

Hardware requirements

The hardware requirements for this product are any hardware that supports OS/390 Version 2 Release 10 or any version of z/OS. However, there are significant performance advantages for those applications doing floating point arithmetic if the machine has binary floating point hardware, such as S/390 Parallel Enterprise Server-Generation 5 and later systems.

The LPAR in which the WebSphere for z/OS run-time and initial application servers run requires a minimum of 512 MB of real storage. You may need to increase the real storage size depending on the size and number of application servers you deploy. In addition, you may want to increase your JES spool space if you use WebSphere for z/OS tracing options to the SYSPRINT DD dataset.

You should plan on an extra 3390-3 volume for the distribution and configuration HFS datasets, two additional 3390-3 volumes for the WebSphere product code, and two further 3390-3 volumes if you will use the Integrated JMS Provider.

Recommendation: We recommend you increase your paging subsystem by one 3390-3 volume if your storage is constrained.

Software requirements

You must install, enable, and configure the following z/OS or OS/390 elements, features, and components. Consult the Program Directory and PSP bucket for the required corrective service.

All of the z/OS sources referenced are available at this Web site: http://www.ibm.com/servers/eserver/zseries/zos/bkserv/

- OS/390 Version 2 Release 10 or z/OS configured as a sysplex (at minimum, you need a monoplex). For details, see *z/OS MVS Setting Up a Sysplex*.

  If you want to utilize the dynamic application environment functionality of WebSphere Application Server for z/OS V5, you must run at least z/OS Version 1.2.
- z/OS UNIX System Services (z/OS UNIX) with a hierarchical file system (HFS). For details, see *z/OS UNIX System Services Planning*.
- eNetwork Communications Server (TCP/IP) or equivalent. In this manual, we refer to eNetwork Communications Server, but you may substitute an equivalent product. For details, see *z/OS Communications Server: IP Migration*.
- Workload management (WLM) set up in goal mode. For details, see *z/OS MVS Planning: Workload Management*.

  If you are not running z/OS V1.2 or above with the WLM-DAE support PTF (APAR OW54622), you must complete some additional steps for WLM.
- Resource recovery services (RRS). For details, see *z/OS MVS Programming: Resource Recovery*.
- A security product such as SecureWay Security Server (RACF). In this manual we refer to Security Server in examples, but you may substitute an equivalent security product. For details, see *z/OS Security Server RACF Migration*.
- IBM Developer Kit for OS/390 Java 2 Technology Edition Version 1.1, an element of WebSphere for z/OS, but also available separately. The SDK level supported by this product is 1.4.1.

  Later releases of the IBM Developer Kit for OS/390 Java 2 Technology Edition are not supported.

Regarding optional functions, consult the following table:

Table 4. Software requirements for optional functions

If you plan to use . . .	Then you need . . .	Notes . . .
Kerberos security	OS/390 SecureWay Security Server Network Authentication and Privacy Service for OS/390	For OS/390 V2R10 and z/OS, this support is part of SecureWay Security Server.
Cryptography	Cryptographic Services System SSL, a component of Cryptographic Services Base, an element of z/OS.	For details, see *z/OS System Secure Sockets Layer Programming*.
EJB roles to secure Web components or enterprise beans	z/OS V1R2 Security Server (RACF) or equivalent	For releases of z/OS earlier than z/OS V1R2, install the appropriate APAR for this support.
Java Message Service (JMS)	• Integrated JMS provider • Either of the following: – MQ stack from WebSphere Application Server for z/OS V5 – Full function MQ using MQ 5.3.1	MQSI, a pub sub product that supports z/OS, does not support new function like MDBs.

Table 4. Software requirements for optional functions (continued)

If you plan to use . . .	Then you need . . .	Notes . . .
WebSphere for z/OS IMS Connect V8 support	IMS/TM 6.1.0	
WebSphere for z/OS CICS Transaction Gateway 5.0.1 support	CICS/TS 1.3	
DB2	DB2 V7.1	DB2 is no longer a prerequisite for WebSphere for z/OS, though some functions may still require it. Check your individual configuration.
DB2 SQLJ in J2EE application components	DB2 V7.1 PTF UQ59527	DB2 is no longer a prerequisite for WebSphere for z/OS, though some functions may still require it. Check your individual configuration.
Logging	z/OS system logger	For details, see *z/OS MVS Setting Up a Sysplex*.
Connectors	WebSphere Application Server for z/OS V5 supports any resource adapter that is designed to use the 1.0 level of the J2EE Connector Architecture (JCA). Install the listed IBM connectors if you want to access your IMS or CICS legacy data.	
	For the CICS Transaction Gateway ECI connector: • CICS Transaction Gateway V5.0.1 • CICS Transaction Server V1.3 • WebSphere Studio Application Developer IE V5.0	For configuration details, see the z/OS view of the WebSphere Application Server InfoCenter, which you can access via the WebSphere for z/OS library Web site.
	For the IMS Connector for Java: • IMS Connect for z/OS V2.1 • IMS V8 • WebSphere Studio Application Developer IE V5.0.1	For configuration details, see the z/OS view of the WebSphere Application Server InfoCenter, which you can access via the WebSphere for z/OS library Web site.
	For the IMS JDBC Connector: • IMS V8	For configuration details, see the z/OS view of the WebSphere Application Server InfoCenter, which you can access via the WebSphere for z/OS library Web site.

Updating your TCP/IP network

WebSphere for z/OS follows the CORBA standard, Internet Inter-ORB Protocol (IIOP), for communications. Accordingly, you must consider changes to your TCP/IP network and modify the TCP/IP configuration.

This section provides background information about changes you will need to make to your Domain Name Server (DNS) and TCP/IP. The actual steps to perform are in the customized instructions of the Customization Dialog.

Tips on TCP/IP and WebSphere for z/OS

Consider the following for your TCP/IP network on z/OS.

- You can get started with a simple Domain Name Service (DNS) name server and a single z/OS image, but you should design your initial configuration with growth in mind. You may, for instance, intend to expand your business applications beyond the monoplex to a full sysplex configuration for performance reasons or to prevent a single point of failure. Several considerations come to bear here.

  Several DNS implementations and network router implementations allow the use of a generic location service daemon IP name while dynamically routing network traffic to like-configured servers. If you intend to expand your system beyond a monoplex, it might be worthwhile to use one of these implementations from the start. Non-round-robin DNS name servers limit your ability to expand without retrofitting a name server that allows dynamic network traffic routing.

  Recommendation: 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. For more information, see Sysplex Distributor.

  Beyond Sysplex Distributor, you have your choice of the following DNS and router implementations on or off z/OS:
  - Non-round-robin DNS name servers.
  - Round robin DNS name servers.
  - Connection optimization, a technique used by z/OS that uses DNS and workload management (WLM). WebSphere for z/OS uses connection optimization to prevent a single point of failure. To use connection optimization, you must run the DNS name server on z/OS. For more information, see Connection optimization.
  - Network routers, such as the IBM Network Dispatcher. For more information, see IBM Network Dispatcher.
- Select the location service daemon IP name, choosing any name you want.

  You must define the location service daemon host IP name during installation and customization. Use the location service daemon IP name you chose. See the WebSphere variables in the Administrative Console or the InfoCenter.
- Select the port for the location service daemon server. If you change the location service daemon port number, you can access existing objects after you recycle all your servers. You cannot, however, access the following:
  - Any object handles your application stored to disk
  - Any object references your application stored in the persistent contexts of the name space.

  If you install WebSphere Application Server for z/OS V5 on a system that already contains WebSphere Application Server V4.0.1 for z/OS and OS/390, double-check your current location service daemon port value (default for WebSphere Application Server V4.0.1 for z/OS and OS/390 is 5555).
- Set location service daemon port numbers and IP addresses, found in sysplex-level WebSphere variables. If you need to use the Internet Inter-ORB Protocol (IIOP) through a firewall, ensure that your firewall supports IIOP.

  To configure the ports for a firewall, set up the following WebSphere variables through the Administrative Console:
  - WAS_DAEMON_protocol_iiop_daemon_listenIPAddress
  - WAS_DAEMON_protocol_iiop_daemon_port (base default value is 5655, ND default value is 5755)
  - WAS_DAEMON_protocol_iiop_daemon_port_ssl (base default value is 5656, ND default value is 5756)

When recovering a server somewhere other than its configured system, ensure that the same port is not already in use on the system on which it is recovering. If it is, configure the server with a unique port to avoid a conflict.

When configuring different WebSphere Application Server products on the same system, realize that WebSphere Application Server for z/OS V5 supports dynamic allocation of the ORB SSL port, while WebSphere Application Server Advanced Edition does not even have the ORB SSL port.

HTTP and HTTPS ports are found in individual servers under the Web container transports, which are in the Administrative Console as 'additional properties' on the Web container configuration page (which is off the server configuration page).

Watch for HTTP transport port conflicts if you previously installed WebSphere Application Server V4.0.1 for z/OS and OS/390.

Also ensure you set up the following port assignments, along with those in z/OS port assignments, on servers that require them in the Administrative Console:

– SSL Firewall port
– Web container HTTP transport
– Web container HTTPS transport

See the administrative console and the InfoCenter for more information on the WebSphere variables and how to set their values.

You define ports differently depending on whether they are for the first server or subsequent servers. The first server you create is defined, along with its ports, through the Customization Dialog. You have the ability to explicitly specify the ports as you define the server. Subsequent servers and their ports are defined through the Administrative Console. This means that you define the server first and the ports are automatically assigned. Then, once defined, you can inspect and change the port definitions through the Administrative Console.

- Some ports, such as the ORB SSL port and the server startup status port, are obtained dynamically.
- Other TCP/IP-related activities include setting up NFS, WebServer and Kerberos, which are all optional.
- If you use the DNS on z/OS, you may wish to change the refresh timer interval (-t value) associated with the named location service daemon. The -t value specifies the time (nn, in seconds) between refreshes of cell names and addresses and of the weights associated with those names and addresses. The default is sixty seconds. Reducing the -t value will shorten the lapse time required to register the location service daemon IP name with the DNS, but will also increase DNS processing overhead. In our testing, we used an interval of 10 seconds.

  If you use the z/OS DNS, you have to set a location service daemon variable. Do this by setting WebSphere Variable at cell level:

  ```
  daemon_wlmable=1
  ```

  For details, see *z/OS Communications Server: IP Configuration Reference*.

Setting up workload management (WLM)

WebSphere for z/OS uses the workload management (WLM) function in z/OS to manage workloads. This section helps you get started and is sufficient to get a functioning WebSphere for z/OS system.

WebSphere for z/OS requires that z/OS run workload management in goal mode. If your system runs in compatibility mode, you must implement goal mode. For details on workload management, see *z/OS MVS Planning: Workload Management*, which is available on the z/OS Internet Library Web site.

In addition to setting up workload management in goal mode, you need to define workload management policies for WebSphere for z/OS servers and your business application servers. This section discusses specifics for the run-time servers. For details on workload management and business applications, see the assembling applications information in the z/OS view of the WebSphere Application Server InfoCenter, which you can access via the WebSphere for z/OS library Web site.

Tip: If you are running z/OS V1.2 or above with the WLM-DAE support PTF (APAR OW54622), you do not need to continue.

To get started, you do not need to define special classification rules and work qualifiers, but you might want to do this for your production system.

Each J2EE application server must have a WLM application environment so it can start servant regions. The application environment name is specified as the ″cluster transition name″ in the Customization Dialog.

WebSphere for z/OS servers (with the exception of the location service daemon and node agent) have a controller and one or more servants. The regions are started by the start procedures.

You have to start the controllers for the WebSphere for z/OS servers yourself. This in turn starts the location service daemon. Workload manager dynamically starts the servants as work requests arrive. Thus, you must create WLM application environments that name servant start procedures to start. For example, specify BBO5ASR as the start procedure name that workload management starts for the server.

Each new server that you create for a business application also must be defined to workload management. For more information, see the assembling applications information in the z/OS view of the WebSphere Application Server InfoCenter, which you can access via the WebSphere for z/OS library Web site.
Use the ISPF application IWMARIN0 to define WLM application environments.

Tip: If you are running z/OS V1.2 or above with the WLM-DAE support PTF (APAR OW54622), you can skip this step.

During the customization process, you will be instructed to create a WLM application environment. This section shows how to use the WLM ISPF Dialog to accomplish this.

You must have access to the IWMARIN0 application and be able to update the workload management policies to complete this process.

Use the ISPF application IWMARIN0 to define WLM application environments according to Default server values for WebSphere Application Server for z/OS V5.

Using IWMARIN0:

Workload management must be running in goal mode, and you must have access to a WLM definition, either saved in a WLM definition data set, or active in the WLM couple data set. The user of IWMARIN0 must have update access to the RACF FACILITY class profile MVSADMIN.WLM.POLICY. Perform the following steps to create the BBOC001 application environment.

1. Open the main panel by issuing IWMARIN0. Either load a WLM goal mode definition from a WLM definition dataset, or extract a working goal mode definition from the WLM couple dataset. Then choose option 9:

```
 File Utilities Notes Options Help ----------------------------
 Functionality LEVEL003     Definition Menu            WLM Appl LEVEL004
Command ===> ______________________________________________________________
Definition data set . . : CB.MYCB.WLM
Definition name . . . . .CB390 (Required)
Description . . . . . . . WLMSetup for WebSphere for z/OS
Select one of the following options. . . . . 9__ 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. Fill in the field on the next panel as shown:

```
Application-Environment Notes Options Help ---------------------------------------------------
      Create an Application Environment
Command ===> ______________________________________________________________
Application Environment . . . BBOC001_________________________
Required Description . . . . . . . . .WebSphere for z/OS IVT server___
Subsystem Type . . . . . . . . .CB__
Required Procedure Name . . . . . . . . .BBO5ASR
Start Parameters . . . . . . .JOBNAME=&IWMSSNM.S;,ENV=CELL1.&SYSNAME..;
                                      &IWMSSNM_______________________________;__
                                       _________________________________
Limit on starting server address spaces for a subsystem instance:
2        1. No limit
         2. Single address space per system
         3. Single address space per sysplex
            .----------------------------------------------------------------------.
            | Selection List empty. Define an application environment. (IWMAM600) |
            .----------------------------------------------------------------------.
```

When filling out the previous panel, follow these guidelines:

- The Application Environment name is either the ClusterTransitionName (if the server is not clustered) or the cluster short name (if the server is clustered).
- The recommended JOBNAME value is `&IWMSSNM.S`. So, for example, if the server short name is BBOS001, then set `JOBNAME=BBOS001S`.
- `ENV=...` must be of form *cell_short_name.node_short_name.server_short_name*
  - In a base server, both *cell_short_name* and *node_short_name* are the same as *system_name*.
  - In a network deployment server, both *cell_short_name* and *node_short_name* are the same as *sysplex_name*. As you add base server nodes to the

deployment manager server's cell, you have to update the ENV= parameter of each in the WLM application environment definition. The reason for this change is because a base server node assumes the cell name of the deployment manager server once the base server node is added to the deployment manager server's cell.

- Change CELL1 to your cell_short_name and &SYSNAME to your node_short_name as needed. Otherwise, the defaults will be assumed.
- The value of the &IWMSSNM variable (defined by WLM) is the server short name.

3. Save the application environment. The following panel appears:

```
Application-Environment Notes Options Help
--------------------------------------------------------------------------
         Application Environment Selection List Row 1 to 12 of 12
Command ===> ______________________________________________________________
Action Codes: 1=Create, 2=Copy, 3=Modify, 4=Browse, 5=Print, 6=Delete,
     /=Menu Bar
Action  Application Environment  Name          Description
    __ BBOC001                                 WebSphere for z/OS IVT server
****************************** Bottom of data ********************************
```

4. From the Utilities menu, select Install definition.
5. From the Utilities menu, select Activate service policy.
6. From the File menu, select exit.

You are done when BBOC001 has an application environment.

Preparing for installation and customization

You must prepare z/OS subsystems and do other tasks in this section before you start installation and customization. Additionally, you must determine important information about WebSphere for z/OS and z/OS subsystems before you start customization.

Follow these steps:

1. Prepare your z/OS subsystems (see Preparing the base z/OS environment). In particular, be sure you have followed instructions and tips for the following:
   - System requirements. See Determining WebSphere for z/OS system requirements.
   - TCP/IP. See background information and tips in Updating your TCP/IP network.
   - Security Server (RACF). See Setting up security.
   - Workload manager (WLM). See Setting up workload management (WLM).
   - Resource Recovery Services. See Recommendations for resource recovery services.

2. If you do not already have one, set up a RACF user ID and authorize it to have read/write access to the WebSphere for z/OS files (BBO.* data sets and HFS files).

   Note: In this book we cite product data set names without high-level qualifiers, unless a full data set name is required for clarity, in which case we use BBO as the qualifier.

You are done when you have successfully finished these preparations.

Installing the code through SMP/E

To install the code through SMP/E, follow one of two documents depending on what you use:

- If you use CBPDO, follow the *WebSphere Application Server for z/OS V5.0: Program Directory*.
- If you use ServerPac, follow *ServerPac: Installing Your Order*.

You can find further information on the eSupport Web site at http://www.ibm.com/software/webservers/appserv/zos_os390/support.html, or check the PSP buckets or contact the IBM Software Support Center.

- You can change the high-level qualifier of the installed data sets (not recommended) or the middle-level qualifier.
- If you are installing from a driving system, make sure the maintenance level of the target system meets requirements for WebSphere for z/OS.
- Make sure the product code HFSes are mounted at `/usr/lpp/java` and `/usr/lpp/zWebSphere/V5R0M0`, or at similar mount points of your choice.

Using the Customization Dialog

The Customization Dialog is intended for the system programmer or administrator responsible for installing and customizing WebSphere for z/OS. It lets you separately configure a base Application Server, integral JMS provider, Deployment Manager, and Web Services. In order to use the dialog, you must know or be able to find the system characteristics for the system on which WebSphere for z/OS will run.

The Dialog covers a portion of WebSphere for z/OS customization. Specifically, it creates tailored jobs to:

- Copy the generated jobs into your system libraries.
- Create the run-time HFS structure and the initial environment file.
- Set up WebSphere for z/OS security controls (RACF).
- Define the WebSphere for z/OS run-time configuration (Application Server, integral JMS provider, Deployment Manager, Web Services, location service daemon).
- Run the Installation Verification Test (IVT).

The Dialog also produces a set of instructions for you to follow in order to effectively run these tailored jobs.

Be aware of other versions of WebSphere for z/OS that you have running on your system, as the Customization Dialog does not detect them for you. Keep this in mind when going through the Dialog and, if you are running other versions of WebSphere for z/OS, watch out for such problems as potential location service daemon port collisions or LPA issues.

This section lays out the general flow of the Customization Dialog.

Table 5.

Stage	Description
Start the Customization Dialog	Get the Customization Dialog up and running so you can begin your customization.

Table 5. (continued)

Stage	Description
Determine which task you wish to complete and fill in the corresponding worksheets	You can configure four different environments with the Customization Dialog-a base Application Server node, an integral JMS provider, a Deployment Manager node, and Web Services. Worksheets are provided for each to help you determine what values you should enter in the ″Define Variables″ stage of customization.
Load customization variable information saved from previous configurations, if available	If you previously configured WebSphere for z/OS and saved the variables, you can load them and use them for your current configuration.
Allocate a new or existing target data set	Designate a target data set to store your configuration information.
Define all variables for your particular customization	Using the worksheets that you completed, fill in a value for each variable.
Save your cutomization variable information to a data set for later use	It's always a good idea to save your variables in case you want to reconfigure or use the same values for another configuration or environment.
Generate the customized jobs	This step creates the jobs you will run when stepping through the generated instructions.
View and follow the generated instructions	These instructions step you through the process of running the jobs created by the Customization Dialog. This is the manual portion of actual customization.

1. Start the Customization dialog.
2. Load the variables.
3. Ensure the target data sets have been allocated.
4. Define variables.
5. Save the variables.
6. Generate the jobs from the dialog and then view the generated jobs.

Once these steps are completed you are ready to configure your system using the Customization Dialog.

Steps for starting the Customization Dialog

You must have the product code installed and have access to the product data sets.

Rules: Regarding your display:
- 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 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.

- If you have a 32-row display, you cannot display the PF key settings. Displaying the PF key settings will obscure lines at the bottom of the dialog panels. Issue

  ```
  PFSHOW OFF
  ```

You should complete the worksheets in this section.

Perform the following steps to run the Customization Dialog:

1. From the ISPF command line, enter the following:*ex 'hlq.sbboclib(bbowstrt)' 'options* where:

   hlq High-level qualifier for the SBBOCLIB data set.

   options
   Command options. Enclose any and all options in a single set of quotes.

   appl(*value*)
   Specifies the ISPF application name. This option creates unique ISPF profiles, usually stored in the "userid.ISPF.ISPFPROF" dataset, that are useful when you want to keep saved variables separate from those in other target environments. The default value is

   ```
   BBO5
   ```

   lang(*value*)
   Specifies the national language. Values can be either `ENUS` (English) or `JAPN` (Japanese). The default is `ENUS`.

   Example:

   ```
   ex 'bbo.sbboclib(bbowstrt)' 'appl(bbo5) lang(enus)'
   ```

   Result: You see the splash screen:

```
   -----------------     WebSphere for z/OS Customization      ------------------
Option  ===>
WebSphere Application Server for z/OS V5
Licensed Material - Property of IBM
5655-I35 (C) Copyright IBM Corp. 2000, 2003
All Rights Reserved.
U.S. Government users - RESTRICTED RIGHTS - Use, Duplication, or
Disclosure restricted by GSA-ADP schedule contract with IBM Corp.
Status  = H28W500
Version = 5.00.000
Press ENTER to continue.
```

2. Press Enter. **Result:** You see the "THIRD PARTY LICENSE TERMS AND CONDITIONS, NOTICES AND INFORMATION" panel, which is too large to reprint here.
3. Press PF3.

   Result: You see the following panel:

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

Use this Dialog to customize WebSphere for z/OS for the first time or
to add Deployment Manager functionality to an existing base application
server.  Specify an option and press ENTER.

1  Configure base Application Server node. If you want to configure
   a stand-alone base application server, use this option.

2  Configure integral JMS provider. If you want to configure for an
   Integral JMS Provider, use this option. You must complete option 1
   before starting this option.

3  Configure Deployment Manager node. If you want to configure for a
```

```
   Deployment Manager, use this option. You must complete option 1
   before starting this option.

4  Federate Base Application Server node. If you want to federate the
   base application server node, use this option.  You must complete
   option 3 before starting this option.

5  Configure Web Services. If you want to configure for Web Services,
   use this option. You must complete option 1 before starting this
   option.
```

You have finished starting the Customization Dialog and can now choose the option for the path you would like to follow.

Steps for allocating the target data sets

You must start the Customization Dialog and select your desired task option.

This section tells you how to complete the ″Allocate target data sets″ option that is in all of the five main tasks you can perform. Complete this step if you didn't specify a HLQ (high-level qualifier for the WebSphere for z/OS installation data sets) when you first start the Customization Dialog or want to change it.

1. Type 1 in the *Option* field to select ″Allocate target data sets″.
2. Press Enter. **Result:** You see:

```
-----------------     WebSphere 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 WebSphere 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 WebSphere customization jobs.

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

3. Follow the options in order and enter information from the Allocate Target Data Sets worksheet, then press **Enter**.

You are done when the data set allocation succeeds.

Allocate Target Data Sets - Worksheet: Print out this worksheet for each option in the Customization Dialog and use it to record your values as you customize WebSphere for z/OS.

Table 6. Allocate Target Data Sets

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
High Level Qualifier	(null)	

This panel asks you to specify the high-level qualifiers (HLQs) for the target data sets. Target data sets are those into which the Customization Dialog places the customized jobs and other data. The data sets are:

hlq.CNTL
A partitioned data set of fixed block, 80-byte records, that contains WebSphere for z/OS customization jobs.

hlq.DATA
A partitioned data set of variable length records that contains other data produced by the Customization Dialog.

Steps for loading customization variables

You must start the Customization Dialog and select your desired task option.

This section tells you how to complete the ″Load customization variables″ option that is in all of the five main tasks you can perform.

When you first run the Customization Dialog, the Dialog loads initial default customization variables. If you have previously run the Dialog and saved your variables, you can follow these instructions to re-load those saved variables.

1. On the main Dialog panel, type L in the *Option* field to select ″Load customization variables″.
2. Press Enter. **Result:** You see the Load Customization Variables panel.:

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

Load Customization Variables

 Specify the name of a data set containing the customization variables.
 IBM-supplied defaults are in 'BBO.SBBOEXEC(BBOWVARS)'
 Press Enter to continue.

 Data set name:


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

 Volume:
```

3. Follow the options in order and enter information from the L Load customization variables worksheet (be sure to enclose your data set name in single quotes), then press Enter.

You are done when you successfully load the variables.

L Load customization variables - Worksheet: Print out this worksheet for each option in the Customization Dialog and use it to record your values as you customize WebSphere for z/OS.

Table 7. Load customization variables

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Dsname	(null)	

Specify the data set from which you will prime the variables.

Steps for defining variables

You must start the Customization Dialog and select your desired task option.

This section describes how to complete the ″Define variables″ option that is in all of the five main tasks you can perform.

1. On the main Dialog panel, type 2 in the *Option* field to select ″Define variables″.
2. Press **Enter**.

   Result: You see:

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

Define Variables to configure base Application Server node

   Specify a number and press ENTER to define the WebSphere variables.
   You should review all of the variables in each of the sections, even
   if you are using all of the IBM-supplied defaults.
   Once you complete all sections, press PF3 to return to the main menu.


                                                             Completed?
  1 - System Locations (directories, HLQs, etc)
  2 - System Environment Customization
  3 - Server Customization
  4 - Security Customization
```

   This example is from ″Option 1: Configure base Application Server node,″ but the steps are relatively the same no matter which option you choose.
3. Follow the options in order and enter information from the define variables worksheets:
   - 2 Define Variables to configure base Application Server node (Worksheets)
   - 2 Define Variables to configure Integral JMS Provider (Worksheets)
   - 2 Define Variables to configure Deployment Manager node (Worksheets)
   - 2 Define variables for Federate Base Application Server node
   - 2 Define variables for Web Services

You are done when you finish all the ″Define Variables to configure...″ panels. You may then move ahead and save the variable definitions.

2 Define Variables to configure base Application Server node - Worksheets: This article presents worksheets that will assist you when completing the WebSphere for z/OS Customization Dialog. See 2 Define Variables to configure base Application Server node - Definitions or the Customization Dialog help panels for definitions of each of the terms.

1 System locations (directories, HLQs, etc)

System Locations (1 of 2)

This panel asks you for information about your base operating system and HFS-resident components.

Table 8. System Locations (1 of 2)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
System name	(System on which the Customization Dialog is running)	
Sysplex name	(Sysplex on which the cutsomization Dialog is running)	
PROCLIB	SYS1.PROCLIB	
PARMLIB	SYS1.PARMLIB	
SYSEXEC	(blank)	
SCEERUN	CEE.SCEERUN	
		In link list or LPA?
SBBOLOAD	BBO.SBBOLOAD	
		In link list or LPA?
SBBOLD2	BBO.SBBOLD2	
		In link list or LPA?
SBBOMIG	BBO.SBBOMIG	
		In link list or LPA?
SBBOLPA	BBO.SBBOLPA	
		In link list or LPA?
SBBOEXEC	BBO.SBBOEXEC	
SBBOMSG	BBO.SBBOMSG	

System Locations (2 of 2)

Table 9. System Locations (2 of 2)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Locations of HFS resident components		
WebSphere SMP/E home directory	/usr/lpp/zWebSphere/V5R0M0	
WebSphere JMS Client Java Feature SMP/E home directory	/usr/lpp/mqm/V5R3M1	
java home directory	/usr/lpp/java/IBM/J1.3	

2 System Environment Customization

System Environment Customization (1 of 4)

Table 10. System Environment Customization (1 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
WebSphere HFS information		
Mount point	/WebSphere/V5R0M0	
Name	OMVS.WAS.CONFIG.HFS	
Volume, or '*' for SMS	*	
Primary allocation in cylinders	250	
Secondary allocation in cylinders	100	

System Environment Customization (2 of 4)

Table 11. System Environment Customization (2 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
WebSphere error log stream information		
Name	WAS.ERROR.LOG	
Data class	STANDARD	
Storage class	(null)	
HLQ for data sets	LOGGER	
Is logstream CF resident (Y|N)	Y	
If yes, structure name	WAS_STRUCT	
If no, specify:		
logstream size	3000	
staging size	3000	
RRS log stream information		
Group name	(Cell on which the Customization Dialog is running)	
Data class	STANDARD	
Storage class	(null)	
HLQ for data sets	LOGGER	
Is logstream CF resident (Y|N)	Y	
Create RRS PROC (Y|N)	Y	

System Environment Customization (3 of 4)

This panel asks you to supply some RACF groups and user IDs that are common throughout WebSphere for z/OS. The Dialog creates the RACF commands to define these new user IDs and groups for your security system.

Table 12. System Environment Customization (3 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
WebSphere Configuration Group Information		
Group	WSCFG1	
GID	2500	
WebSphere Administrator Information		
User ID	WSADMIN	
UID	2403	
Password	WSADMIN	
WebSphere Common Groups and User IDs		
Servant group for base servers	WSSR1	
Servant GID for base servers	2501	
Unauthenticated User Definitions for Base servers		
User ID	WSGUEST	
UID	2402	
Group	WSCLGP	
GID	2502	

System Environment Customization (4 of 4)

Table 13. System Environment Customization (4 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
CTRACE Writer Definitions		
Procedure name	BBOWTR	
User ID	STCRACF	
Group	SYS1	
Trace Data Set Information		
Name	SYS1.*systemname*. WAS390.CTRACE	
Volume, or " *" for SMS	*	
Primary space in cylinders	10	
Secondary space in cylinders	0	
Trace Parmlib member suffix	50	

3 Server Customization

Server Customization (1 of 4)

Table 14. Server Customization (1 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Application Server definitions		
WAS home directory	/WebSphere/V5R0M0/AppServer	
Cell name (short)	*sysplex*	
Cell name (long)	*sysplex*	
Node name (short)	*system*	
Node name (long)	*system*	
Server name (short)	BBOS001	
Server name (long)	server1	
Cluster transition name	BBOC001	

Server Customization (2 of 4)

Table 15. Server Customization (2 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Application Server definitions		
Controller information		
Jobname	BBOS001	(cannot change)
Procedure name	BBO5ACR	
Userid	ASCR1	
UID	2431	
Servant information		
Jobname	BBOS001S	(cannot change)
Procedure name	BBO5ASR	
Userid	ASSR1	
UID	2432	

Server Customization (3 of 4)

Table 16. Server Customization (3 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Application Server definitions		
Node host name	(null)	
SOAP JMX Connector port	8880	
DRS Client Address port	7873	
ORB Listener host name	*	
ORB port	2809	
ORB SSL port	0	
HTTP transport host name	*	
HTTP port	9080	
HTTP SSL port	9443	

Server Customization (4 of 4)

Table 17. Server Customization (4 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
location service daemon definitions		
Daemon Home Directory	/WebSphere/V5R0M0/Daemon	(cannot change)
Daemon job name	BBODMNB	
Procedure name	BBO5DMN	
Userid	WSDMNCR1	
UID	2411	
IP name	(null)	
Port	5655	
SSL Port	5656	
Register Daemon with WLM DNS	N	

4 Security Customization

This panel allows you to specify authentication and authorization options for your run-time resources.

Table 18. Security Customization (1 of 1)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Installation dependent WebSphere Security Customization definitions		
SSL Customization		
WebSphere Certificate Authority Keylabel	WAS TestCertAuth	
RACF Keyring Name	WASKeyring	
Use RACF Authorization and Delegation	N	
Support Passtickets for z/SAS authentication	N	
Passticket Profile name	CBS390	(cannot change)
Passticket KEYMASK value	(null)	

Table 18. Security Customization (1 of 1) (continued)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Enable WebSphere Application Server to authenticate RACF users via WebSphere login tokens	Y	
Generate RACF commands for above	Y	
Installation dependant z/OS Security Customization definitions		
Authorize Servers to APPL profile	N	
Sample OPERCMDS definitions for WebSphere Servers	N	

2 Define Variables to configure base Application Server node - Definitions: This article lists definitions for the terms you will come across in the WebSphere for z/OS Customization Dialog. It is a companion to 2 Define Variables to configure base Application Server node (Worksheets).

1 System locations (directories, HLQs, etc)

System Locations (1 of 2)

System name
The system name for the target z/OS system on which WebSphere for z/OS is installed.

Sysplex name
The sysplex name for the target z/OS system on which WebSphere for z/OS is installed.

Tip: If you are not sure what the system name (&SYSNAME) and sysplex name (&SYSPLEX) are, use the console command `D SYMBOLS` on the target z/OS system to display them.

For the following, specify the fully-qualified data set names without quotes.

PROCLIB
An existing procedure library where the WebSphere for z/OS cataloged procedures are to be added.

PARMLIB
An existing parameter library for system definitions to support WebSphere for z/OS. This data set must be in the parmlib concatenation for the target z/OS system.

SYSEXEC
A variable-block (RECFM=VB, LRECL=255) data set into which the customization process places REXX EXECs to be called from TSO, such as the WebSphere for z/OS error log browser, BBORBLOG. You must allocate this data set and concatenate it as part of the SYSEXEC DD allocation in your installation-wide TSO logon PROC or allocation exec.

If your existing SYSEXEC DD data set concatenation consists of fixed-blocked (RECFM=FB) data sets, you must make a copy of the hlq.DATA data set (produced by the Customization Dialog) after the customization process is complete, and place the copy in the SYSEXEC concatenation.

If you do not specify a data set name, the customization process does not place any REXX EXECs in any data set.

Specify the following Language Environment and WebSphere for z/OS data sets and whether they are ("Y") or are not ("N") in the link list or the link pack area (LPA). "N" indicates the generated JCL will contain STEPLIB statements for these data sets. Refer to your SMP/E installation for the location of these data sets listed by their DD Name.

SCEERUN
Your existing Language Environment run-time load module library.

SBBOLOAD
WebSphere for z/OS load module library that you installed through SMP/E. It has members that should go into the link list or LPA.

SBBOLD2
WebSphere for z/OS load module library that you installed through SMP/E. It has members that should go into the link list. **DO NOT** place them in LPA.

SBBOMIG
WebSphere for z/OS IPCS data set that you installed through SMP/E. It is not used during normal operations, but for dump formatting in IPCS only. **DO NOT** place them in LPA.

SBBOLPA
WebSphere for z/OS data set that you installed through SMP/E. It has members that should go into the link list or LPA.

Specify the following WebSphere for z/OS libraries so they can be accessed by the customized job streams the Dialog produces. These data sets must be cataloged.

SBBOEXEC
WebSphere for z/OS variable length file distribution PDS you installed through SMP/E.

SBBOMSG
SBBOMSG WebSphere for z/OS message skeletons for language translation you installed through SMP/E.

System Locations (2 of 2)

Locations of HFS resident components:

WebSphere SMP/E home directory
The name of the directory where WebSphere for z/OS files reside after SMP/E installation.

WebSphere JMS Client Java Feature SMP/E home directory
The name of the directory where the WebSphere JMS Client Java Feature files reside after SMP/E installation.

java home directory
The name of the directory where the Java SDK files reside after SMP/E installation.

2 System Environment Customization

System Environment Customization (1 of 4)

WebSphere configuration HFS Information

Mount point
Read/write HFS directory mount point where application data and environment files are written. The customization process creates this mount point, if it didn't already exist.

Name Hierarchical File System data set mounted at the above mount point.

Rule: You can specify up to 42 characters for the data set name.

Volume, or '*' for SMS
Specify either the DASD volume serial number containing the above data set or " *" to let SMS select a volume. Using " *" requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.

Primary allocation in cylinders
Initial size allocation in cylinders for the above data set.

Recommendation: The minimum suggested size is 250 cylinders (3390).

Secondary allocation in cylinders
Size of each secondary extent in cylinders.

Recommendation: The minimum suggested size is 100 cylinders.

System Environment Customization (2 of 4)

WebSphere Error Logstream Information

Note: Having the error log in the logstream is optional. By default, the error log is written to JES SYSOUT.

Name Name of your WebSphere error log stream that is created.

Rules:

- The name must be 26 characters or fewer.
- Do NOT put quotes around it.

Data class
An existing DFSMS data class for the log stream data set allocation. Applicable only if you use SMS.

Storage class
An existing DFSMS storage class for allocation of the DASD staging data set for this log stream. Applicable only if you use SMS.

HLQ for data sets
The high-level qualifier for your log stream data set name and staging data set name that is created.

Is logstream CF resident (Y|N)
If you want the log stream to be created on a coupling facility, specify " Y". If on DASD, specify " N".

If yes, specify structure name
If using the coupling facility, specify the coupling facility structure to be used for the log stream.

Rule: The name can be 1 to 16 characters, including alphanumeric characters, national characters, and an underscore, where the first character is uppercase alphabetic.

If no, specify: logstream size
Specifies the size, in 4K blocks, of the log stream DASD data sets for the log stream being defined.

If no, specify: staging size
Specifies the size, in 4K blocks, of the DASD staging data set for the log stream being defined.

RRS Logstream Information

If you do not have the RRS log streams set up, the Customization Dialog will create the jobs you can use to set up the log streams.

Group name
Specify the XCF group name.

Recommendation: Use your cell name.

Data class
Specify an existing DFSMS Data Class for the log stream data set allocation. Applicable only if you use SMS.

Storage class
An existing DFSMS storage class for allocation of the DASD staging data set for this log stream. Applicable only if you use SMS.

HLQ for data sets
The high-level qualifier for your log stream data set name and staging data set name.

Is logstream CF resident (Y|N)
If the log stream is to be created on a coupling facility, specify " Y". If on DASD, specify " N".

Create RRS PROC (Y|N)
If you answer " Y", the Dialog copies the ATRRRS cataloged procedure into SYS1.PROCLIB so that RRS can be started.

If you already have RRS set up, specify " N".

System Environment Customization (3 of 4)

To minimize the number of RACF definitions, RACF authorizations are at the group level rather than the user ID level. In a later panel, the Dialog asks for user IDs for the run-time servants and controllers. These user IDs will be connected to their proper RACF groups.

For controllers, which run system authorized code, you can create a single group. Thus, you can safely use the RACF group that the Dialog created for all controllers on your system.

This Dialog creates a RACF group for the initial base application server. Depending on the resources accessed, your installation will either reuse this group or define new ones to handle additional application servers.

Rules:

- User IDs and groups must be unique names (one to eight characters).
- UIDs (user identifiers) must be unique numbers within the system between 1 and 2,147,483,647.
- Do not assign a UID of 0 (Superuser) to any of these users.
- GIDs (group identifiers) should be unique numbers between 1 and 2,147,483,647.

WebSphere Configuration Group Information

Group The default group name for the WebSphere for z/OS administrator and base server. This group allows you to more easily control who can and can not make configuration changes to your servers.

Rule: These two users must have the same default group.

Note: If you are building base Application Server nodes that you might later federate into a Deployment Manager cell, make sure they all have the same group value.

GID The group identifier for the WebSphere for z/OS configuration group.

WebSphere Administrator Information

User ID
The user ID you use to log onto telnet to perform administrative actions against your server.

UID The user identifier for the WebSphere for z/OS user ID.

Password
The password for the WebSphere for z/OS user ID.

WebSphere common groups and user IDs

Servant group for base servers
Specifies an additional group name to which the base server's user ID connects. This is used to control access to resources that are external to the Application Server (for example, DB2).

Servant GID for base servers
A group identifier that the Dialog uses for the WebSphere for z/OS base server's servant group.

Unauthenticated user definitions for base servers

User ID
The default user ID under which the unauthenticated client requests run.

UID The user identifier for the unauthenticated user.

Group The group for unauthenticated users.

GID The group identifier for unauthenticated users.

System Environment Customization (4 of 4)

WebSphere for z/OS uses component trace (CTRACE) to capture and to display trace data in trace data sets. WebSphere for z/OS identifies itself to CTRACE with the ″cell short name.″

CTRACE Writer Definitions

Procedure name
This is the CTRACE external writer start procedure to be created. It is identified in the WebSphere for z/OS CTRACE member (CTIBBOxx) in PARMLIB.

Rule: The name can be 1 to 7 characters.

Userid
RACF user ID to be created and associated with the CTRACE external writer start procedure.

Group RACF group name to be created and associated with this user.

Trace Data Set information

Name Specify a fully-qualified data set name, such as WAS390.CTRACE1, for the data set to be created. The default includes the system name of the system on which the Customization Dialog is running.

Rules:

- You can specify up to 42 characters for the data set name.
- Do not use quotes.

Volume, or ″*″ for SMS
Specify either the DASD volume serial number containing the above data set or " *" to let DFSMShsm select a volume. Using "*" requires SMS. Using "*" requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.

Primary space in cylinders

The primary space for the trace data set.

Secondary space in cylinders

The secondary space for the trace data set.

Trace Parmlib member suffix

3 Server Customization

Server Customization (1 of 4)

WAS home directory

Directory in which the Application Server resides.

Note: You can alter only the portion after ″/WebSphere/V5R0M0.″

Cell name (short)

Parameter passed to the server's start procedures that specifies the location of the cell's configuration files and identifies the cell to certain WebSphere for z/OS-exploited z/OS facilities (for example, SAF).

Rule: Name must be 8 or fewer characters and all uppercase.

Cell name (long)

Primary external identification of this WebSphere for z/OS cell. This name identifies the cell as displayed through the Administrative Console and also appears as part of the directory path under the WebSphere for z/OS home's configuration directory.

Note: If you intend to ever add this base server node to a deployment manager cell, ensure that the base server cell name is different than the deployment manager cell name.

Rule: Name must be 60 or fewer characters and can be of mixed case.

Node name (short)

Parameter passed to the server's start procedures that specifies the location of the node's configuration files and identifies the node to certain WebSphere for z/OS-exploited z/OS facilities (for example, SAF).

Rules:

- Name must be 8 or fewer characters and all uppercase.
- Name must be unique within the cell.

Node name (long)

Primary external identification of this WebSphere for z/OS node. This name identifies the node as displayed through the Administrative Console and also appears as part of the directory path under the WebSphere for z/OS home's configuration directory.

Note: If you intend to ever add this base server node to a deployment manager cell, ensure that all the nodes in the deployment manager cell are unique.

Rules:

- Name must be 60 or fewer characters and can be of mixed case.
- Name must be unique within the cell. The Application Server must be defined on its own node; no other server may exist on the same node as the Application Server.

Server name (short)

Name of the Application Server server. This is the server's jobname, as specified in the MVS START command JOBNAME parameter. This value is also passed as a parameter to the server's start procedures to specify the location of the server's configuration files and identify the server to certain WebSphere for z/OS- exploited z/OS facilities (for example, SAF).

Note: The server short name is used as the server JOBNAME.

Rule: Name must be 7 or fewer characters and all uppercase.

Server name (long)
Name of the Application Server server and the primary external identification of this WebSphere for z/OS server. This name identifies the server as displayed through the Administrative Console and also appears as part of the directory path under the WebSphere for z/OS home's configuration directory.

Rule: Name must be 60 or fewer characters and can be of mixed case.

Cluster transition name
WLM APPLENV (WLM application environment) name for this server.

Note: If this server is converted into a clustered server, this name becomes the cluster short name. The cluster short name is the WLM APPLENV name for all servers that are part of the same cluster.

Rule: Name must be 8 or fewer characters and all uppercase.

Server Customization (2 of 4)

Controller information

Jobname
The jobname, specified in the MVS START command JOBNAME parameter, associated with the Application Server controller.

Procedure name
Name of member in your procedure library to start the Application Server controller.

Rule: Name must be 7 or fewer characters.

User ID
The user ID associated with the Application Server controller.

Note: If you are using a non-IBM security system, the user ID may have to match the procedure name. Please refer to your security system's documentation.

UID The user identifier associated with this user ID.

Rule: UIDs must be unique numbers, between 1 and 2,147,483,647, within the system.

Servant information

Jobname
The jobname specified in the IWMSSNM parameter of the WLM Application Environment for the server.

Procedure name
Name of member in your procedure library to start the Application Server servant.

Rule: Name must be 7 or fewer characters.

User ID
The user ID associated with the Application Server servant.

Note: If you are using a non-IBM security system, the user ID may have to match the procedure name. Please refer to your security system's documentation.

UID The user identifier associated with this user ID.

Rule: UIDs must be unique numbers, between 1 and 2,147,483,647, within the system.

Server Customization (3 of 4)

Application Server definitions

Node host name
IP name or address of the system on which the server is configured. This value is used by other WebSphere for z/OS functions to connect to this server.

Note: The node host name must always resolve to an IP stack on the system where the application server runs. The node host name cannot be a DVIPA or a DNS name that, in any other way, causes the direction of requests to more than one system.

SOAP JMX Connector port
Port number for the JMX HTTP connection to this server based on the SOAP protocol. JMX is used for remote administrative functions, such as invoking scripts through wsadmin.sh.

Rule: Value cannot be 0.

DRS Client Address port
Port address for access to the server's data replication service. This is important for configurations that define replication groups.

Rule: Value cannot be 0.

ORB Listener host name
IP address on which the server's ORB listens for incoming IIOP requests. The default is ″*″, which instructs the ORB to listen on all available IP addresses.

ORB port
Port for IIOP requests which acts as the bootstrap port for this server and also as the port through which the ORB accepts IIOP requests. Port value cannot be 0.

ORB SSL port
Port for secure IIOP requests. The default is ″0″, which allows the system to choose this port.

HTTP transport host name
IP address on which the server's Web container should listen for incoming HTTP requests. The default is ″*″, which instructs the Web container to listen on all available IP addresses.

Note: The ″transport host name″ becomes the ″hostname″ in the virtualhosts.xml file, which makes setting a specific IP address here less than ideal because, if you do so, you are restricting yourself to that IP address until you go into the Administrative Console and add another virtual host.

HTTP port
Port for HTTP requests. Port value cannot be 0.

HTTP SSL port
Port for secure HTTP requests. Port value cannot be 0.

Server Customization (4 of 4)

location service daemon definitions

The location service daemon is the initial point of contact in WebSphere for z/OS for clients and the server contains the location service agent to place sessions in a cell.

Daemon Home Directory
Directory in which the location service daemon resides.

Daemon Job Name
Specifies the jobname of the location service daemon, specified in the JOBNAME parameter of the MVS start command used to start the location service daemon.

Caution: When configuring a second cell, ensure you change the daemon job name from the default or value you used for the first cell.

Note: A server automatically starts the location service daemon if it is not already running.

Procedure name
Name of the member in your procedure library to start the location service daemon.

Rule: Name must be 7 or fewer characters.

User ID
The user ID associated with the location service daemon.

UID The user identifier associated with this user ID.

Rule: UIDs must be unique numbers, between 1 and 2,147,483,647, within the system.

IP Name
The fully-qualified IP name, registered with the Domain Name Service (DNS), that the location service daemon will use.

Note: In a sysplex, you should consider using a virtual IP address (VIPA) for the location service daemon IP name.

Port The port number on which the location service daemon listens.

SSL Port
The port number on which the location service daemon listens for SSL connections.

Register Daemon with WLM DNS

Note: Select the IP name and port number for the location service daemon carefully. You can choose any name you want, but, once chosen, it is difficult to change, even in the middle of customization.

The WebSphere for z/OS run-time requires four base cell servers: Application Server, Deployment Manager, node agent, location service daemon. The panels corresponding to the following tables set up the names, network configuration, start procedures, and user IDs for a base server.

Recommendation: Use the IBM default names the first time you install WebSphere for z/OS to make the installation instructions easier to follow.

For identification, each controller and servant start procedure must have a user ID and will be defined in the STARTED class.

4 Security Customization

In the following, specifying “ Y” (yes) tells the Dialog to define the profile or enable an option in RACF. Specifying “ N” (no) tells the Dialog to not define the profile or enable the option.

Use EJBROLE class to control EJB method access
Specify “ Y” to create the RACF EJBROLE class. EJBROLE controls method accesses for enterprise beans.

Use OPERCMDS to control commands
Specify “ Y” to create the RACF OPERCMDS class. OPERCMDS controls the ability of an operator to start servers.

Use Kerberos over SSL
Specify “ Y” to create Kerberos security. With this option, SSL provides message security and authenticates the server to the client. Kerberos provides the ability for the server to authenticate the client.

Use SSL basic authorization
Specify “ Y” to create SSL basic authorization security. With this option, the server proves its identity by passing a digital certificate to the client. The client proves its identity by passing a user identity and password known by the target server.

Use SSL client certificates
Specify “ Y” to create SSL client certificate security. With this option, both the server and client pass digital certificates to prove their identities to each other.

Use passtickets
Specify “ Y” to use passtickets.

Use CLASS(APPL)
If the APPL class is activated in your installation, specify “ Y” to create a CBS390 profile in that APPL class to represent WebSphere for z/OS and permit the WebSphere for z/OS server identities to that profile. If your installation does not require use of the APPL class, specify “ N”.

Test certificate authority label
The Dialog uses this label to create a test certificate authority certificate.

Recommendation: Use this certificate for testing purposes only.

PassTicket Profile name
Name of the PassTicket Profile

PassTicket KEYMASK value
Specify any string of 16 hexadecimal characters as a mask for PassTickets.

2 Define Variables to configure Integral JMS Provider - Worksheets: This article presents worksheets that will assist you when completing the WebSphere for z/OS Customization Dialog. See 2 Define Variables to configure Integral JMS Provider - Definitions or the Customization Dialog help panels for definitions of each of the terms.

1 System locations (directories, HLQs, etc)

System Locations (1 of 2)

This panel asks you for information about your base operating system, HFS-resident components, and MQ subsystem.

Table 19. System Locations (1 of 2)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
System name	(System on which the Customization Dialog is running)	
Sysplex name	(Sysplex on which the cutsomization Dialog is running)	
Jes3 (Y/N)	N	
MACLIB	SYS1.MACLIB	
PROCLIB	SYS1.PROCLIB	
PARMLIB	SYS1.PARMLIB	
SCSQAUTH	CSQ531.SCSQAUTH	
		In link list or LPA?
SCSQANLx	CSQ531.SCSQANLE	
		In link list or LPA?
SCSQLINK	CSQ531.SCSQLINK	
		In link list or LPA?
SCSQMVR1	CSQ531.SCSQMVR1	
		In link list or LPA?
SCEERUN	CEE.SCEERUN	
		In link list or LPA?
User load	CSQ531.USER.LOAD	
		In link list or LPA?
SCSQLOAD	CSQ531.SCSQLOAD	
		In link list or LPA?
SCSQMACS	CSQ531.SCSQMACS	
		In link list or LPA?
SCSQPROC	CSQ531.SCSQPROC	
		In link list or LPA?
SCSQSNLx	CSQ531.SCSQSNLE	
		In link list or LPA?
User load	CSQ531.USER.LOAD	
		In link list or LPA?

System Locations (2 of 2)

Table 20. System Locations (2 of 2)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
High Level Qualifier of IJP Operational Datasets		
CSQHLQ	CSQ531	
Integral Provider Output Volumes		
BSDS Vol1	(blank)	
BSDS Vol2	(blank)	
LogCopy 1 Vol1	(blank)	
LogCopy 1 Vol2	(blank)	
LogCopy 2 Vol1	(blank)	
LogCopy 2 Vol2	(blank)	
Integral Provider Page Volumes		
PageSet Vol0	(blank)	
PageSet Vol1	(blank)	
PageSet Vol2	(blank)	

Table 20. System Locations (2 of 2) (continued)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
PageSet Vol3	(blank)	
PageSet Vol4	(blank)	
PageSet Vol5	(blank)	
PageSet Vol6	(blank)	
PageSet Vol7	(blank)	
PageSet Vol8	(blank)	

2 Server Customization

These panels allow you to define values for your Integral JMS Provider. The values you initially set for your base application server and location service daemon are displayed for reference purposes only. If you wish to change them, you need to go back and reconfigure your base Application Server.

Recommendation: Use the IBM default names the first time you install WebSphere for z/OS to make the installation instructions easier to follow.

For identification, each controller and servant start procedure must have a user ID and will be defined in the STARTED class. For more information, see Cluster authorizations.

Server Customization (1 of 2)

Table 21. Server Customization (1 of 2)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
WebSphere HFS Information		
Mount point	/WebSphere/V5R0M0	(cannot change)
Application Server definitions		
WAS home directory	/WebSphere/V5R0M0/AppServer	(cannot change)
Cell name (short)	*sysplex*	(cannot change)
Cell name (long)	*sysplex*	(cannot change)
Node name (short)	*system*	(cannot change)
Node name (long)	*system*	(cannot change)
Server name (short)	BBOS001	(cannot change)
Server name (long)	server1	(cannot change)
Integral JMS definitions		
JMS Server name (short)	WMQX	
Command Prefix	+	

Server Customization (2 of 2)

Table 22. Server Customization (2 of 2)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Integral JMS Server Parameters		
Archive retention (days)	9999	
Stat Interval Time (secs)	30	
JMS Server Queued Address port	5558	

Table 22. Server Customization (2 of 2) (continued)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
JMS Server Direct Address port	5559	
JMS Server Security port	5557	

3 Security Customization

This panel allows you to specify authentication and authorization options for your run-time resources.

Security Customization (1 of 1)

Table 23. Security Customization (1 of 1)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Queue Manager Security		
Default User ID for Commands	(blank)	
Resource Auditing (Y/N)	Y	

2 Define Variables to configure Integral JMS Provider - Definitions: This article lists definitions for the terms you will come across in the WebSphere for z/OS Customization Dialog. It is a companion to 2 Define Variables to configure Integral JMS Provider (Worksheets).

1 System locations (directories, HLQs, etc)

System Locations (1 of 2)

System name
The system name, designated in your base cell customization, for the target z/OS system on which WebSphere for z/OS is installed.

Sysplex name
The sysplex name, designated in your base cell customization, for the target z/OS system on which WebSphere for z/OS is installed.

Tip: If you are not sure what the system name (&SYSNAME) and sysplex name (&SYSPLEX) are, use the console command `D SYMBOLS` on the target z/OS system to display them.

Jes3 (Y/N)
Indicate ″Y″ if you are using JES 3 or ″N″ if you are using JES 2.

For the following, specify the fully-qualified data set names without quotes.

Rule: You can specify up to 42 characters for the data set names.

MACLIB

PROCLIB
An existing procedure library where the WebSphere for z/OS cataloged procedures are added.

PARMLIB
An existing parameter library for system definitions to support WebSphere for z/OS.

Specify the following Language Environment and WebSphere for z/OS data sets and whether they are (″Y″) or are not (″N″) in the link list or the link pack area

(LPA). ″N″ indicates the generated JCL will contain STEPLIB statements for these data sets. Refer to your SMP/E installation for the location of these data sets listed by their DD Name.

SCSQAUTH
The main repository for all MQSeries product load modules. It also contains the default parameter modules CSQZPARM and CSQXPARM. Ensure this library is APF-authorized.

SCSQANLx
Contains the load modules for various versions of MQSeries. The different versions, designated by letters in place of the ″x″, are U.S. English: mixed case (″E″), U.S. English: uppercase (″U″), Simplified Chinese (″C″), and Japanese (″K″).

SCSQLINK
The early code library. Contains the load modules that must reside in the link list because they are loaded at system initial program load (IPL). Ensure this library is APF-authorized and in the link list.

SCSQMVR1
Contains the load modules for distributed queuing when using LU 6.2 or TCP/IP with either the OpenEdition sockets or IUCV interface. Ensure this library is APF-authorized.

SCEERUN
The LE runtime library. Access is required. If it is not in your link list, concatenate it in the STEPLIB DD statement. You need to stop and restart your queue manager to do this.

SCSQLOAD
The load library. Contains load modules for non-APF code, user exits, utilities, samples, installation verification programs, and adapter stubs. The library does not need APF-authorization nor must it reside in the link list.

SCSQMACS
Contains Assembler macros including sample macros, product macros, and system parameter macros.

SCSQPROC
Contains sample JCL and default system initialization data sets.

SCSQSNLx
Contains the load modules for various versions of the MQSeries modules that are required for special purpose function (for example, the early code). The different versions, designated by letters in place of the ″x″, are U.S. English: mixed case (″E″), U.S. English: uppercase (″U″), Simplified Chinese (″C″), and Japanese (″K″).

Specify and catalog the following parameter linkedit target data sets.

User load
Controls the logging, archiving, tracing, and connection environments that MQSeries uses in its operation.

System Locations (2 of 2)

Each MQSeries subsystem in WebSphere for z/OS contains two bootstrap data sets (BSDSs), two sets of log data sets, and nine page data sets. Along with the volume 0 page data set, which is fixed and untailorable, there are eight other page data sets that you can organize to suit your needs.

Note: You must define all the BSDS, LogCopy, and PageSet volumes, but you can choose to make the values different or all the same. You might specify the same value for all the volumes if, for example, you run only a test system, but the system performs better if they are spread out.

High Level Qualifier of IJP Operational Data sets:

CSQHLQ
The high level qualifier of your IJP operational data sets.

Integral Provider Output Volumes

BSDS Vol1
The bootstrap data set volume 1.

BSDS Vol2
The bootstrap data set volume 2.

LogCopy 1 Vol1
Volume 1 of copy 1 of the log data set.

LogCopy 1 Vol2
Volume 2 of copy 1 of the log data set.

LogCopy 2 Vol1
Volume 1 of copy 2 of the log data set.

LogCopy 2 Vol2
Volume 2 of copy 2 of the log data set.

Integral Provider Page Volumes

PageSet Vol0
The volume 0 page data set, which is fixed and untailorable.

PageSet Vol1 - PageSet Vol8
Eight volumes of page data sets. You must define them all, but you can choose to make the values different or all the same.

2 Server Customization

Server Customization (1 of 2)

WebSphere HFS Information

Mount point
Read/write HFS directory mount point where application data and environment files are written. The customization process creates this mount point, if it did not already exist.

Application Server definitions

WAS home directory
Directory in which the Application Server resides.

Note: You can alter only the portion after ″/WebSphere/V5R0M0.″

Cell name (short)
Parameter passed to the server’s start procedures that specifies the location of the cell’s configuration files and identifies the cell to certain WebSphere for z/OS-exploited z/OS facilities (for example, SAF).

Cell name (long)
Primary external identification of this WebSphere for z/OS cell. This name identifies the cell as displayed through the Administrative Console and also appears as part of the directory path under the WebSphere for z/OS home’s configuration directory.

Node name (short)
Parameter passed to the server’s start procedures that specifies the location of the node’s configuration files and identifies the node to certain WebSphere for z/OS-exploited z/OS facilities (for example, SAF).

Node name (long)
Primary external identification of this WebSphere for z/OS node. This

name identifies the node as displayed through the Administrative Console and also appears as part of the directory path under the WebSphere for z/OS home's configuration directory.

Server name (short)

Name of the Application Server server. This is the server's jobname, as specified in the MVS START command JOBNAME parameter. This value is also passed as a parameter to the server's start procedures to specify the location of the server's configuration files and identify the server to certain WebSphere for z/OS- exploited z/OS facilities (for example, SAF).

Server name (long)

Name of the Application Server server and the primary external identification of this WebSphere for z/OS server. This name identifies the server as displayed through the Administrative Console and also appears as part of the directory path under the WebSphere for z/OS home's configuration directory.

Integral JMS definitions

JMS Server name (short)

MQSeries subsystem base server name. This value defines the Integral JMS Queue Manager as a z/OS subsystem and forms the name of the Integral JMS Broker.

Rule: The JMS Server name must contain only 4 characters.

Command Prefix

Character string value that WebSphere System Manager uses, in conjunction with the JMS Server short name, to form system commands that direct both the operation and administration of the Integral JMS Queue Manager.

Server Customization (2 of 2)

Integral JMS Server Parameters

Archive retention (days)

Specifies the retention period, in days, set when you create the archive log data set. The default is 9999.

Rule: The parameter must be in the range 0 through 9999.

Stat Interval Time (secs)

The interval, in seconds, between listener restart attempts. The default is 60.

JMS Server Queued Address port

JMS Server Direct Address port

JMS Server Security port

3 Security Customization

Security Customization (1 of 1)

Default User ID for Commands

Specifies the default user ID used for command security checks. Ensure this user ID is defined to the ESM.

Rule: The name must be between 1 and 8 alphanumeric characters and start with a letter.

In the following, specifying " Y" (yes) tells the dialog to define the profile or enable an option in RACF. Specifying " N" (no) tells the Dialog to not define the profile or enable the option.

Resource Auditing (Y/N)

If you want to enable the creation of an audit trail, specify ″Y″. Otherwise, specify ″N″. Creating an audit trail may impact performance.

SAF Authentication (Y/N)

If you want your queue manager to have SAF authentication, specify ″Y″. Otherwise, specify ″N″. Specifying ″Y″ will authenticate userids and passwords and perhaps impact performance.

2 Define Variables to configure Deployment Manager node - Worksheets: This article presents worksheets that will assist you when completing the WebSphere for z/OS Customization Dialog. See 2 Define Variables to configure Deployment Manager node - Definitions or the Customization Dialog help panels for definitions of each of the terms.

1 System locations (directories, HLQs, etc)

This panel asks you for information about your base operating system and HFS-resident components.

System Locations (1 of 2)

Table 24. System Locations (1 of 2)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
System name	(System on which the Customization Dialog is running)	
Sysplex name	(Sysplex on which the cutsomization Dialog is running)	
PROCLIB	SYS1.PROCLIB	
PARMLIB	SYS1.PARMLIB	
SYSEXEC	(blank)	
SCEERUN	CEE.SCEERUN	
		In link list or LPA?
SBBOLOAD	BBO.SBBOLOAD	
		In link list or LPA?
SBBOLD2	BBO.SBBOLD2	
		In link list or LPA?
SBBOMIG	BBO.SBBOMIG	
		In link list or LPA?
SBBOLPA	BBO.SBBOLPA	
		In link list or LPA?
SBBOEXEC	BBO.SBBOEXEC	
SBBOMSG	BBO.SBBOMSG	

System Locations (2 of 2)

Table 25. System Locations (2 of 2)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Locations of HFS resident components		
WebSphere SMP/E home directory	/usr/lpp/zWebSphere/V5R0M0	

Table 25. System Locations (2 of 2) (continued)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
WebSphere JMS Client Java Feature SMP/E home directory	/usr/lpp/mqm/V5R3M1	
java home directory	/usr/lpp/java/IBM/J1.3	

2 System Environment Customization

System Environment Customization (1 of 1)

Table 26. System Environment Customization (1 of 1)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
WebSphere HFS information		
Mount point	/WebSphere/V5R0M0	
Name	OMVS.WAS.CONFIG.HFS	
Volume, or '*' for SMS	*	
Primary allocation in cylinders	250	
Secondary allocation in cylinders	100	

3 Server Customization

Server Customization (1 of 4)

Table 27. Server Customization (1 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Deployment Manager definitions		
WAS home directory	/WebSphere/V5R0M0/DeploymentManager	
Cell name (short)	*sysplex*	
Cell name (long)	*sysplex*	
Node name (short)	*system*	
Node name (long)	*system*	
Server name (short)	BBODMGR	
Server name (long)	dmgr	(cannot change)
Cluster transition name	BBODMGR	

Server Customization (2 of 4)

Table 28. Server Customization (2 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Deployment Manager definitions		
Controller information		
Jobname	BBODMGR	(cannot change)
Procedure name	BBO5DCR	
Userid	DMCR1	
UID	2421	
Servant information		
Jobname	BBODMGRS	(cannot change)

Table 28. Server Customization (2 of 4) (continued)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Procedure name	BBO5DSR	
Userid	DMSR1	
UID	2422	

Server Customization (3 of 4)

Table 29. Server Customization (3 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Deployment Manager definitions		
Node host name	(null)	
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	9090	
HTTP SSL port	9043	

Server Customization (4 of 4)

Table 30. Server Customization (4 of 4)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
location service daemon definitions		
Daemon Home Directory	/WebSphere/V5R0M0/Daemon	(cannot change)
Daemon job name	BBODMNC	
Procedure name	BBO5DMN	
Userid	WSDMNCR1	
UID	2411	
IP name	(null)	
Port	5755	
SSL Port	5756	
Register Daemon with WLM DNS	N	

4 Security Customization

This panel allows you to specify authentication and authorization options for your run-time resources.

Table 31. Security Customization (1 of 1)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Installation dependent WebSphere Security Customization definitions		
SSL Customization		

Table 31. Security Customization (1 of 1) (continued)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
WebSphere Certificate Authority Keylabel	WAS TestCertAuth	
RACF Keyring Name	WASKeyring	
Use RACF Authorization and Delegation	N	
Generate default RACF realm name	N	
Default RACF realm name	MCLXCF01	
Enable WebSphere Application Server to authenticate RACF users via WebSphere login tokens	Y	
Generate RACF commands for above	Y	
Installation dependant z/OS Security Customization definitions		
Authorize Servers to APPL profile	N	
Sample OPERCMDS definitions for WebSphere Servers	N	

2 Define Variables to configure Deployment Manager node - Definitions: This article lists definitions for the terms you will come across in the WebSphere for z/OS Customization Dialog. It is a companion to 2 Define Variables to configure Deployment Manager node - Worksheets.

1 System locations (directories, HLQs, etc)

System Locations (1 of 2)

System name
: The system name for the target z/OS system on which WebSphere for z/OS is installed.

Sysplex name
: The sysplex name for the target z/OS system on which WebSphere for z/OS is installed.

Tip: If you are not sure what the system name (&SYSNAME) and sysplex name (&SYSPLEX) are, use the console command `D SYMBOLS` on the target z/OS system to display them.

For the following, specify the fully-qualified data set names without quotes.

PROCLIB
: An existing procedure library where the WebSphere for z/OS cataloged procedures are to be added.

PARMLIB
: An existing parameter library for system definitions to support WebSphere for z/OS. This data set must be in the parmlib concatenation for the target z/OS system.

SYSEXEC
: A variable-block (RECFM=VB, LRECL=255) data set into which the customization process places REXX EXECs to be called from TSO, such as the WebSphere for z/OS error log browser, BBORBLOG. You must allocate this data set and concatenate it as part of the SYSEXEC DD allocation in your installation-wide TSO logon PROC or allocation exec.

If your existing SYSEXEC DD data set concatenation consists of fixed-blocked (RECFM=FB) data sets, you must make a copy of the hlq.DATA data set (produced by the Customization Dialog) after the customization process is complete, and place the copy in the SYSEXEC concatenation.

If you do not specify a data set name, the customization process does not place any REXX EXECs in any data set.

Specify the following Language Environment and WebSphere for z/OS data sets and whether they are (″Y″) or are not (″N″) in the link list or the link pack area (LPA). ″N″ indicates the generated JCL will contain STEPLIB statements for these data sets. Refer to your SMP/E installation for the location of these data sets listed by their DD Name.

SCEERUN
Your existing Language Environment run-time load module library.

SBBOLOAD
WebSphere for z/OS load module library that you installed through SMP/E. It has members that should go into the link list or LPA.

SBBOLD2
WebSphere for z/OS load module library that you installed through SMP/E. It has members that should go into the link list. **DO NOT** place them in LPA.

SBBOMIG
WebSphere for z/OS IPCS data set that you installed through SMP/E. It is not used during normal operations, but for dump formatting in IPCS only. **DO NOT** place them in LPA.

SBBOLPA
WebSphere for z/OS data set that you installed through SMP/E. It has members that should go into the link list or LPA.

Specify the following WebSphere for z/OS libraries so they can be accessed by the customized job streams the Dialog produces. These data sets must be cataloged.

SBBOEXEC
WebSphere for z/OS variable length file distribution PDS you installed through SMP/E.

SBBOMSG
SBBOMSG WebSphere for z/OS message skeletons for language translation you installed through SMP/E.

System Locations (2 of 2)

Locations of HFS resident components:

WebSphere SMP/E home directory
The name of the directory where WebSphere for z/OS files reside after SMP/E installation.

WebSphere JMS Client Java Feature SMP/E home directory
The name of the directory where the WebSphere JMS Client Java Feature files reside after SMP/E installation.

java home directory
The name of the directory where the Java SDK files reside after SMP/E installation.

2 System Environment Customization

System Environment Customization (1 of 1)

WebSphere configuration HFS Information

Mount point
Read/write HFS directory mount point where application data and environment files are written. The customization process creates this mount point, if it didn’t already exist.

Name Hierarchical File System data set mounted at the above mount point.

Rule: You can specify up to 42 characters for the data set name.

Volume, or ′*′ for SMS
Specify either the DASD volume serial number containing the above data set or “ *” to let SMS select a volume. Using “ *” requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.

Primary allocation in cylinders
Initial size allocation in cylinders for the above data set.

Recommendation: The minimum suggested size is 250 cylinders (3390).

Secondary allocation in cylinders
Size of each secondary extent in cylinders.

Recommendation: The minimum suggested size is 100 cylinders.

3 Server Customization

The WebSphere for z/OS run-time requires four base cell servers: Application Server, Deployment Manager, node agent, location service daemon. The panels corresponding to the following tables set up the names, network configuration, start procedures, and user IDs for a Deployment Manager server.

Recommendation: Use the IBM default names the first time you install WebSphere for z/OS to make the installation instructions easier to follow.

For identification, each controller and servant start procedure must have a user ID and will be defined in the STARTED class.

Server Customization (1 of 4)

WAS home directory
Directory in which the Deployment Manager resides.

Note: You can alter only the portion after ″/WebSphere/V5R0M0.″

Cell name (short)
Parameter passed to the server’s start procedures that specifies the location of the cell’s configuration files and identifies the cell to certain WebSphere for z/OS-exploited z/OS facilities (for example, SAF).

Rule: Name must be 8 or fewer characters and all uppercase.

Cell name (long)
Primary external identification of this WebSphere for z/OS cell. This name identifies the cell as displayed through the Administrative Console and also appears as part of the directory path under the WebSphere for z/OS home’s configuration directory.

Note: Ensure that the cell name is unique (different from those in other deployment manager or base server configurations).

Rule: Name must be 60 or fewer characters and can be of mixed case.

Node name (short)

Parameter passed to the server's start procedures that specifies the location of the node's configuration files and identifies the node to certain WebSphere for z/OS-exploited z/OS facilities (for example, SAF).

Rules:

- Name must be 8 or fewer characters and all uppercase.
- Name must be unique within the cell. The Deployment Manager must be defined on its own node; no other server may exist on the same node as the Deployment Manager.

Node name (long)

Primary external identification of this WebSphere for z/OS node. This name identifies the node as displayed through the Administrative Console and also appears as part of the directory path under the WebSphere for z/OS home's configuration directory.

Note: Ensure that the node name is unique (different from those in other deployment manager or base server configurations).

Rules:

- Name must be 60 or fewer characters and can be of mixed case.
- Name must be unique within the cell. The Deployment Manager must be defined on its own node; no other server may exist on the same node as the Deployment Manager.

Server name (short)

Name of the Deployment Manager server. This is the server's jobname, as specified in the MVS START command JOBNAME parameter. This value is also passed as a parameter to the server's start procedures to specify the location of the server's configuration files and identify the server to certain WebSphere for z/OS- exploited z/OS facilities (for example, SAF).

Rule: Name must be 7 or fewer characters and all uppercase.

Server name (long)

Name of the Deployment Manager server and the primary external identification of this WebSphere for z/OS server. This name identifies the server as displayed through the Administrative Console and also appears as part of the directory path under the WebSphere for z/OS home's configuration directory.

Rule: Name must be 60 or fewer characters and can be of mixed case.

Cluster transition name

WLM APPLENV name for this server.

Note: The Deployment Manager is not clusterable, so this value never actually becomes the cluster short name of this server's cluster. However, like an Application Server, the Deployment Manager still needs an APPLENV, so the cluster transition name is used for this purpose.

Rule: Name must be 8 or fewer characters and all uppercase.

Server Customization (2 of 4)

Controller information

Jobname

The jobname, specified in the MVS START command JOBNAME parameter, associated with the Deployment Manager controller.

Procedure name
Name of member in your procedure library to start the Deployment Manager controller.

Rule: Name must be 7 or fewer characters.

User ID
The user ID associated with the Deployment Manager controller.

Note: If you are using a non-IBM security system, the user ID may have to match the procedure name. Please refer to your security system's documentation.

UID The user identifier associated with this user ID.

Rule: UIDs must be unique numbers, between 1 and 2,147,483,647, within the system.

Servant information

Jobname
The jobname specified in the IWMSSNM parameter of the WLM Application Environment for the server.

Procedure name
Name of member in your procedure library to start the Deployment Manager servant.

Rule: Name must be 7 or fewer characters.

User ID
The user ID associated with the Application Server servant.

Note: If you are using a non-IBM security system, the user ID may have to match the procedure name. Please refer to your security system's documentation.

UID The user identifier associated with this user ID.

Rule: UIDs must be unique numbers, between 1 and 2,147,483,647, within the system.

Server Customization (3 of 4)

Deployment Manager definitions

Node host name
IP name or address of the system on which the server is configured. This value is used by other WebSphere for z/OS functions to connect to this server.

Note: The node host name must always resolve to an IP stack on the system where the application server runs. The node host name cannot be a DVIPA or a DNS name that, in any other way, causes the direction of requests to more than one system.

SOAP JMX Connector port
Port number for the JMX HTTP connection to this server based on the SOAP protocol. JMX is used for remote administrative functions, such as invoking scripts through wsadmin.sh.

Rule: Value cannot be 0.

CELL DISCOVERY ADDRESS port
Port number used by node agents to connect to this Deployment Manager server.

DRS Client Address port

Port address for access to the server's data replication service. This is important for configurations that define replication groups.

Rule: Value cannot be 0.

ORB Listener host name

IP address on which the server's ORB listens for incoming IIOP requests. The default is ″*″, which instructs the ORB to listen on all available IP addresses.

ORB port

Port for IIOP requests which acts as the bootstrap port for this server and also as the port through which the ORB accepts IIOP requests. Port value cannot be 0.

ORB SSL port

Port for secure IIOP requests. The default is ″0″, which allows the system to choose this port.

HTTP transport host name

IP address on which the server's Web container should listen for incoming HTTP requests. The default is ″*″, which instructs the Web container to listen on all available IP addresses.

Note: The ″transport host name″ becomes the ″hostname″ in the virtualhosts.xml file, which makes setting a specific IP address here less than ideal because, if you do so, you are restricting yourself to that IP address until you go into the Administrative Console and add another virtual host.

HTTP port

Port for HTTP requests. Port value cannot be 0.

HTTP SSL port

Port for secure HTTP requests. Port value cannot be 0.

Server Customization (4 of 4)

location service daemon definitions

The location service daemon is the initial point of contact in WebSphere for z/OS for clients and the server contains the location service agent to place sessions in a cell.

Daemon Home Directory

Directory in which the location service daemon resides.

Daemon Job Name

Specifies the jobname of the location service daemon, specified in the JOBNAME parameter of the MVS start command used to start the location service daemon.

Caution: When configuring a second cell, ensure you change the daemon job name from the default or value you used for the first cell.

Note: A server automatically starts the location service daemon if it is not already running.

Procedure name

Name of the member in your procedure library to start the location service daemon.

Rule: Name must be 7 or fewer characters.

User ID

The user ID associated with the location service daemon.

UID The user identifier associated with this user ID.

Rule: UIDs must be unique numbers, between 1 and 2,147,483,647, within the system.

IP Name
The fully-qualified IP name, registered with the Domain Name Service (DNS), that the location service daemon will use.

Note: The location service daemon IP name must be unique in the sysplex and must not have the same value as any one system's node host name.

Note: In a sysplex, you should consider using a virtual IP address (VIPA) for the location service daemon IP name.

Port The port number on which the location service daemon listens.

SSL Port
The port number on which the location service daemon listens for SSL connections.

Note: Select the IP name and port number for the location service daemon carefully. You can choose any name you want, but, once chosen, it is difficult to change, even in the middle of customization.

4 Security Customization

Security Customization (1 of 1)

In the following, specifying " Y" (yes) tells the dialog to define the profile or enable an option in RACF. Specifying " N" (no) tells the Dialog to not define the profile or enable the option.

Use EJBROLE class to control EJB method access
Specify " Y" to create the RACF EJBROLE class. EJBROLE controls method accesses for enterprise beans.

Use OPERCMDS to control commands
Specify " Y" to create the RACF OPERCMDS class. OPERCMDS controls the ability of an operator to start servers.

Use Kerberos over SSL
Specify " Y" to create Kerberos security. With this option, SSL provides message security and authenticates the server to the client. Kerberos provides the ability for the server to authenticate the client.

Use SSL basic authorization
Specify " Y" to create SSL basic authorization security. With this option, the server proves its identity by passing a digital certificate to the client. The client proves its identity by passing a user identity and password known by the target server.

Use SSL client certificates
Specify " Y" to create SSL client certificate security. With this option, both the server and client pass digital certificates to prove their identities to each other.

Use passtickets
Specify " Y" to use passtickets.

Use CLASS(APPL)
If the APPL class is activated in your installation, specify " Y" to create a CBS390 profile in that APPL class to represent WebSphere for z/OS and permit the WebSphere for z/OS server identities to that profile. If your installation does not require use of the APPL class, specify " N".

Test certificate authority label

The Dialog uses this label to create a test certificate authority certificate.

Recommendation: Use this certificate for testing purposes only.

PassTicket Profile name

Name of the PassTicket Profile

PassTicket KEYMASK value

Specify any string of 16 hexadecimal characters as a mask for PassTickets.

2 Define variables for Federate Base Application Server node - Worksheets and definitions: This article presents worksheets and definitions that will assist you when completing the WebSphere for z/OS Customization Dialog. See the Customization Dialog help panels for the same definitions of each of the terms.

Define variables for Web Services (1 of 1)

Table 32. Define variables for Web Services (1 of 1)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
WAS home directory	/WebSphere/V5R0M0/AppServer	(cannot change)
Node host name	(null)	
JMX Soap port	8879	
Deployment Manager Security is Enabled	N	
Userid	WSADMIN	
Password	WSADMIN	
Include Apps	Y	
Base server ORB port	9810	
Server name (short)		
Server name (long)		(cannot change)
SOAP JMX Connector port	8878	
DRS Client Address port	7888	
Node Discovery port	7272	
Node Multicast Discovery port	5000	
ORB Listener host name	*	
ORB port	2809	
ORB SSL port	0	

WAS home directory

Directory in which the Application Server resides.

Note: You can alter only the portion after ″/WebSphere/V5R0M0.″

Node host name

IP name or address of the system on which the server is configured. This value is used by other WebSphere for z/OS functions to connect to this server.

Note: The node host name must always resolve to an IP stack on the system where the application server runs. The node host name cannot be a DVIPA or a DNS name that, in any other way, causes the direction of requests to more than one system.

JMX Soap port

Deployment Manager Security is Enabled

User ID

The user ID you use to log onto telnet to perform administrative actions against your server.

Password
The password for the WebSphere for z/OS user ID.

Include Apps

Base server ORB port

Server name (short)
Name of the Application Server server. This is the server's jobname, as specified in the MVS START command JOBNAME parameter. This value is also passed as a parameter to the server's start procedures to specify the location of the server's configuration files and identify the server to certain WebSphere for z/OS- exploited z/OS facilities (for example, SAF).

Note: The server short name is used as the server JOBNAME.

Rule: Name must be 7 or fewer characters and all uppercase.

Server name (long)
Name of the Application Server server and the primary external identification of this WebSphere for z/OS server. This name identifies the server as displayed through the Administrative Console and also appears as part of the directory path under the WebSphere for z/OS home's configuration directory.

Rule: Name must be 60 or fewer characters and can be of mixed case.

SOAP JMX Connector port
Port number for the JMX HTTP connection to this server based on the SOAP protocol. JMX is used for remote administrative functions, such as invoking scripts through wsadmin.sh.

Rule: Value cannot be 0.

DRS Client Address port
Port address for access to the server's data replication service. This is important for configurations that define replication groups.

Rule: Value cannot be 0.

Node Discovery port

Node Multicast Discovery port

ORB Listener host name
IP address on which the server's ORB listens for incoming IIOP requests. The default is "*", which instructs the ORB to listen on all available IP addresses.

ORB port
Port for IIOP requests which acts as the bootstrap port for this server and also as the port through which the ORB accepts IIOP requests. Port value cannot be 0.

ORB SSL port
Port for secure IIOP requests. The default is ″0″, which allows the system to choose this port.

2 Define variables for Web Services - Worksheets and definitions: This article presents worksheets and definitions that will assist you when completing the WebSphere for z/OS Customization Dialog. See the Customization Dialog help panels for the same definitions of each of the terms.

Define variables for Web Services (1 of 1)

Table 33. Define variables for Web Services (1 of 1)

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
WebSphere SMP/E home directory	/usr/lpp/zWebSphere/V5R0M0	
Mount point	/WebSphere/V5R0M0	
Web Services Home directory	*	
Owner	WSADMIN	(cannot change)
Group	WSCFG1	(cannot change)
SBBOEXEC	BBO.SBBOEXEC	

WebSphere SMP/E home directory
The name of the directory where WebSphere for z/OS files reside after SMP/E installation.

Mount point
Read/write HFS directory mount point where application data and environment files will be written. The customization process creates this mount point, if it didn't already exist.

Web Services Home directory
The name of the directory under the mount point that corresponds to the Application Server on which you want to install the Web Services Technology Preview. The default is "*", which specifies all Application Servers under that mount point.

Owner and Group information

These are the values you initially set for your base system and are displayed for reference purposes only. If you wish to change them, you need to go back and reconfigure your base server.

Owner
USS user ID assigned to the files that move into the Application Server's directories.

Group Group assigned to the files that move into the application server's directories.

SBBOEXEC
SMP/E distribution PDS where WebSphere for z/OS is installed.

Steps for saving the customization variables

You must have completed the definition of the variables, which is outlined in the article Steps for defining variables.

This section tells you how to complete the "Save customization variables" option that is in all of the five main tasks you can perform.

1. On the main Dialog panel, type S in the *Option* field to select "Save customization variables".
2. Press Enter. **Result:** You see the Save Customization Variables panel.:

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

Save Customization Variables

  Specify the name of a sequential data set to contain the customization
  variables. If the data set does not exist, the Dialog displays the
```

```
 Allocate New Data Set panel, through which you can allocate a data set.
 Press Enter to continue.

Data set name:
```

3. Follow the options in order and enter information from the Save customization variables worksheet (be sure to enclose your data set name in single quotes), then press Enter.

You are done when you successfully save the variables.

S Save customization variables - Worksheet: Print out this worksheet for each option in the Customization Dialog and use it to record your values as you customize WebSphere for z/OS.

Table 34. Save customization variables

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)
Dsname	(null)	

Specify the name of the data set into which you want to save the customization variables.

Rules:
- The data set must be a sequential data set. Do not specify a member name.
- Place quotes around the data set name.

Steps for generating customization jobs

You must complete the process of Defining variables, which is outlined in Steps for defining variables.

This article describes how to complete the ″Generate customization jobs″ option that is in all of the five main tasks you can perform.

Recommendation: When you have finished entering all your customization data, before you generate the customization jobs, use the S option to save your customization variables for future reference.

1. On the main Dialog panel, type 3 in the *Option* field to select ″Generate customization jobs″.
2. Press Enter.

   Result: If all variables are defined correctly, you see the Specify Job Cards panel:

```
-----------------     WebSphere 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 using other data sets, then END from this
 panel and select option 1 (Allocate target data sets).
```

```
All the jobs that will be tailored for you will need a jobcard.
Please enter a valid jobcard 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 as needed.

Specify the job cards.  Press ENTER to continue.

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

3. Fill in the job card information using the Generate customization jobs worksheet at Generating the jobs, then press **Enter**.

You are done when all the jobs are generated. You may then move ahead to viewing the generated jobs.

Generate customization jobs - Worksheet: Print out this worksheet for each option in the Customization Dialog and use it to record your values as you customize WebSphere Application Server for z/OS.

Table 35. Generate customization jobs

Item	Value in the Dialog after you load IBM defaults	Your value (Fill in the blanks)

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

Specify the job card according to your installation requirements.

The Dialog generates a job name and the ″JOB″ keyword for each job.

Steps for viewing and following the generated customization instructions

Before you begin: You must complete the generation of the customization jobs, which is detailed in the article Steps for generating customization jobs.

This section tells you how to complete the ″View instructions″ option that is in all of the five main tasks you can perform. Once you view the steps, you can then follow them.

There are four phases you go through when following the generated instructions:

1. Perform manual system updates:
   - WLM
   - SCHEDxx
   - APF
   - SMF
   - BPXPRMxx
   - LPA
   - Link list.
2. Run system-related jobs:
   - Logstream

- RRS
- CTRACE
- RACF profiles and resources.

3. Run your server's HFS-related jobs:
   - Create and mount the HFS file system.
   - Populate the HFS with directories and XML.
   - Create the was.env files for the location service daemon and servers.
   - Batch install the Administration application.
   - Clean up the HFS structure.
4. Start your servers:
   - Start CTRACE.
   - Start your controller. (The controller automatically starts the location service daemon, and WLM starts the servants.)

1. On the main Dialog panel, type 4 in the *Option* field to select ″View instructions″.
2. Press Enter. You will then see the generated instructions file.
3. View the instructions. You may print the instructions according to your local print procedures.

You are done when you view or print the instructions. You can now go on to follow them, using the worksheets for your task as guides.

- Following the customized instructions
- Following the customized instructions
- Following the customized instructions
- Following the customized instructions
- Following the customized instructions

Following the customized Application Server node instructions: The major stages of the Application Server node customization process are:

Table 36. Following the customized Application Server node instructions

Stage	Description
1	Make a variety of configuration changes to your z/OS system configuration (PARMLIB, TCP/IP, workload management, and so forth). The customized instructions provide details and pointers to relevant documentation.
2	If you require message translation, set up MMS to translate messages for WebSphere for z/OS with job BBOMSGC.
3	Define log streams used by WebSphere for z/OS and RRS through jobs BBOERRLG and BBORRSLS. You do not need to run BBORRSLS if RRS is already running on your target MVS system.
4	Allocate the CTRACE data set through job BBOWCTR.
5	Create a customized set of RACF commands for initial WebSphere for z/OS security setup through job BBOCBRAJ. The RACF commands are saved in member BBOWBRAK of the *hlq*. DATA data set. Job BBOWBRAK executes these RACF commands. Later, you can use the RACF commands saved in BBOWBRAK to help in defining security for additional servers or users.
6	Allocate and mount the WebSphere for z/OS run-time HFS through job BBOWCHFS. If your root HFS is mounted read-only, you may need to create one or more directories manually. See the instructions for details. Job BBOMCFG creates subdirectories in the WebSphere for z/OS run-time HFS and job BBOMCFGU, which is optional, configures the runtime HFS to include the directory for UDDIReg.
7	Copy customized PARMLIB and PROCLIB members into their proper locations using job BBOWCPY1.

Table 36. Following the customized Application Server node instructions (continued)

Stage	Description
8	Copy customized HFS files into their proper locations using job BBOWCPY2.
9	Create the was.env file for the location service daemon and the application servers through job BBOWC2N.
10	Install the Administrative Console and the IVT application through job BBOWIAPP.
11	Complete the HFS initialization with job BBOMCFG2.
12	Activate Resource Recovery Services (RRS), if it is not already active, with the MVS command `START ATRRRS,SUB=MSTR`.
13	Start the CTRACE writer used by WebSphere for z/OS with the MVS command `TRACE CT,WTRSTART= BBOWTR .`

14	Start the Application Server with the following MVS command:
	`S`
	`<`*`controller_procname`*`,JOBNAME=`*`server_shortname`*`,ENV=<`*`cell_shortname.node_shortname.serv`*
	process causes information to be stored in the HFS. Once the process is complete, you have a working WebSphere for z/OS run-time that you will use to run the Installation Verification Test (using job BBOWIVT) and your own applications.

15	Use BBOW5SH to execute certain shell scripts from the WebSphere for z/OS bin directory in order to carry out certain administrative tasks.

Following the customized integral JMS provider instructions: The major stages of the integral JMS provider customization process are:

Table 37. Following the customized integral JMS provider instructions

Stage	Description
1	Make a variety of configuration changes to your z/OS system configuration. The customized instructions provide details and pointers to relevant documentation.
2	Establish security identity for the MQ started tasks.
3	Creates dual bootstrap data sets for usage by the Integral JMS Provider WebSphere MQ Queue Manager with job BBOJBSDS.
4	Create page data sets for usage by the Integral JMS Provider WebSphere MQ Queue Manager with job BBOJPAGR.
5	Assemble your parameters with jobs BBOJXPRR and BBOJZPRR. BBOJXPRR assembles the X parameters for usage by the Integral JMS Provider WebSphere MQ Channel Initiator, and BBOJZPRR assembles the Z parameters for usage by the Integral JMS Provider WebSphere MQ Queue Manager.
6	Copy the tailored start procedures and parameters to the run-time libraries with jobs BBOWCPYJ and BBOWCPYZ.
7	Stop the Application Server with the MVS command `STOP BBOS001`.
8	Run job BBOWCPJ2 to write z/OS UNIX resident files into the HFS for use by the application server.
9	Initialize the Integral JMS Provider runtime environment in the HFS for the application server with job BBOWJWC.

Table 37. Following the customized integral JMS provider instructions (continued)

Stage	Description

10	Start the Application Server with the following MVS command:
	`S`
	`<controller_procname,JOBNAME=server_shortname,ENV=<cell_shortname.node_shortname.server_`
	process causes information to be stored in the HFS. Once the process is complete, you have a working WebSphere for z/OS run-time that you will use to run the Installation Verification Test (using job BBOWIVT) and your own applications.

Following the customized Deployment Manager node instructions: The major stages of the Deployment Manager node customization process are:

Table 38. Following the customized Deployment Manager node instructions

Stage	Description
1	Make a variety of configuration changes to your z/OS system configuration (PARMLIB, TCP/IP, workload management, and so forth). The customized instructions provide details and pointers to relevant documentation.
2	Create a customized set of RACF commands for initial WebSphere for z/OS security setup through job BBODBRAJ. The RACF commands are saved in member BBODBRAK of the *hlq*. DATA data set. Job BBODBRAK executes these RACF commands. Later, you can use the RACF commands saved in BBODBRAK to help in defining security for additional servers or users.
3	Allocate and mount the WebSphere for z/OS run-time HFS through job BBOWCHFS. If your root HFS is mounted read-only, you may need to create one or more directories manually. See the instructions for details. Job BBOMCFG creates subdirectories in the WebSphere for z/OS run-time HFS.
4	Configure the runtime HFS to include the directory for UDDIReg with the job BBOMCFGU, if desired.
5	Copy customized PROCLIB members into their proper locations using job BBODCPY1.
6	Copy customized HFS files into their proper locations using job BBODCPY2.
7	Create the was.env file for the location service daemon and the application servers through job BBODC2N.
8	Install the Administrative Console and, if you are running a base configuration, the IVT application through job BBODIAPP.
9	Complete the HFS initialization with job BBODCFG2.
10	Activate Resource Recovery Services (RRS), if it is not already active, with the MVS command `START ATRRRS,SUB=MSTR`.
11	Start the Deployment Manager (and, hence, the location service daemon) with the MVS command `START BBO5DCR,JOBNAME=BBODMGR`. This process stores information in the HFS. Once the process is complete, you have a working WebSphere for z/OS network deployment run-time that you will use to run your own applications.
12	Stop the Application Server with the MVS command `STOP BBOS001`.
13	Add the application server to the deployment manager's cell with job BBOWADDN.
14	Update the application server's WLM application environment according to the examples in the generated instructions.
15	Start the node agent server with the MVS command `START BBO5ACR,JOBNAME=BBON001` if desired.

Table 38. Following the customized Deployment Manager node instructions (continued)

Stage	Description

16	Start the Application Server, if desired, with the following MVS command:
	`S`
	`<controller_procname,JOBNAME=server_shortname,ENV=<cell_shortname.node_shortname.serve`
	Note: You must start the node agent server before you start the Application Server.

Following the customized Federate Base Application Server node instructions: The major stages of the base Application Server node federation process are:

Table 39. Following the customized Federate Base Application Server node instructions

Stage	Description
1	Make a variety of configuration changes to your z/OS system configuration. The customized instructions provide details and pointers to relevant documentation.
2	Add the application server(s) associated with the base application server node to the deployment manager's cell with job BBOWADDN.
3	Update the application server's WLM application environment.
4	Start the node agent server with the MVS command `START controller_procname,JOBNAME=server_shortname,ENV=cell_shortname.node_shortname.server_`
5	Start the JMS server with the MVS command `START controller_procname,JOBNAME=server_shortname,ENV=cell_shortname.node_shortname.server_`

6	Start the Application Server with the following MVS command:
	`S`
	`<controller_procname,JOBNAME=server_shortname,ENV=<cell_shortname.node_shortname.serve`

Following the customized Web Services instructions: The major stages of the Web Services customization process are:

Table 40. Following the customized Web Services instructions

Stage	Description
1	Make a variety of configuration changes to your z/OS system configuration. The customized instructions provide details and pointers to relevant documentation.
2	Configure the runtime HFS to include the directory for ″webservices″ with job BBOMCFGW.

z/OS port assignments

Purpose

The following table lists the default server values for WebSphere Application Server for z/OS V5.

z/OS port assignments

The following table lists the z/OS port assignments.

Port	Base location service daemon	Base Application Server	ND location service daemon	ND Application Server	Node Agent	JMS Server	Deploy. Manager
HTTP		9080		9080			9090
HTTP/S		9443		9443			9043
Bootstrap		2809		9810	2809	2810	9809
ORB	5655	2809	5755	9810	2809	2810	9809
ORB SSL	5656	0	5756	0	0	0	0
SOAP/JMX		8880		8880	8878	8876	8879
DRS		7873		7873	7888		7989
JMS Queued		5558				5558	
JMS Direct		5559				5559	
Node Discovery					7272		
Node Multi-cast Discovery					5000		
Cell Discovery							7277

Automatic restart management

If you have an application that is critical for your business, you need facilities to manage failures. z/OS provides rich automation interfaces, such as automatic restart management, that you can use to detect and recover from failures. Automatic restart management handles the restarting of servers when failures occur.

WebSphere Application Server for z/OS uses the z/OS Automatic Restart Management (ARM) to recover application servers. Each application server running on a z/OS system (including servers you create for your business applications) are automatically registered with an ARM group. Each registration uses a special element type called SYSCB, which ARM treats as restart level 3, assuring that RRS restarts before any application server.

Tips:

- If you have automatic restart management (ARM) enabled on your system, you may wish to disable ARM for the WebSphere for z/OS address spaces before you install and customize WebSphere for z/OS. During customization, job errors may cause unnecessary restarts of the WebSphere for z/OS address spaces. After installation and customization, consider enabling ARM.
- If you are ARM-enabled and you cancel or stop a server, it will restart in place using the `armrestart` command.
- If you start the location service daemon on a system that already has one, it will terminate.
- Every other server will come up on a dynamic port unless the configuration has a fixed port. Therefore, the fixed ports must be unique in a sysplex.
- If you issue STOP, CANCEL, or MODIFY commands against server instances, be aware of how automatic restart management behaves regarding WebSphere for z/OS server instances:

Table 41. Behavior of automatic restart management regarding WebSphere for z/OS server instances

If you issue . . .	Automatic restart management will . . .
STOP *address_space*	not restart the address space
CANCEL *address_space*	not restart the address space
CANCEL *address_space*, ARMRESTART	restart the address space
MODIFY *address_space*, CANCEL	not restart the address space
MODIFY *address_space*, CANCEL,ARMRESTART	restart the address space

Activating automatic restart management

You must have access to the couple data set format utility, IXCL1DSU, in SYS1.MIGLIB. If you plan to modify the automatic restart management policy, you must have access to the administrative data utility, IXCMIAPU, also in SYS1.MIGLIB, and have UPDATE authorization to the RACF FACILITY class MVSADMIN.XCF.ARM. To start a policy, you must have READ authorization to the RACF FACILITY class MVSADMIN.XCF.ARM.

Though servers automatically register with automatic restart management, you must activate the arm component itself, which means you must:

1. Allocate an ARM couple data set
2. Start the automatic restart management policy

If automatic restart management is not active, WebSphere for z/OS issues an error message to the hardcopy log.

You are not required to change the automatic restart management policy. However, you will have to modify this policy if you want to create custom restart groups. For instance, it is not required or recommended that you start the node agent or deployment manager servers on another system. These servers will never have any transactional recovery to perform. Therefore, they should only be set up for restart-in-place. For complete information about how to modify the policies, see *z/OS MVS Setting Up a Sysplex, SA22-7625*.

The following procedure is intended to give you enough information to get automatic restart management running. Defining automatic restart management policies would require the z/OS manual mentioned above.

1. If you have not already formatted a couple data set for policies, do so now. For details, see *z/OS MVS Setting Up a Sysplex*
2. Submit the job to format the ARM couple data set.
3. Optional: Modify the automatic restart management policy. To get started, you do not need to modify the policy. If you do want to modify the automatic restart management policy, go to *z/OS MVS Setting Up a Sysplex*, and follow the instructions in that manual.
4. Issue the following operator commands to start the automatic restart management policy:

   ```
   SETXCF COUPLE,TYPE=ARM,PCOUPLE=(dsname,vvvvvv)
   SETXCF START,POLICY,TYPE=ARM
   ```

   where

dsname
Is the data set name for the couple data set.

vvvvvv
Is the volume serial of the volume on which the couple data set resides.

You are done when the SETXCF commands complete successfully.

Changing automatic restart management policies

Because server instances register with the default restart group, when a system failure occurs, automatic restart management attempts to restart the entire default group on another system in the sysplex. If you want to create a restart group other than this default group, you must follow the following rules and restrictions that apply for z/OS automatic restart management policies. For more information about how to actually change these policies, see, *z/OS MVS Setting Up a Sysplex, SA22-7625*.

- To change the policy, you need to know the existing element names for the server instances and how to name new elements for additional instances. The element names for these server instances are formed by concatenating the cell short name and the servers specific short name.

  For example, if you have a cell named PLEX1 and server named BBOS001, the ARM element name would be PLEX1BBOS001.

  Since wildcard characters can be used in the ARM policy, it is possible to exclude an entire group of servers by using a common naming scheme within your cell.

  For example, the following section of the ARM policy will prevent any node agents from starting, assuming each node agent in your cell has a name that adheres to the form BBONxxx:

  ```
  RESTART_GROUP(WEBSPHERE)
  ELEMENT(PLEX1BBON*)
  RESTART_ATTEMPTS(0,150)
  RESTART_TIMEOUT(600)
  READY_TIMEOUT(1200)
  TERMTYPE(ALLTERM)
  RESTART_METHOD(BOTH,PERSIST)
  ```

  This ARM policy will also prevent the node agent from restarting in-place. This specification can be modified by changing the RESTART_METHOD and TERMTYPE parameters. See z/OS MVS Setting Up a Sysplex, SA22-7625 for more information.
- If you create a restart group, keep the following in the same restart group and set the restart order for the elements as indicated:
  1. RRS
  2. DB2 with IRLM, IMS, CICS, and other transaction or resource managers, if used by your application servers in the restart group
  3. Your server instances.

IBM recommends that you should either set up the Daemon and node agent for restart-in-place or remove them from your ARM policy. Since WebSphere Application Server must be running on all systems that might be used to perform recovery, the application servers will use the Daemon and node agent that are already running on the alternate system. If the Daemon attempts to restart on the alternate system, it will fail. If the node agent restarts on the alternate system, it will have no recovery work to do.

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 for z/OS environment:

1. Initialize all servers.
2. Display all registered address spaces (including the address spaces of server instances). Issue the following command:

   ```
   d xcf,armstatus,detail
   ```

Tips for installing the embedded messaging feature

When installing IBM WebSphere Application Server for z/OS, you can install the embedded messaging feature for use as the Java message service (JMS) provider, which is also frequently called the integral JMS provider. The integral JMS provider is installed using the customization dialog. This process is documented in detail in the WebSphere Application Server for z/OS Installation and Customization.

Installing the embedded messaging server feature if WebSphere MQ Version 5.3 is already installed

You have a choice if you already have WebSphere MQ Version 5.3 installed:

- You can install only the embedded messaging client feature on a machine that already has WebSphere MQ Version 5.3.

  To use WebSphere MQ Version 5.3 as the JMS provider, install the IBM WebSphere Application Server product with only the embedded messaging client feature. Installing and using the WebSphere Application Server embedded messaging client feature is recommended with either the server feature or the full WebSphere MQ Version 5.3 product.

  WebSphere Application Server messaging applications can use the WebSphere MQ Version 5.3 product as the JMS provider. Using the client feature, however, requires that you install the WebSphere MQ Version 5.3 Java messaging feature.
- You can install the embedded messaging server and client features on a machine that already has WebSphere MQ Version 5.3.

  To install the embedded messaging server feature when WebSphere MQ Version 5.3 is already installed, upgrade WebSphere MQ Version 5.3:
  - Apply the CSD01 update to the original WebSphere MQ Version 5.3 release, or move to the WebSphere MQ Version 5.3 refresh release (which includes CSD01).
  - Install the WebSphere MQ Version 5.3 features, server and Java messaging, which the WebSphere Application Server embedded messaging server feature requires.

If you install WebSphere MQ Version 5.3 without the required features, the installation of either IBM WebSphere Application Server embedded messaging feature is unsuccessful because of prerequisite check errors. The WebSphere Application Server Enterprise package includes installation images of the WebSphere MQ Version 5.3 product and the WebSphere MQ Event Broker product, with restricted licensing. You can use the products to install the required WebSphere MQ Version 5.3 features or to install the refresh release of WebSphere MQ Version 5.3 for use with WebSphere Application Server Enterprise.

Avoiding a coexistence problem between embedded messaging, IBM WebSphere Studio Application Developer Integration Edition, and IBM WebSphere Application Server

The IBM WebSphere Studio Application Developer Integration Edition and IBM WebSphere Application Server both include an option to install embedded messaging. The embedded messaging option in these two products is incompatible.

To avoid this problem, do not install embedded messaging for both products on the same machine.

Migrating and coexisting

Determine whether you have an existing version of WebSphere Application Server installed on the machine where you plan to install your Version 5.x product.

If you have a previous version, you must plan whether to copy the configuration and applications of the previous version to the new version, which is *migration*. Migration does not uninstall the previous version. The earlier release is still functional. If you run it at the same time as the WebSphere Application Server for z/OS V5 installation, the two versions are *coexisting*. The migration articles will discuss migration from WebSphere Application Server for z/OS V3.5SE and WebSphere Application Server for z/OS V4.0.1 levels and coexisting with these prior levels. Unless otherwise noted, information pertaining to V3.5SE migration is valid for customers moving from the V4.0.1 Alternate Configuration Option.

If you have V4.0.1 installed, see Migrating from V4.0.1 to V5 for more information.

WebSphere Application Server provides support for migrating V4.0.1 applications to V5. This utility is available as V4.0.1 service (W401502) against 390fy [Direct Deployment Tool] and provides a new -v5mp option. V5 server configurations must be defined manually.

In overview, the migration tool perform a fairly routine migration from V4 to V5. For example, Java 2 Platform, Enterprise Edition (J2EE) 1.2 EAR files in V4 work in V5 of WebSphere Application Server, which also supports the J2EE 1.3 specification. Similarly, it is not necessary to redeploy EJB 1.1 JAR files when moving them from V4 to V5, which also supports EJB 2.0 JAR files.

The migration from V3.5 to V5 involves significant changes in application structures, development, and deployment. This migration is a manual effort.

If you have V3.5SE installed, see Migrating from V3.5SE to V5 for more information.

The remaining migration articles assume that V5 is being installed in an environment where it must coexist with prior levels of WebSphere Application Server. There are several items that should be considered during the planning phase to enable coexistence.

- Update prerequisites to the levels required by V5. Prior levels of WebSphere Application Server for z/OS will continue to run at the higher prereq levels. For more information, see “Prerequisites needed for WebSphere Application Server for z/OS V3.5SE, V4.0.1, and V5” on page 124.
- Set up V5 to eliminate potential LPA conflicts with a prior V4.0.1 install. V5 and V4.0.1 both require the placement of some code into LPA (SBBOLPA). In addition it is recommended that additional product code (SBBOLOAD) be

placed into LPA for performance reasons. Because of naming conflicts, V5 and V4.0.1 product code can not be in LPA at the same time. To support coexistence:
  – Place the V5 SBBOLPA dataset in the STEPLIB of the V5 daemon
  – Add either a V4.0.1 or V5 STEPLIB for SBBOLOAD.

  Note: In V5, put BBORTSS5 into LPA so ctrace will work. This does not cause any coexistence issues since the dll name is different than the V4.0.1 version.
- Utilize WLM DAE when configuring V5 to allow both the use of a specific server name by V5 and by a server on V4.0.1.
- Review the ports that have been defined to ensure that the V5 installation does not conflict. In particular, when installing to coexist with V4.01, note that the default daemon port defintion for both V5.0 and V4.0.1 is the same. For more information, see Coexistence port definitions. See ″z/OS port assignments″ in WebSphere Application Server Version 5 for z/OS Installation and Customization for more information on ports.

There are two other problems that might occur when you choose both migration and coexistence:
- One problem is conflicting context roots when attempting to share the same Web server.

  A procedure for configuring a Web server for sharing between WebSphere Application Server versions is described in the InfoCenter for the base WebSphere Application Server product.
- The other problem occurs if you install WebSphere Application Server Version 5 multiple times on the same z/OS image or sysplex.

You can coexist with, or migrate the applications and configuration from a previous version of WebSphere Application Server.

Migration from V4.0.1 to V5 does not require extensive tuning.

Migration from V3.5SE to V5 does require you to examine the migrating applications.

After planning for migration and coexistence, continue the installation by returning to Installing the product.

Coexistence support

Coexistence, as it applies to WebSphere Application Server products, is the ability of multiple installations of WebSphere Application Server to run on the same z/OS image or sysplex at the same time. Multiple installations include multiple versions and multiple instances of one version. Coexistence also implies various combinations of Web server interaction.

The Version 5 WebSphere Application Server products can coexist with supported, previous versions as described below.
- IBM WebSphere Application Server for z/OS Version 3.5SE
- IBM WebSphere Application Server for z/OS V4.0.1

In addition to coexistence with earlier versions of WebSphere Application Server for z/OS, WebSphere Application Server for z/OS also lets you install multiple times on one z/OS image or sysplex (multiple installation instances), or install once and have multiple configurations (multiple configuration instances).

Multiple Version 5 instances on one machine include:

- Multiple base Application Server instances from multiple installations of the Network Deployment product
- Multiple base Application Server configuration instances from a single installation of the Network Deployment product
- Multiple deployment manager instances from multiple installations of the Network Deployment product
- Multiple deployment manager configuration instances from a single installation of the Network Deployment product

Migrating from V4.0.1 to V5

Determine whether you have an existing version of WebSphere Application Server installed on the machine where you plan to install your Version 5 product.

To migrate V4.0.1 applications and configurations to Version 5, follow this procedure:

1. Prepare to migrate or update product prerequisites and corequisites to supported versions. If your existing configuration makes use of a non-390 Web server and associated AE plugin, you should review Web server levels required by V5 and update as appropriate. Refer to “Prerequisites needed for WebSphere Application Server for z/OS V3.5SE, V4.0.1, and V5” on page 124 for current requirements.

   Functional dependencies on the 390 infrastructure have changed in V5. See “Prerequisites needed for WebSphere Application Server for z/OS V3.5SE, V4.0.1, and V5” on page 124 comparing V3.5SE, V4.0.1 and V5.0 prerequisites.

   V4.0.1 must be at service level W401501 to allow coexistence with V5.
2. Review existing V4.0.1 for possible coexistence issues related to port definitions and placement of WebSphere Application Server modules in LPA. See “Migrating and coexisting” on page 106 for more information.
3. Install the V5 product. Note that V5 requires the use of SMP/E 3.1. Installation instructions have been provided with your ServerPac or PDO order.
4. Customize V5 utilizing the customization dialog provided with the product and verify success by running the Installation Verification Test. Note that the V5 default */install root* directory on z/OS is *WebSphere/V5R0M0/AppServer*, and it does not conflict with mount points or directory structure of the existing V4.0.1 release. See WebSphere Application Server for z/OS V5 Installation and Customization for detailed instructions.
5. Run the V5 samples. There have been significant changes to Admin, system structure, etc. with the V5 level. For this reason, it is recommended to run the V5 samples. See Chapter 2, “Samples Gallery,” on page 173 for more information.
6. Define V5 servers for applications being migrated.

   V5 now requires the use of the HTTP transport, and HTTP listening ports must be selected so as to not conflict with the existing V4.0.1 configuration.
   - There is no tooling support for migrating Server configurations because of the massive change in the Administration console support between V4.0.1 and V5. This process will need to be done manually. Use the V4.0.1 administrative console capability to to dump existing V4.0.1 settings for review. See “Utilizing WebSphere administrative capabilities to support migration from V4.0.1 to V5” on page 112 for more information.
   - Establish a naming convention that lends itself to changes and extensions when establishing Server configurations.

- Create V5 copies of policies such as WLM and ARM and modify them for the V5 Server configuration.

7. Install resources into the V5 server. Prerequisites levels needed for WebSphere for z/OS may have changed from V4.0.1. Review these prerequisites at "Prerequisites needed for WebSphere Application Server for z/OS V3.5SE, V4.0.1, and V5" on page 124.
8. Move applications to V5 level tooling. Most V4.0.1 applications will continue to run on V5. However, WSAD 4.0 and WSAD IE 4.1.1 generated applications are not formally supported for deployment into V5 runtimes. Formal support necessitates movement of the application to V5 level tooling. In the case of WSAD 5.0, this tool has the flexibility to generate both J2EE 1.2 and J2EE 1.3 applications allowing convergence to a single tool to support V4.0.1 and V5. In the case of WSAD IE 5.0, there are conflicts in tooling jars with WSAD IE 4.1.1. Customers will need to maintain both tooling levels until they have migrated everything to V5 levels. You can also find information about migrating to the V5 tooling levels in the IBM Redbook, Migrating to WebSphere V5.0: An End-to-End Migration Guide, SG24-6910-00. There is a link to the book in the Resources for Learning topic.
9. Review applications to be migrated to determine if any changes are required. The Class API checker tool can be used to analyze servlets and ejbs to determine if any APIs are unsupported or deprecated. The CACT tool can be found at WebSphere Developer Domain downloadable tools. If required changes are found, it will be necessary to modify them and then install into the V5 servers. If changes are not required, then the migration utility can be utilized. Some of the key things to note that may cause a need for application changes are:
   - The default code page has changed to be consistent with the V5 family. In V5, the JVM is started with a default file encoding of iso-8859-1 (US ASCII). This enables applications that rely on JVM behavior instead of explicit specification of encoding to execute unchanged on a WebSphere Application Server for z/OS V5 system. This should improves portability of applications across platforms.

     Even though the Java language allows programmers to explicitly specify the encoding of a file that is to be opened and read by the JVM, many programmers do not take advantage of this capability and rely on the default encoding. If you have applications which access EBCDIC encoded files, you must ensure that they explicitly specify this encoding in the application code, or you must change the encoding of the target file to match this new JVM setting.
   - With the installation of APAR PQ74162, JCA connector applications developed using WSAD IE 4.1.1 to create an atomic JCA connector Web service that communicates to a backend resource manager should still be able to run under V5. However, to do so, one must first redeploy the application to V5. The redeploy can be done during the installation of the application. Although, these applications should run under V5 after redeploying, this should be viewed as a temporary method of operation. Customers should still eventually transition their WSAD IE 4.1.1 JCA connector applications to the WSAD IE 5.0 tooling.
   - There are cases where Web Applications may need to be modified. See Migrating Web application components for more information.
10. Utilize the migration utility that was shipped in W401502 to process V4.0.1 EARs. This utility is being provided as an option on 390fy. In V4.0.1 it was required that EARs be processed through the 390 AAT or 390fy to condition them for deployment into V4.0.1. In V5, this conditioning is not required. This utility will remove the conditioning that was done for existing V4.0.1 EARs

that are being migrated to V5. It should be noted that if the application is being sent back through the WSAD 5.0 tooling, that the resulting EAR is ready for installation into V5 and that the use of migration utility is not required. Support for migrating V4.0.1 applications to V5 has been shipped in V4.0.1 service (W401502). This support has been provided as a new -v5mp option for the 390fy tool. See “Utilizing WebSphere administrative capabilities to support migration from V4.0.1 to V5” on page 112 for more information.

11. Address differences between V4.0.1 and V5 by use of the AAT.
    - Review the changes in the classloader options in V5 compared to V4.0.1. Determine the new option that is appropriate and utilize the AAT to make the change. See Migrating the class-loader Module Visibility Mode setting for more information.
    - A change was made in the web container behavior that was required by J2EE that involves the setting of content type. If a servlet writer does not set the content type it is no longer defaulted by the web container and it is returned as ″null″. This can cause some browsers to incorrectly display the resulting tags sent by the web container. This problem can be eliminated by using the AAT to set the autoResponseEncoding IBM extension to true for web modules while migrating enterprise applications.
12. Install the application into the V5 server. For Web applications being migrated from V4.0.1 the following should be noted. In V4.0.1, the classes generated from JSPs are in a package based on the directory structure of the WAR. Any JSP at the top of the context root is in the unnamed package, and JSPs in subdirectories of the root are in packages named after the subdirectories. In V5, the classes generated from JSPs are all in the package **org.apache.jsp**. Therefore, the classfiles are not compatible between versions. When migrating an enterprise application from V4.0.1 to V5, the JSPs need to be recompiled so that the classfiles are regenerated into the correct packages. Use the *-preCompileJSPs* option of wsadmin tool during the installation of the application to force this recompilation.
13. Migrate clients for the applications that have been migrated to the V5 Server.

    Update clients that reference EJBs.

    The naming structure has changed in V5. This makes the EJB references that were valid in previous versions no longer work in V5. Existing V4.0.1 clients will need to be changed appropriately. Alternatively, a name binding can be added to map the old name into the new name in V5. If you pick this alternative, you can find some helpful examples in the IBM Redbook, Migrating to WebSphere V5.0: An End-to-End Migration Guide, SG24-6910-00. There is a link to the book in the Resources for Learning topic.

    Update data resources for client enterprise applications.

    The client container supported by V4.0.1 utilized a WCCM model that is different from the one used by V5. As a result, data resources will need to be converted to run correctly with V5.
14. Update your Webserver to the V5 prerequisite level and the appropriate plugin. Most V4.0.1 configurations will include a Webserver. This can be a 390 Webserver or any Webserver supported by V4.0.1 AE. If a Webserver configuration is required for V5, then the Webserver to be used must be updated to the V5 prerequisite level and the appropriate V5 plugin must be installed. It should be noted that the V5 plugin can support both V4.0.1 and V5 application servers. With this in mind, several migration configurations are possible:
    - Install the V5 plugin into the existing Webserver (the V4.0.1 plugin and V5 plugin coexist).
    - Create a unique Webserver instance and install the V5 plugin.

- Replace the V4.0.1 plugin with the V5 plugin and setup to drive both the V4.0.1 and V5 images.

  See "Setting up Version 4.0.x and Version 5 coexistence" for more information.

15. Migrate the existing V4.0.1 security configuration for the applications that have been migrated. See Migrating security configurations from WebSphere Application Server Version 4.0.1 for more information.
16. Make the necessary operational changes to support production.
    - Scripting based on SM API is not valid at the V5 level. It must be converted. See Migrating from Systems Management Scripting API (SMAPI) V4 to wsadmin V5 for more information.
    - ARM policies should be updated based on the new topology/infrastructure.

    See the Operations and Administration Guide for more information.
17. Post migration activities.
    - Some interfaces have been deprecated at the V5 level. Although they are still supported at V5, they should be eliminated to facilitate migration to later WebSphere Application Server levels. See Migrating security configurations from WebSphere Application Server Version 4.0.1 for some areas that should be changed.
    - Migrate applications to the latest specification levels.

Migration from Version 4.0.1 to Version 5 does not require extensive tuning.

Migration from Version 3.5SE to Version 5 does require you to examine the migrating applications.

After planning for migration and coexistence, you are ready to continue the installation. Return to Installing the product to continue. Review WebSphere Application Server for z/OS V5 Installation and Customization for detailed installation directions.

Setting up Version 4.0.x and Version 5 coexistence:

You must migrate prerequisite and corequisite programs to the levels required by WebSphere Application Server, Version 5. You must also identify ports in use in Version 4.0.1 before you begin the Version 5 installation, to avoid possible conflicts during coexistence.

You can install WebSphere Application Server Version 4.0.1, with Service level W401502, and Version 5 on the same node.

By default, there are port conflicts between Version 4.0.1 and Version 5 that you must resolve. See Migrating from V4.0.1 to V5 for more information. The next step is to setup the Webserver to support V4.0.1 and V5 coexistence.
Associate a Web server with each WebSphere Application Server.

- Use a separate Web server for each WebSphere Application Server.
  1. Create a Web server instance using the Web server documentation.
  2. Install the appropriate Web Server plug-in feature, and identify the Web server configuration file location. (For example, identify the location of the `httpd.conf` file for IBM HTTP Server.)
- Use the same Web server for both WebSphere Application Server versions.

  To use the same Web server for both Application Server versions, you must first upgrade the Web server to the common level supported by both versions of the application server.

Follow this procedure to use the same Web server for both WebSphere Application Server versions.

1. Install the appropriate Web Server plug-in feature, and identify the Web server configuration file location. (For example, identify the location of the `httpd.conf` file for the IBM HTTP Server that is associated with Version 4.0.1.)
2. Edit the Web server configuration file to remove entries for Version 4.0.1, as described in the *Migrating plug-ins, one machine at a time* (tins_websmig2) topic in the base Application Server InfoCenter. The WebSphere Application Server, Version 5 plug-in acts as the routing agent to route requests to both versions.
3. Generate the plug-in configuration files for both versions of the Application Server.
4. Edit the Version 4.0.1 `plugin-cfg.xml` file and the Version 5 `plugin-cfg.xml` file, to merge their entries into a combined file of all Web context roots. Web context roots must be unique across Application Server versions. If there is a common context root across versions, requests are served by the plug-in instance that was last loaded.

   If you have both Version 4.0.1 Samples and Version 5 Samples in the file, you can access only the Version 5 Samples after merging the configuration files because there is one context root for Samples.

   This is described in the *Preparing to install and configure a Web server* (tins_webserver) topic in the base Application Server InfoCenter.

   Whenever you modify the `plugin-cfg.xml` file for either version, you must manually merge the files again, to make a master file.

   This is described in the *Preparing to install and configure a Web server* (tins_webserver) topic in the base Application Server InfoCenter.

   Whenever you modify the `plugin-cfg.xml` file for either version, you must manually merge the files again, to make a master file.
5. Replace the original `plugin-cfg.xml` file of the Version 5 installation with the master file.

Return to Migrating from, or coexisting with a previous version to continue.

Utilizing WebSphere administrative capabilities to support migration from V4.0.1 to V5:

There are a few steps in utilizing WebSphere Application Server V5 for z/OS administrative capabilities to support migration from V4.0.1 to V5:

1. Migrating your server
2. Migrating your applications
3. Migrating your server
4. Mapping of V4.0.1 environment variables to V5 WebSphere console settings

Migrating your server from WebSphere Application Server for z/OS V4.0.1 to V5:

Migrating your server from WebSphere Application Server for z/OS V4.0.1 to V5 is a manual process. The export server capability of the V4.0.1 adminstrative console allows the dumping of V4.0.1 server settings to an XML file for review prior to manually configuring the V5 server.

1. Use the ″Export server″ task within the admistrative console to specify <export_dir> as the Output directory within your HFS. You can take an ACTIVE conversation (one currently being used), and find the J2EE Server (for example, the Server name J2SERV1) that you want to EXPORT.

Then, you can specify <export_dir> (for example, in the Output directory) for output and export the J2EE server by clicking on OK.

Note: The directory ″/export_dir/J2SERV1″ **MUST** already exist in order for this export to succeed.

- ″/export_dir/J2SERV1″ should contain the *.xml file for each J2EE Resource that is used within the J2EE server that has been exported.
- ″/export_dir/J2SERV1″ should contain the server.xml file that contains the configuration settings of the exported J2EE server.
- ″/export_dir/J2SERV1″ should contain the <UUID> .ear file for each application that has been installed into the J2EE server being exported.

  You can look into server.xml to find out exactly which <UUID> .ear file maps to which application name.

2. Use <export_dir>/<server_name>/server.xml and transfer (or dump) the configuration.
3. Configure the V5 setup.
   a. Change the current configuration settings using the administrative console.

      Note: You should review the settings when you did a dump of the server.
   b. Save your SM configuration by one of two ways:
      1) Use the following administrative console command to commit the SM configuration:

         ```
         wsadmin> $AdminConfig  save
         ```
      2) Press the ″Save″ link in the administrative console to save your SM configuration into the master configuration.

      Here's an example of where it may be saved:

      /WebSphere/V5R0M0/AppServer/config/cells/SY1/...*.xml

Migrating your applications from WebSphere Application Server for z/OS V4.0.1 to V5:

Utilize the migration utility that was shipped in W401502 to process V4.0.1 EARs. This utility is being provided as an option on 390fy. In V4.0.1 it was required that EARs be processed through the 390 AAT or 390fy to condition them for deployment into V4.0.1. In V5, this conditioning is not required. This utility will remove the conditioning that was done for existing V4.0.1 EARs that are being migrated to V5. It should be noted that if the application is being sent back through the WSAD 5.0 tooling, that the resulting EAR is ready for installation into V5 and that the use of migration utility is not required. Support for migrating V4.0.1 applications to V5 has been shipped in V4.0.1 service (W401502). This support has been provided as a new -v5mp option for the 390fy tool.

The following are the steps needed to migrate your V4.0.1 application to V5:

1. Use the ″Export server″ task to specify <export_dir> within your HFS. See "Migrating your server from WebSphere Application Server for z/OS V4.0.1 to V5" on page 112 for more information on how to do this.
2. Change the <UUID> .ear name format to the regular name.

   Note: This step is optional.

3. Run the 390fy -v5mp <export_dir> /<server_name> option to migrate all applications that are exported into the <export_dir> /<server_name> directory. Each of the exported ear files will be in <UUID> format unless they have been changed in optional step 2 above.

   Your J2EE server (for example, J2SERV1) might contain the following installed applications:
   - J2APPL1
   - BVTEC
   - PolicyIVP

   When you export the J2EE server (J2SERV1 in this example) into the <export_dir>/<server_dir> (for example, ″/export_dir/J2SERV1″) directory, the directory will look like the following:

   ```
   /export_dir/J2SERV1/> ls -1
   <UUID_for_J2SERV1>.ear
   <UUID_for_BVTEC>.ear
   <UUID_for_PolicyIVP>.ear
   <UUID_for_some_DataSource1>.xml
   <UUID_for_some_DataSource2>.xml
   server.xml
   ```

   When you run the 390fy command in this directory it will generate a new *_v5mp.ear for each EAR file processed by the 390fy v5mp option.

   Note: The last parameter of the ″390fy″ command for the ″-v5mp″ option is a directory. In this case, it is the directory of the <export_dir>/<server_name> where the *.ear has been exported. For example:

   ```
   /export_dir/J2SERV1/> /usr/lpp/WebSphere/bin/390fy -v5mp .
   ```

   where ″**.**″ is the target directory where the *.ear files can be found, or

   ```
   /export_dir/J2SERV1/> /usr/lpp/WebSphere/bin/390fy -v5mp /export_dir/J2SERV1/
   ```

   where ″**/export_dir/J2SERV1/**″ is the target directory where the *.ear files can be found.

   The following demonstrates the 390fy v5mp option for this example:

   ```
   /export_dir/J2SERV1/> /usr/lpp/WebSphere/bin/390fy -v5mp .
   /export_dir/J2SERV1/> ls -1
   <UUID_for_J2SERV1>.ear
   <UUID_for_BVTEC>.ear
   <UUID_for_PolicyIVP>.ear
   <UUID_for_J2SERV1>_v5mp.ear
   <UUID_for_BVTEC>_v5mp.ear
   <UUID_for_PolicyIVP>_v5mp.ear
   <UUID_for_some_DataSource1>.xml
   <UUID_for_some_DataSource2>.xml
   server.xml
   ```

   The UUID for each application can be looked up in the server.xml file to find the original application name.

Mapping of V4.0.1 environment variables to V5 WebSphere console settings: The following table lists only V5 WebSphere variables related to diagnosis, along with their equivalent V4.0.1 environment variables. This information is provided only as an aid to IBM service personnel.

Warning: Do not use this information to manually modify the contents of a *was.env* file. The *was.env* file is managed by WebSphere, and its content is rewritten with each change made to the WebSphere configuration. Therefore, any hand-editing will be overwritten.

Table 42. V4.0.1 environment variables and their equivalent V5 WebSphere variables

V4.0.1 environment variables	Equivalent V5 WebSphere variables
BBOC_HTTP_INPUT_TIMEOUT	**protocol_http_timeout_input**
BBOC_HTTP_OUTPUT_TIMEOUT	**protocol_http_timeout_output**
BBOC_HTTP_PERSISTENT_SESSION_TIMEOUT	**protocol_http_timeout_persistentSession**
BBOC_HTTP_SSL_OUTPUT_TIMEOUT	**protocol_https_timeout_output**
CLIENT_TIMEOUT	**protocol_iiop_local_timeout**
IIOP_SERVER_SESSION_KEEPALIVE	**protocol_iiop_server_session_keepalive**
OTS_DEFAULT_TIMEOUT	**transaction_defaultTimeout**
	control_region_wlm_dispatch_timeout
OTS_MAXIMUM_TIMEOUT	**transaction_maximumTimeout**
RECOVERY_TIMEOUT	**transaction_recoveryTimeout**
SSLIIOP_SERVER_SESSION_KEEPALIVE	**protocol_iiop_server_session_keepalive_ssl**

Migrating from V3.5 Standard Edition (SE) to V5

Determine whether you have WebSphere Application Server for OS/390 V3.5 Standard Edition installed on the system where you plan to install your Version 5 product.

To migrate V3.5 Standard Edition (hereafter referred to as V3.5SE) applications and configurations to Version 5, follow this procedure:

1. Prepare to migrate or update product prerequisites and corequisites to supported versions. Significant functional dependencies on the 390 infrastructure have been added to V5 over those that had been required for V3.5SE. See “Prerequisites needed for WebSphere Application Server for z/OS V3.5SE, V4.0.1, and V5” on page 124 comparing V3.5SE, V4.0.1 and V5 prerequisites. Review the PSP bucket for the latest PTF information. There is no dependency on a particular V3.5SE service level to allow coexistence with V5.

   Unlike V3.5SE, V5 for z/OS allows distributed platform Web servers for distributed platforms to communicate with the WebSphere Application Server for z/OS product. Refer to IBM WebSphere Application Server supported hardware, software, and APIs for supported distributed platform Web servers.
2. Install the V5 product.

   Note: V5 requires the use of SMP/E 3.1. Installation instructions have been provided with your ServerPac or PDO order.
3. Customize V5 utilizing the customization dialog provided with the product and verify success by running the Installation Verification Test. The customization dialog will prompt you for various TCP/IP ports that the V5

Application Server should listen on for requests. You should insure that you configure the new application server not to make use of the same ports as an existing server on that same image.

Note: V 5 does not conflict with the mount points or directory structure of the existing V3.5SE release. See WebSphere Application Server Version 5 for z/OS Installation and Customization for more information.

4. Run the V5 samples. There have been significant changes to Admin, system structure, etc. with the V5 level. For this reason, it is recommended to run the V5 samples. See Chapter 2, “Samples Gallery,” on page 173 for more information.
5. Define V5 servers for applications being migrated.

   There is no tooling support for migrating Server configurations because of the massive differences in application packaging/deployment, server structure/architecture, application administration compared to V3.5SE. This process will need to be done manually. Some key differences that should be noted:
   - The servlet engine provided by V3.5SE has been replaced by a full J2EE 1.3 WebContainer provided by V5.
   - The WAS.CONF file has been eliminated. All configuration that was previously done through the WAS.CONF is now done through the J2EE tooling (WSAD, Application Assembly Tool, and the V5 adminstrative console). See “Migrating your existing was.conf file settings” on page 119 for more information.
   - The configuration is no longer restricted to running as a GWAPI out of the IBM HTTP Server for z/OS and OS/390. V5 has included HTTP Transport function as part of its server structure. A default internal transport is provided, but a specific HTTP transport can also be configured using the V5 adminstrative console. HTTP listening ports must be selected so as to not conflict with existing V3.5SE configuration. Because of the HTTP Transport function, customers now have three configuration options:
     a. Allow direct HTTP access to the J2EE server via the HTTP transport
     b. Utilize IBM HTTP Server V5.3 for z/OS and OS/390 with the WebSphere HTTP Plug-in for z/OS to front end the HTTP transport. This option requires new ServerInit, Service, and ServerTerm directives to be added to the HTTP server `httpd.conf` file. See ″The WebSphere for z/OS HTTP internal transport″ in WebSphere Application Server Version 5 for z/OS Installation and Customization for more information.
     c. Utilize any V5 supported distributed platform Web server, along with the appropriate WebSphere plug-in for Web servers, to front end the HTTP transport.
   - When establishing Server configuration, care should be taken to establish a naming convention that lends itself to changes and extensions.
   - Create WLM and ARM policies for the Servers as part of the server configuration. See WebSphere Application Server for z/OS V5 Installation and Customization for more information.
6. Install resources into the V5 server.

   Note: Prerequisites levels needed for WebSphere for z/OS may have changed from V3.5SE. Review these prerequisites at “Prerequisites needed for WebSphere Application Server for z/OS V3.5SE, V4.0.1, and V5” on page 124.
7. Create new DB2 tables if your application uses HTTP Session State Persistence The format of the DB2 tables needed to support HTTP Session State persistence has changed in V5. Consequently HTTP sessions can not be shared

between V3.5SE and V5. If your application uses HTTP Session State Persistence using DB2, you must create new tables. See Creating a DB2 table for session persistence for more information.

8. Review the existing security configuration for Web applications The authentication and authorization policy for URLs within J2EE Web applications is able to be fully described by the web.xml deployment descriptor that is contained in the Web application archive (.war) file that is deployed into the WebSphere Application Server for z/OS V5 J2EE Server. The enforcement of the rules at runtime is carried out by the Web container in the V5 J2EE Server in conjunction with the configured user registry, such as SAF or LDAP. Implementing a security policy for the installation consists of the deployer granting permission for users and groups to be in various ROLEs that are described in the deployment descriptor of the application. At runtime, the Web container will authenticate the user using the means in the deplyoment descriptor and will subsequently insure that either the authenticated user (or a group in which he is a member) has been granted access to the necessary J2EE Roles by the deployer. For more information on J2EE Security, please see Migrating security configurations from WebSphere Application Server Version 4.0.1.

   The V3.5SE server executes as a GWAPI plug-in within the address space of an IBM HTTP Server for z/OS. Security policy for URLs (including those that get serviced by the WASv3.5SE plugin) that are served from the Web server are configured using the security directives in the httpd.conf file of that Web server. At runtime, the Web server is able to authenticate the user to the configured security server such as RACF or LDAP and determine if the user or a group that the user is a member of has been granted access to the specified URL before allowing access.

   In order to migrate Web applications and their security policy from V3.5SE to V5 and execute that server independent of the existing Web server, you need to insure that the deployment descriptor for the Web applications that are being migrated contains security constraints that match what currently exists within the httpd.conf. It is recommended to make use of the deployment descriptors within the Web application as a means for describing the security policy of the application.

   V3.5SE does provide extended capability to supplement the Web server authorization processing by performing SAF checks against resources profiles that are defined in the SOMDOBJS facility class. If your existing V3.5SE servers use this capability, you will need to insure that you understand the policy and configure the equivalent within the deployment descriptor.

9. Move applications to V5 level tooling. WSAD 4.0 and WSAD IE 4.1.1 generated applications are not supported for deployment into V5 runtimes. This necessitates movement of the application to V5 level tooling. You can also find information about migrating to the V5 tooling levels in the IBM Redbook, Migrating to WebSphere V5: An End-to-End Migration Guide, SG24-6910-00. There is a link to the book in the Resources for Learning topic.

10. Review applications to be migrated to determine if any changes are required. See Migrating Web application components for more information. Some of the areas to review are:
    - **Update application to supported specification level.** V3.5SE provided a compatibility option allowing the use of Servlet 2.1/JSP 1.0 levels. Servlets/JSPs must be brought up to at least the minimal level that is supported by J2EE 1.3 (Servlet 2.2/JSP 1.1). The MigrateWC tool has been provided to assist in conversion of JSPs and Servlets coded at specification

levels of previous WebSphere versions. The MigrateWC tool can be found on the WebSphere Developer Domain downloadable tools Web site at http://www.ibm.com/software/.

- **Repackage application as a J2EE application.** V3.5SE did not require J2EE packaging. In V5, Web applications must be packaged as WAR/EAR files for deployment into a V5 webcontainer. J2EE packaging includes the declaration of application properties through deployment descriptors that are contained in a separate xml file. WSAD/WSAD IE tooling provides this packaging capability.
- **Review classloader options.** V3.5SE provided a single class loader and it was highly recommended that a web application only reference entities contained within the web application. Review the changes in the classloader options in V5 compared to V3.5SE and change as appropriate. See "Class loader options" on page 121 for more information.
- **Review servlet for change related to content type setting.** A change was made in the web container behavior (required by J2EE) that involves the setting of content type. If a servlet writer does not set the content type it is no longer defaulted by the web container and it is returned as "null". This can cause some browsers to incorrectly display the resulting tags sent by the web container. This problem can be eliminated by either modifying the servlet to set the content type, or by setting the autoResponseEncoding IBM extension to *true* for web modules.
- **Indicate external resources.** Web applications that use JDBC must include a deployment descriptor indicating that JDBC is an external resource that needs to be accessed.
- **Modify JNDI calls.** The com.ibm.ejs.ns.jndi.CNInitialContext Factory class can not be used for a V5 Web application to obtain an initial context factory for gaining access to the JNDI name space. Change the application to obtain JNDI access through the J2EE defined mechanism.
- **Remove any usage of Connection Manager APIs.** These APIs were deprecated in V3.5SE and are not supported in V5.
- **Review application resource authentication for possible changes.**
  - Existing application components that specify a user ID and password on the getConnection method must be reassembled with a resource authentication property of *Application* to maintain the current behavior of the application. Otherwise, the J2EE server ignores the user ID passed on the method.
  - Application code dependent on the ACEE being placed on the thread of execution will need to change. An example of this is the CTG best practice recommendation to not provide a password/ID on the getConnection call for applications running in V3.5SE. This type of application must be changed.
- **Convert to J2EE Connector Architecture.** Customers should recode, reassemble, and reinstall existing applications to work with WebSphere for z/OS-supported connectors that are designed to implement the J2EE Connector Architecture.
- **Make code page related changes.** The JVM default code page has changed from EBCDIC to ASCII to be consistent with the WebSphere Application Server V5 family. Application changes may be required. See "Code page changes" on page 121 for more information.

11. Install the application into the V5 server using the V5 adminstrative console.
12. Choose your Web server. Existing V3.5SE configurations include a V3.5 SE plug-in running as a GWAPI out of the IBM HTTP Server for z/OS and OS/390. V5 does not require the use of the IBM HTTP Server for z/OS and OS/390. You can choose to either:

- Use a distributed platform Web server, in which a V5 WebSphere plug-in for Web servers has been installed, or
- Run without a Web server fronting the HTTP transport provided as part of the V5 J2EE server

Either of these alternatives can coexist with the existing V3.5SE configuration. Alternatively, you may choose to continue to use an IBM HTTP Server for z/OS and OS/390, in which the V5 WebSphere HTTP Plug-in for z/OS has been installed. In this case, the IBM HTTP Server for z/OS and OS/390 you are using must be at a V5.3 level.

With this in mind, several migration configurations are possible:
- Install the V5 WebSphere HTTP Plug-in for z/OS into the existing IBM HTTP Server V5.3 for z/OS and OS/390 (V3.5SE plug-in and V5 WebSphere HTTP Plug-in for z/OS can coexist).
- Install the V5 WebSphere HTTP Plug-in for z/OS into a separate IBM HTTP Server V5.3 for z/OS and OS/390.

13. Make necessary operational changes to support production.
    - V5 provides scripting capability that can be used for automation.
    - ARM policies should be updated based on the new topology/infrastructure.

    See WebSphere Application Server for z/OS V5 Operations and Administration for more information.

Migration from Version 4.x to Version 5 does not require extensive tuning.

Migration from Version 3.5.x to Version 5 does require you to examine the migrating applications.

After planning for migration and coexistence, you are ready to continue the installation. Return to Installing the product to continue. Review WebSphere Application Server for z/OS V5 Installation and Customization for detailed installation directions.

Migrating your existing was.conf file settings:

The following table describes how to migrate your was.conf configuration file settings to your WebSphere Application Server V5 environment.

was.conf file setting	How to set in V5
appserver.compliance.mode	Not applicable.
appserver.version	Not applicable.
appserver.usesystemclasspath	Defined using administrative console J2EE server settings.
appserver.libpath	Defined using administrative console shared library settings.
appserver.classpath	Defined using administrative console runtime and application settings.
appserver.name	Defined using administrative console when application server is created.
appserver.jvmpropertiesfile	Not applicable. JVM properties are defined using the administrative console.
appserver.loglevel	Defined using administrative console trace and error control settings.

appserver.logdirectory	Not applicable. The name of the log directory is pre-defined.
appserver.jspbasehrefadd	Defined at the application level.
appserver.workingdirectory	Not applicable.
appserver.permisssions	Not applicable.
appserver.nodetach	Not applicable.
appserver.java.system.property	Not applicable.
appserver.java.extraparm	Not applicable.
appserver.configviewer	Not applicable.
appserver.initializeonwebappfailure	Not applicable.
session.enable	Not applicable.
session.invalidationtime	Defined using administrative console session management settings.
session.tableoverflowenable	Defined using administrative console session management settings.
session.tablesize	Not applicable.
session.dbenable	Defined using administrative console session management settings.
session.dbjdbcpoolname	Defined using administrative console data source settings.
session.dbtablename	Defined using administrative console Web container custom properties settings.
session.urlrewriting.enable	Defined using administrative console session tracking settings.
session.cookies.enable	Defined using administrative console session tracking settings.
session.protocolswitching.enable	Defined using administrative console session tracking settings.
session.cookie.name	Defined using administrative console session management cookie settings.
session.cookie.comment	Not applicable.
session.cookie.maxage	Defined using administrative console session management cookie settings.
session.cookie.path	Defined using administrative console session management cookie settings.
session.cookie.secure	Not applicable.
session.cookie.domain	Defined using administrative console session management cookie settings.
Connection pool properties	Not applicable.
Virtual host properties	Defined using administrative console virtual host settings.
Application definition (webapp properties)	Defined during the application assembly process.
Application deployment (deployedwebapp properties)	Defined using administrative console application install settings.
Java Virtual Machine (JVM) definition	Defined using administrative console Java Virtual Machine settings.

objectleveltrace.enabled	Not applicable.
objectleveltrace.host	Not applicable.
objectleveltrace.port	Not applicable.
inline.comment	Not applicable.

Class loader options:

WebSphere Application Server V3.5 Standard Edition for z/OS and OS/390 maintains an Application Server class loader that is global to the JVM. In addition each Web application has its own class loader.

The Application Server class loader serves as the parent to all of the Web application class loaders. The default search mode used when locating a class to load is "Parent First". In Parent First mode, the servlet engine first attempts to locate the specified class in the parent class loader or load it from one of the libraries or directories supported by that class loader. In the case where a Web application is being loaded, the servlet engine will first search the application level class loader for the target class. If not found, the servlet engine will then proceed to locate the class in the Web application specific class loader or from one of the libraries or directories specified in the deployedwebapp.*web_app_name* classpath directive for that Web application.

The application server classpath is used to specify classes that are used by multiple Web applications such as utility functions. The Web application contains the artifacts that are to be referenced by a single Web application. Web Applications do not have access to artifacts that are packaged with another Web application.

In V3.5, any text based files (properties or resource bundles) that are loaded from an open directory by the Application Level class loader are required to be encoded in EBCDIC. This anomoly no longer exists in V5. All property files and resource bundles are required to be encoded as US ASCII with no exception.

The technique for specifying a class loader policy is more robust in V5. Migrating the class loader Module Visibility Mode describes how to specify these policies in V5.

Code page changes:

WebSphere Application Server for z/OS V5 is much closer to the other WebSphere Application Server platforms in how it treats code page issues. This improvement should allow greater portability of applications from other platforms or from the development environment. Encoding changes should no longer be required as part of deployment.

If you are migrating from WebSphere Application Server Standard Edition V3.5, you should be aware of the following potential migration issues:

1. V3.5 supports JSP .91, 1.0, and 1.1 specification levels. JSPs that were configured as .91 or 1.0 levels required the JSP source files to be stored in EBCDIC encoding. JSP specification level 1.1 clarifies the requirements for content of JSPs. Minimally they must be stored in ISO-88590-1 (US ASCII) encoding or a proper derivative. To successfully migrate your level .91, and 1.0 JSPs to V5.0, you must convert the source files to ASCII encoding.
2. V3.5 requires property files and resources bundles that were loaded from an open directory by the application server class loader to be encoded as EBCDIC.

In V5, any resource bundles or property files that are used by your application that had previously been converted to EBCDIC encoding must be changed to ASCII before deployment.

Note: Properties or resource bundles that were contained in .jar files or that were loaded from a Web application class loader were not required to be in EBCDIC. Therefore they should already be in ASCII format.

3. V3.5 does not override the default native file encoding of the JVM. This implies that the default file encoding used by the JVM in V35 is assumed to be CP1047 - EBCDIC.

   Even though the Java language allows programmers to explicitly specify the encoding of a file that is to be opened and read by the JVM, many programmers do not take advantage of this capability and rely on the default encoding. V5 should improves portability of applications by starting the JVM with a default file encoding of ISO-8859-1 (US ASCII). This implies that applications that rely on JVM behavior instead of explicit specification of encoding will be able to execute unchanged on a WebSphere Application Server for z/OS V5 system. If you have applications which access EBCDIC encoded files, you must ensure that they explicitly specify this encoding in the application code, or you must change the encoding of the target file to match this new JVM setting.

   Note: If you have applications which access EBCDIC encoded files, you need to insure that they are explicitly specifying this encoding in the application code or change the encoding of the target file to match the new JVM setting.

Specification level summary of SE V3.5, V4.0, V4.0.1, and V5 WebSphere Application Server

This article shows the various releases of the WebSphere for z/OS family and the specification and functional differences between them.

Table 43. Specification and functional differences between WebSphere for z/OS releases

Specifications	WebSphere V3.5SE	WebSphere V4.0	WebSphere V4.0.1	WebSphere V5

Table 43. Specification and functional differences between WebSphere for z/OS releases (continued)

Java related	• Requires SDK 1.3 • Supports specification levels: – Servlet 2.1/2.2 – JSP .091/1.0/1.1 – JDBC 1.2	• Requires SDK 1.3 • Supports J2EE 1.2 levels: – Servlet 2.2 – JSP 1.1 – JDBC 2.0 – EJB 1.1 – JNDI 1.2 – JTA 1.0 – RMI/IIOP 1.0	• Requires SDK 1.3 • Supports J2EE 1.2 levels: – Servlet 2.2 – JSP 1.1 – JDBC 2.0 – EJB 1.1 – JNDI 1.2 – JTA 1.0 – RMI/IIOP 1.0 – JMS 1.1 – JavaMail 1.1 – JAP 1.0 – Client container	• Requires SDK 1.3 • Supports J2EE 1.3 levels: – Servlet 2.3 – JSP 1.2 – JDBC 2.0 – EJB 2.0 – part of J2SE 1.3 – JTA 1.0 – part of J2SE 1.3 – JMS 1.1 – JavaMail 1.2 – JAP 1.0 – Client container – Java Authentication and Authorization Service (JAAS) 1.0 – Java API for XML Parsing (JAXP) 1.1 – J2EE Connector Architecture 1.0
Web Services	N/A	N/A	Introduces Web Services equivalent to those provided by: • WebSphere AE V4.0 – HTTP to Stateless Session beans • Requires SDK 1.3 PTF • Supports specification levels: – SOAP 1.1 – Apache SOAP V2.2	• UDDI V2 • SOAP 2.3 • jaxrpc 1.0 tech preview • jsr109 V1.0 tech preview

- **WebSphere V3.5SE** adds support for Servlet/JSP specification levels required by the J2EE 1.2 specifications, allowing customers to take advantage of the latest levels and to start early migration of applications in preparation for movement to J2EE 1.2 servers.
- **WebSphere V4.0** introduces a J2EE server that provides support for enterprise applications consisting of Web applications and enterprise java beans (EJBs). Support for Servlets, JSPs, and EJBs is compliant to the J2EE 1.2 architecture and provides all the benefits of this architecture. Tooling supports the J2EE 1.2 required packaging scheme. The J2EE server, unlike prior JAVA runtimes provided by WAS SE, provides container managed services for items such as transactions and security, simplifying the application programmers job. In addition the J2EE server utilizes the z/OS infrastructure to provide QOS expected on the 390 platform.

- **WebSphere V4.0.1** finishes delivery of all functional items required for certification. This includes delivery of JMS, JAVAMail, and client container support. In addition, WebSphere 4.0.1 provides the initial delivery of Web Services.
- **WebSphere V5** provides support for next generation technologies- J2EE 1.3 compatible with support for key Web services. V5 is Network Deployment compliant and continues to build towards WebSphere family consistency in architecture, administrave console, and programming APIs.

Prerequisites needed for WebSphere Application Server for z/OS V3.5SE, V4.0.1, and V5

The following table describes the prerequisites needed for WebSphere Application Server for z/OS V3.5SE and V4.0.1 Alternate Configuration Option, V4.01, and V5.

Table 44. Prerequisites needed for WebSphere Application Server for z/OS V3.5SE and V4.0.1 Alternate Configuration Option, V4.01, and V5.

Pre-Reqs	V3.5 and V4.0.1 Alternate Configuration Option	V4.0.1	V5.0
OS/390 or z/OS	R8 and above	R8 and above	R10 and above z/OS1.2 is required for the Dynamic application environment
HTTP Server	IBM HTTP Server for OS/390 ONLY	Options: • Direct browser to HTTP transport • IBM HTTP Server for OS/390 with no HTTP transport • IBM HTTP Server for OS/390 with HTTP transport • Any HTTP server that supports AEplugin 4.0.2 and above.	Must use HTTP transport Options for connecting to HTTP transport: • Direct Browser • IBM HTTP Server for OS/390 (R10) • Any HTTP Server supported by V5.0 family plugins
Sysplex	Not required	Required	Required
OS/390 Communications server (TCP/IP)	Required	Required	Required
OS/390 Unix System Services and Hierarchical file system (HFS)	Required	Required Shared HFS is required for a multiple image sysplex	Required Shared HFS is supported, but is no longer required for a multiple image sysplex
SecureWay Security Server (RACF) or Equivalent	Required	Required	Required
System Logger	Not required	Required	Required
LightWeight Directory Access Protocol (LDAP) Server	Not required	Required	Not required

Table 44. Prerequisites needed for WebSphere Application Server for z/OS V3.5SE and V4.0.1 Alternate Configuration Option, V4.01, and V5. (continued)

Workload Manager in Goal mode	Only required for scaleable mode	Required	Required Dynamic AE (programmatic setup of WLM application environment) requires PTF against z/OS 1.2
Resource Recovery Services (RRS)	Not required	Required	Required
FTP server	Not required	Required	Not required
System SSL Security	Required for SSL	Required for SSL	Required for SSL
DB2	V5 or later if JDBC or Session State is used	V7 required for Admin DB	Not required or supported for SM function V7 required for user data (JDBC compliant driver)
WSMQ	Not required	Not required Optional support for point to point: MQSeries 5.2+PTF; SupportPac MA88 Optional support for pub/sub: MQ SI 2.1	Not required - Integrated JMS provided as part of WAS 5.0 If integrated provider is not required: • Can continue to use the same product stack as in WAS 4.0.1 **Note:** Functions such as XA support and Java client are only available with MQ5.3.1. • Can purchase and install full MQ5.3.1 product.
IMS JCA	Not applicable	IMS Connect 1.2+PTFs (requires IMS V7.1) IMS Connect 2.1 (requires IMS V8.1)	IMS Connect 2.1(requires IMS V8.1)
CICS JCA	Not applicable	CICS Transaction Gateway 4.0.2, 5.0, 5.0.1 (requires CICS TS 1.3)	CICS Transaction Gateway 5.0.1 (requires CICS TS 1.3)

Federating multiple Version 5 installation instances

There are two ways to federate a WebSphere Application Server node into a deployment manager cell:

- Using the deployment manager administrative console
- Using the **addNode** command line script from the `bin` directory of the node you are federating

Always use the `-includeapps` parameter of the **addNode** command unless you included the applications during a previous federation of the WebSphere Application Server node into the cell. See the description of the **addNode** command for more information about its parameters. The administrative console also provides an option for including applications on the base node into the cell.

Federating from the deployment manager administrative console requires a running WebSphere Application Server server.

A coexistence environment might have multiple WebSphere Application Server product installations on one machine. You can federate each installation into the same cell, or into different cells. Whenever the deployment manager federates a node into its cell, it configures the nodeagent server process for the node with a set of default ports. If the base node has the embedded messaging feature, the deployment manager also configures a WebSphere Application Server JMS provider, jmsserver, which is a server process with another set of default ports.

Plan to configure the node agent and the JMS provider with port assignments that differ from the defaults, to ensure there is no conflict with coexisting installation instances that have similar configurations.

Ensure that the fix level of the Version 5 WebSphere Application Server product matches or is lower than the fix level of the Network Deployment product.

1. Use the **stopNode** command to stop any running node agents that are using default ports on the same physical machine.

   Issue the command from the `bin` directory of the Application Server installation root.
2. Federate the Application Server node into the deployment manager cell, using the **addNode** command from the `bin` directory of the node to federate. This command automatically instantiates the `nodeagent` process, with default ports.

   For example, suppose you federate an installation instance that has a node name of ***MyNode***.
3. Log on to the deployment manager administrative console. Go to **System Administration > Node Agents > nodeagent > End Points**. Change port numbers for all end points and save the changes.
4. Log on to the deployment manager administrative console, if you have installed the embedded messaging feature on the node you federated:
   a. Go to **System Administration > Node Agents > jmsserver > Security Port Endpoints**. Change port numbers for all end points and save the changes.
   b. Go to **System Administration > Node Agents > jmsserver > End Points**. Change port numbers for all the end points and save the changes.
5. Synchronize the changes using **System Administration > Nodes**. Select the node, MyNode, and click **Synchronize**.
6. Stop the node agent on the Application Server node using the **stopNode** command line script. Restart the node agent using the **startNode** command.
7. Change port assignments in the configuration files if you have an Application Server or deployment manager node that you cannot start because of port conflicts.

To continue, return to one of these topics:
- Establishing multimachine environments
- Installing multiple Version 5 configuration instances
- Setting up Version 5 coexistence

Coexistence port definitions

The following table defines port definitions for WebSphere Application Server for z/OS versions.

See "z/OS port assignments" in Port number settings in WebSphere Application Server versions for more information on ports.

Table 45. Coexistence port definitions

WebSphere Application Server for z/OS V5	**WebSphere Application Server for z/OS V4.0.1**	**WebSphere Application Server for z/OS V3.5SE**
Daemon 5555 5655 5656 (SSL)	Daemon 5555	
Application server	Application server	Web server
IIOP Dynamically chosen Equal to bootstrap port	IIOP Dynamically chosen	
Bootstrap 2809 if base server 9810 if part of ND cell	Application Server did not have a bootstrap port	
IIOPS Dynamically chosen	IIOPS Dynamically chosen	
HTTP 9080 (same as distributed)	HTTP No Default (8080 in examples)	HTTP 80
HTTPS 9443 (same as distributed)	HTTPS No Default (8080 in examples)	HTTPS 443
NodeAgent	SM	
Bootstrap 2089 (same as distributed)	Bootstrap 900	
IIOP 2089 (separate in distributed)		
IIOPS Dynamically chosen		
LDAP Not Applicable	LDAP 1439	
Naming/IR Not Applicable	Naming/IR Dynamically chosen	

Installing service to WebSphere Application Server for z/OS

Maintaining the base z/OS environment in addition to the WebSphere Application Server can require an in-depth knowledge of applying service and the product service stream. This is especially important in a high yield production environment where availability is of the utmost importance. Therefore planning service and knowing how to best apply service are essential skills.

Contact the IBM Software Support Center for information about preventive service planning (PSP) upgrades for WebSphere Application Server for z/OS. For more information about PSP upgrades, see *WebSphere Application Server for z/OS V5.0: Program Directory*. Although the *Program Directory* contains a list of required program temporary fixes (PTFs), the most current information is available from the IBM Software Support Center.

Make copies of the product data sets and HFS structure, and apply maintenance to the copies when applying service to WebSphere Application Server for z/OS. Use the following procedure to put the maintenance into production.

1. Stop the Application Servers and the WebSphere Application Server for z/OS location service daemon.
2. Switch to the newly-serviced WebSphere Application Server for z/OS product data sets. You can do this by:

    a. Rename the new data sets to replace the old ones.
    b. Recatalog product data sets, if the names are identical, or change WebSphere Application Server for z/OS cataloged procedures to refer explicitly to the new data sets.

   Verify that the MVS link list and APF list refer to the newly-serviced data sets.
3. If the WebSphere Application Server for z/OS run-time is loaded into the link pack area, delete the old modules and load the new ones, or IPL the system to load the new modules into the LPA.
4. Verify that the newly-serviced HFS data sets are correctly mounted.
5. Perform any other migration actions as instructed in PTF or APAR cover letters.
6. Start the location service daemon and application servers.
7. Complete any necessary post-install tasks.

You can maintain service to clients when upgrading the host cluster of WebSphere Application Server for z/OS.

Applying or restoring off a service level

Following the delivery of WebSphere for z/OS service level W500104, there are additional actions that you need to perform when using SMP/E to apply a new service level or restore off a service level. These new WebSphere for z/OS add-ons, which you perform on the WebSphere run-time home directory created in the HFS by the Customization Dialog, were put into place to reduce some of the post-install service actions you previously had to perform by hand. It also makes possible tasks you couldn't perform before, such as updating customized files (such as setupCmdLine.sh or server.xml, which contain unique content for each node) that exist in each WebSphere run-time home directory, or expecting new files to appear in the WebSphere run-time home directory after you add them to the SMP/E home directory.

Post-install tasks for after applying a new service level

This topic describes post-install tasks you complete after applying a new service level.

The new post-install functionality includes actions you perform the first time the server is restarted after you install service with SMP/E. It includes a new step, which automatically launches the ″applyPTF.sh″ shell script, in the server proc. applyPTF.sh, which is located in your bin directory ($WAS_HOME/bin, where $WAS_HOME is the absolute path of the WebSphere run-time home directory), makes sure that any pending post-install actions are properly applied before starting the server.

You don't have to perform any new manual steps when using SMP/E to apply service unless there is something in your configuration that prevents the applyPTF.sh shell script from running properly or if you are applying a particular service release that requires manual intervention.

Since post-install processing executes under the controller proc, it executes with the identity assigned to that particular proc. While that identity has sufficient authority to perform MOST HFS file actions required on the WebSphere run-time home directory, you cannot assume it to have the appropriate authority for your applyPTF.sh shell script. Therefore, you cannot perform any post- install action that requires special authority by ″inline″ apply processing. You must manually perform any such apply processing in ″batch″ mode, running applyPTF.sh or the appropriate jobs from a user that has the needed authority to apply a particular action.

Note:

- When you launch the shell script from the server proc, it automatically runs in ″inline″ mode and can only apply service that qualifies as ″inline.″
- If batch service is delivered, you'll find the needed user authority noted in the service level documentation.

The post installer component enforces that certain actions are performed successfully and preconditions are met before applying service. Otherwise, warnings or error messages result. If you launched the applyPTF.sh script from the server proc, the output is appended to the file $WAS_HOME/properties/service/logs/ applyPTF.out. If you ran applyPTF.sh manually from the shell, the output goes right to stdout (on the shell from where the applyPTF.sh script was run) and WTO messages are issued. In the latter case, no output is appended to the applyPTF.log log file.

Whenever actions are performed on the WebSphere run-time home directory, a log file is kept. (This does not apply if you ran applyPTF.sh manually from the shell, as indicated above.) When warnings or errors occur, the absolute path to the log file is displayed so that you can examine the details of the problem. The following is a list of common errors that might occur when the post installer (applyPTF.sh) applies service. The errors will most likely appear in the form of error codes on the console on which applyPTF.sh was issued.

The post installer ran and determined that the SMP/E home directory was SMP/E rolled back in service to an earlier version than that at which the WebSphere run-time home directory is running.

This occurs if the SMP/E restore off was performed without first running the backoutPTF.sh shell script. The server will not start if it detects this condition, and post-install action halts without applying service.

The post installer issued a warning message while installing service.

Warnings are still a success, but you should still look at the log file. A console message is issued for each warning hit during apply processing, but the server will still start.

Note: These warning messages appear only the first time you run the post installer as it does not retry to apply service for defects that issue warnings. It is important to check the log file in all cases because this could indicate that a certain function is disabled.

The post installer incurred an error and stopped processing.

When this happens, make sure you review the log and correct the error. The server will not start until it is able to successfully complete the post-install action on which it is failing.

You tried to start the server, but the post installer came in and found ″batch″ service pending.

The post installer checks batch service before starting anything. It issues such a message and exits without doing any of the pending service if it finds any.

Note: If the post installer detects this condition, it won't allow the server to start.

To start the post installer, you can run applyPTF.sh OR you can submit specific jobs depending on the intended node. So, perform ONE of the tasks below.

1. Run applyPTF.sh in batch mode by issuing ″run applyPTF.sh″ in the bin directory of each existing node OR
2. Submit BBOWAPLB for a base Application Server node, or BBOWAPLD for a Deployment Manager node.

Post-install tasks for after restoring off a service level

This topic describes post-install tasks you complete after restoring off of a new service level.

When restoring off service, you need to ″back off″ any post- install actions applied during that service. Follow these steps to run the ″backoutPTF.sh″ shell script, which is located in your bin directory ($WAS_HOME/bin, where $WAS_HOME is the absolute path of the WebSphere run-time home directory) and will take care of this for you.

The following is a list of common errors that might occur when you use backoutPTF.sh to back out of service. The output goes right to stdout (on the shell from where the applyPTF.sh script was run).

A valid, currently installed PTF or APAR name must be specified (case sensitive).
If a PTF or APAR isn’t specified, then the ″usage″ of the command is printed along with a listing of the currently installed and pending service units.

Level check failed message.
This indicates that the WebSphere run-time home directory is at a higher service level than the SMP/E home directory, which can happen if you SMP/E restore off a PTF or APAR level without first running the backoutPTF.sh script. You will see a message like this: ″Error: The WebSphere run-time home directory is at a higher service level than the SMP/E home directory. You must use SMP/E to return to level <service level that was just SMP/E restored off> and use the backoutPTF.sh to return to level <current service level>.″

Service pending warning
This means you are running the backoutPTF.sh script but still have service pending. This error could happen if:

- You installed a new service level with SMP/E but haven’t started the server or explicitly run applyPTF.sh yet.
- You installed a new service level with SMP/E but the post- install of the new level failed. So, you decided to roll back service, creating new pending service in the process.
- You ran the backoutPTF.sh script more than once without SMP/E backing off a PTF level.

1. Bring down those servers in the node for which you are doing backoff processing.
2. Launch the backoutPTF.sh script, specifying the committed service level to which you intend SMP/E to revert (″restore off″).

   Example: This is an example of the command used to run the backoutPTF.sh shell script, run from the shell by a user with the proper authority. It prepares the WebSphere run-time home directory to run at service level ″W500104.″

   ```
   backoutPTF.sh W500104
   ```

   Note:

- The shell script uses service levels of the form ″Wvrmxyy″ (PTF), ″AQxxxxx″ (++APAR), or UQxxxx (SMP/E service).
- The shell script is case sensitive.

3. After using the backoutPTF.sh shell script, follow the normal SMP/E procedures for restoring off a service level.

You know you are done when you are successfully back on the desired service level.

Using the installation verification test

After installing the base product during an umbrella installation, you are ready to use the installation verification test (IVT).

After performing the SMP/E code install and customizing the WebSphere Application Server functions, you are ready to use the installation verification test (IVT). There are two ways you can opt to invoke the IVT, which verifies that WebSphere Application Server is configured correctly for your system.

Note: These options are only available if the user is running a base Application Server configuration.

If you want to run the Installation Verification Test at a time other than during initial customization, you can follow either of two methods:

1. Select the method of invoking the IVT.
2. Optional: Use the BBOWIVT job
3. Optional: Run `ivt.sh` from a command line

Return to Installing WebSphere Application Server to continue.

Running the Installation Verification Test with a job

The Application Server must be running.

1. Verify the server is running.
2. Confirm the ivtApp application is installed and started.
3. Submit the job BBOWIVT.

The IVT runs a series of verification tests and reports pass or fail status in the messages generated by the BBOWIVT job. It also logs results to the `install_root\logs\ivt.log` file. The IVT will end unsuccessfully if the server is not running.

Running the Installation Verification Test from a command line

The Application Server must be running.

1. Verify that the application server is running. 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/V5R0M0/AppServer/bin` directory.
4. Issue the following execution: `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/V5R0M0/AppServer/bin> ivt.sh -p 9090 -host myhost
```

You are done when the Installation Verification Test runs successfully.

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.

Migrating XML applications

The following step is needed to migrate Version 3.5SE XML applications to supported XML APIs.
Migrate Version 3.5.x XML applications to supported XML APIs

If your XML applications use XML for Java API, Version 2.0.x or earlier, you must migrate them to API Version 3.1 or the equivalent open-source version. Although there are inherent performance improvements in later versions of the XML for Java API, you can gain additional performance by explicitly using nonvalidating parsers in application environments where you can trust the data.

The most significant change is that the TX-compatible APIs are no longer available. The Document API retains the XML manipulation APIs that were in TXDocument, but you must rewrite the following functionality:

- Creating and loading an XML parser: Use a Java API for XML Processing (JAXP) factory class.
- Writing out the Document Object Model (DOM) tree: Use a serializer. One drawback to the DOM Level 2 implementation in this level of the XML for Java API is that the grammar (DTD or schema) is no longer a node in the DOM tree, so you cannot write it out. As a result, only external grammars are recommended. You can query the system ID of the root element and use it to retrieve the name from the statement. After the tree is written to an XML file, you can read the file as text and insert a statement.

Note: In addition to the XML API changes, it is important to understand that J2EE Version 1.3 mandates the use of JAXP 1.1, DOM Version 2, and SAX Version 2. JAXP Version 1.2, DOM Version 3, and SAX Version 3 are not allowed in products that are compliant with the J2EE Version 1.3 specification. This prohibition exists because the newer versions were *experimental* at the time of the J2EE Version 1.3 specification. Because WebSphere Application Server is compliant with the J2EE Version 1.3 specification, WebSphere Application Server has support for JAXP Version 1.1, DOM Version 2 and SAX Version 2 only.

You must only recompile a Version 4.0.1 XML application to migrate it to the Version 5 level.

Creating multiple Version 5 configuration instances

Multiple configuration instances, as it applies to WebSphere Application Server products, is the ability to install the base WebSphere Application Server product, the Network Deployment product, or the WebSphere Application Server for z/OS product once, and to use the **wsinstance** command to create multiple ***instances of the initial installation***, all running on the same machine at the same time.

In contrast to multiple configuration instances is *coexistence*, which refers to multiple *installations* of WebSphere Application Server, running on the same machine at the same time. Both coexistence and multiple configuration instances of the base WebSphere Application Server product imply various combinations of Web server interaction.

You can find a description of coexistence in Coexistence support.

You can install the base WebSphere Application Server product, the Network Deployment product, or the WebSphere Application Server for z/OS product one time and use the wsinstance tool to create multiple configuration instances. Each WebSphere Application Server configuration instance is a stand-alone instance with a unique name, and its own set of configuration files and user data folders. Configuration folders include `config`, `etc` and `properties`. User data folders include `installedApps`, `installableApps`, `temp`, `logs`, `tranlog` and `wstemp`. Each instance has the administrative console application to manage the configuration instance. The configuration instance shares all run-time scripts, libraries, the Software Development Kit, and other files with the initial Application Server.

Each configuration instance of the Network Deployment product and the WebSphere Application Server for z/OS product is a stand-alone deployment manager (dmgr) with its own set of unique configuration files and user data folders for the cell. Configuration folders include `config`, `etc` and `properties`. User data folders include `installedApps`, `installableApps`, `temp`, `logs`, `tranlog` and `wstemp`. The dmgr configuration instance also includes the administrative console and the file transfer applications to manage the configuration instance.

You can configure and operate each configuration instance independently of other instances.

These limitations apply to multiple configuration instances:

- You cannot federate a configuration instance of the base product into a deployment manager cell. A deployment manager cannot manage a configuration instance.
- You cannot use the **Uninstall.exe** or **uninstall** command (or the **uninstall.sh** command on Linux for S/390 platforms) to remove a configuration instance. You must remove configuration instances with the -delete option of the **wsinstance** command, before uninstalling the installation instance.
- You cannot migrate the configuration and applications from an earlier release to a Version 5 configuration instance.

Reasons to use configuration instances include:

- You install the product only once.
- You can have stand-alone environments when installing the full product multiple times is not an acceptable solution because of space or procedural constraints. (Configuration instances use a minimal amount of space.)

- You can have a centralized development environment that is easy to maintain, where multiple developers and testers share one machine.
- You can create identical environments, where everyone is developing, testing, or performing in an identical workspace.

One reason to not use configuration instances is that you cannot federate a configuration instance of the base product into a deployment manager cell. A deployment manager cannot manage a configuration instance.

Use the following procedure to create and configure multiple V5 instances:

1. Install the WebSphere Application Server for z/OS product.

   Use the WebSphere Application Server for z/OS Customization Dialogs to create and customize the base Application Server, the integral JMS provider, and the Network Deployment function. The Customization Dialog process is documented in detail in WebSphere Application Server for z/OS V5 Installation and Customization
2. Use the **wsinstance** command in the `*WAS_HOME*\bin\wsinstance` folder, to create a configuration instance of the server you installed in step 1.

   Refer to the description of the **wsinstance** command to learn more about the command, and to see examples of use.

   You must specify a unique directory path and a unique node name for each configuration instance.

   You must also specify unique port numbers for each configuration instance. For example, on a Windows platform, specify ports beginning at `20002`, for node shasti, in configuration instance_root `G:\shasti\WebSphere`, on the planetjava machine, by issuing this command:

   ```
   wsinstance.bat -name shasti
                  -path G:\shasti\WebSphere
                  -host planetjava
                  -startingPort 20002
                  -create
   ```

   This command creates a separate set of configuration and other data files.
3. Run the **setupCmdLine.bat** (or **setupCmdLine.sh**) script in the `bin` directory of the *`instance_root`* folder to set the WebSphere Application Server environment to the configuration instance.

   Or, you can set WAS_USER_SCRIPT to `instance_root\bin\setupCmdLine.bat`, which has the same effect as running the **setupCmdLine** command.

   On Linux and UNIX-based platforms, ***source*** the script to the parent shell to inherit the exported variables by running this command:

   ```
   # . ./setupCmdLine.sh
   ```

   Or, you can set WAS_USER_SCRIPT to `instance_root/bin/setupCmdLine.sh`, which has the same effect as running the **setupCmdLine** command. For example:

   ```
   # export WAS_USER_SCRIPT=/opt/inst1/bin/setupCmdLine.sh
   ```

   After completing this step, you can start the Application Server configuration instance with the **startManager** or **startServer** command.

   Note: If you are using the embedded messaging feature, and you are logging on as a non-root user on a UNIX-based or Linux platform, you must add the user to the mqm and mqbrkrs operating system groups, as described in the Installing WebSphere embedded messaging as the JMS provider topic.

4. Create additional servers in a multiple instance or coexistence environment.
5. Change port assignments in configuration files if you have a node that you cannot start because of port conflicts.

Refer to the description of the **wsinstance** command to learn more about the command, and to see examples of use.

Return to Installing WebSphere Application Server to continue.

wsinstance command

The **wsinstance.bat** (or **wsinstance.sh** for Linux and UNIX-based platforms) is a command line tool that creates multiple configuration instances of one initial installation of the product.

Overview

The **wsinstance.bat** (or **wsinstance.sh** for UNIX-based platforms) command line tool creates multiple configuration instances of one initial installation of either the base WebSphere Application Server or the deployment manager.

Each WebSphere Application Server configuration instance is a stand-alone instance with a unique name, and its own set of configuration files and user data folders. Configuration folders include `config`, `etc` and `properties`. User data folders include `installedApps`, `installableApps`, `temp`, `logs`, `tranlog` and `wstemp`. It also has the administrative console application to manage the configuration instance. It shares all run-time scripts, libraries, the Software Development Kit, and other files with the initial application server.

Each deployment manager configuration instance is a different cell. You can federate application server installation instances into a deployment manager configuration instance.

Each configuration instance of the Network Deployment product is a stand-alone deployment manager (dmgr) with its own set of unique configuration files and user data folders for the cell. Configuration folders include `config`, `etc` and `properties`. User data folders include `installedApps`, `installableApps`, `temp`, `logs`, `tranlog` and `wstemp`. The dmgr configuration instance also includes the administrative console and the file transfer applications, to manage the configuration instance.

The **wsinstance.bat** command file is located in the `wsinstance` subdirectory of the `bin` directory in the *`install_root`* of the product.

Windows platforms:

```
wsinstance.bat  -name instanceName
                -path instanceLocation
                -host hostName
               [-startingPort startingPort]
                -readOnly
                -portDefs pathname
                -create|-delete
                -debug
```

Linux or UNIX-based platforms:

```
wsinstance.sh  -name instanceName
               -path instanceLocation
               -host hostName
              [-startingPort startingPort]
              [-readOnly]
              [-portDefs pathname]
               -create|-delete
              [-debug]
```

Parameters

Supported arguments include:

-create|-delete
Specifies whether to create or delete the configuration instance.

-debug
Optional parameter. Turns on the debug function of the Ant utility, which the wsinstance tool uses.

-host
Specifies the hostname, which is the name of the host on which you are creating the configuration instance. This should match the host name you specified during installation of the initial product.

-name
Specifies the instance name of the new configuration instance. Ensure this value is unique. The **wsinstance** command uses this name to construct the node name, which is *instanceName_hostname* for the new configuration instance of a deployment manager. The instance name is also the name of the cell. It is important to have a unique name for each configuration instance, to provide a unique combination of hostname and instance name, which together form the WAS_NODE name for the configuration instance.

-path
Specifies the file path of the instance. All required folders for the instance are in this directory, which is unique to the configuration instance.

Optional parameter. When specified, the **wsinstance** command looks in the path for a file that defines port settings for the new instance. When omitted, the **wsinstance** command looks in the installation root directory of the original product, in the `/bin` directory.

Optional parameter. The **wsinstance** command can detect whether you are creating an instance from the base WebSphere Application Server or Network Deployment product in a read-only partition.

The wsinstance command registers a new wsinstance in the `properties/wsinstance.properties` file of the original installation, if write access to the original product is available. If the original installation is in a read-only partition or if you use the -readOnly option, the **wsinstance** command does not register the wsinstance within the `properties/wsinstance.properties` file of the original installation.

Creating configuration instances from an original installation in a read-only partition has service implications. There is no mechanism in the original installation to identify and update a configuration instance that you create from a read-only partition. Normally, the registration file in the original installation provides the mechanism for updating configuration instances. When you apply a fix pack or an interim fix to the original installation, the update installer also updates the registered configuration instances. However, there is no registration entry for a configuration instance created from a read-only partition, where no entry is made in the registration file.

When you install a fix pack or interim fix to the original installation, you must use the **wsinstance** command to delete the configuration instance and recreate it. A configuration instance must be at the same fix level as the product from which you create it. Otherwise, the configuration instance is unsupported.

-startingPort
Optional parameter. Specifies the starting port number for generating all ports for the configuration instance. If not specified, the **wsinstance** command uses default ports, or custom-defined ports from a file that you can create.

Example of creating an instance on a Windows platform

To create a configuration instance for user `shasti`:

```
wsinstance.bat -name shasti
               -path G:\shasti\WebSphere
               -host planetnt
               -create
```

On a base WebSphere Application Server product installation, the command creates an application server configuration instance named `shasti` in location `G:\shasti\WebSphere` with a node name of `shasti_planetnt`.

On a Network Deployment product installation, the command creates a deployment manager configuration instance named `shasti` in a cell named `shasti` in location `G:\shasti\WebSphere` with a node name of `shasti_planetnt`.

Example of creating an instance on a Linux or UNIX-based platform

To create a configuration instance for user `shasti`:

```
wsinstance.sh -name shasti
              -path ~/shasti/WebSphere
              -host planetaix
              -create
```

On a base WebSphere Application Server product installation, the command creates an application server configuration instance named `shasti` in location `~/shasti/WebSphere` with a node name of `shasti_planetaix`. If user shasti is to use the embedded messaging feature, you must add the user (shasti) to the mqm and mqbrkrs operating system groups, as described in Installing WebSphere embedded messaging as the JMS provider. This is true for any non-root user on a Linux or UNIX-based platform.

On a Network Deployment product installation, the command creates a deployment manager configuration instance named `shasti` in a cell named `shasti` in location `~/shasti/WebSphere` with a node name of `shasti_planetaix`.

Example of a creating an instance in a multiuser environment on a Linux or UNIX platform

Example of a creating a configuration instance for user `shasti` in a multiuser environment:

1. Create the configuration instance:

```
>wsinstance.sh -name shasti
             -path ~shasti/WebSphere
             -host myhost
             -startingPort 12000
             -create
```

2. Change the owner of the folder:

```
>chown shasti ~shasti/WebSphere
```

3. As a convenience, add a call to script `~shasti/WebSphere/bin/setupCmdLine.sh` in the profile of user `shasti` to set the environment when user shasti logs in.

4. Give these folder permissions to user `shasti`:

```
install_root/bin         --- rx (read and execute)
install_root/java        --- rx
install_root/properties  ----r  (read)
install_root/deploytool  ----r
install_root/config      ----r
install_root/lib         ----r
install_root/classes     ----r
install_root/null        ----r
install_root/samples     ----r
install_root/Web         ----r
```

Example of deleting a configuration instance

The following command deletes the configuration instance named shasti:

```
wsinstance.sh -name shasti
               -host planetaix
               -delete
```

Using predefined port numbers

When you use the **wsinstance** command without the -startingPort parameter, the tool attempts to examine a file at the location specified on the command line (-portdefs), or named with the convention *instanceName*_portdef.props in the /bin/wsinstance directory of the installation root directory of the original product. You can create the file by copying the portdef.props template file and customizing the port numbers.

Example of using default port assignments

When a port definition file is not present, the **wsinstance** command generates a *hostname_instanceName*_portdef.props file in the /bin directory of the installation root directory of the original product, using the portdef.props template file.

The **wsinstance** command adds one (1) to the default port numbers for the base WebSphere Application Server product. If you create two configuration instances using default port assignments, both instances have the same, conflicting port numbers.

Incrementing default port numbers from a starting point

The **wsinstance** command can assign port numbers based on a starting port value that you give on the command line, using the -startingPort parameter. The tool assigns port numbers sequentially from the starting port number value as it creates an *hostname_instanceName*_portdef.props file in the wsinstance subdirectory of the bin directory in the *install_root* folder.

For example, an *hostname_instanceName*_portdef.props file created with -startingPort 20002 would appear similar to this:

```
HTTPS_TRANSPORT_ADMIN=20002
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS=20004
HTTP_TRANSPORT_ADMIN=20003
HTTP_TRANSPORT=20005
HTTPS_TRANSPORT=20006
INTERNAL_JMS_SERVER=20007
BOOTSTRAP_ADDRESS=20008
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS=20009
DRS_CLIENT_ADDRESS=20011
SOAP_CONNECTOR_ADDRESS=20010
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS=20012
JMSSERVER_QUEUED_ADDRESS=20013
JMSSERVER_DIRECT_ADDRESS=20014
```

The order of port assignments can be arbitrary. Predicting assignments is not possible.

Example of startingPort parameter use

The following **wsinstance** command creates the shasti_portdef.props file with content shown in the previous example:

```
wsinstance.bat -name shasti
               -path G:\shasti\WebSphere
               -host planetnt
               -startingPort 20002
               -create
```

Setting up and using the configuration instance environment

Most of the tasks you perform in a configuration instance are done from one shell or command window after you run the setupCmdLine.sh (setupCmdLine.bat) script in the /bin directory of the configuration instance. Such a scenario might be:

1. 1. Create the configuration instance using the **wsinstance** command.
2. 2. In that command window or in another, change directories to the /bin directory of the new instance.
3. 3. Establish a temporary override for the normal WebSphere Application Server environment by running the setupCmdLine.sh (setupCmdLine.bat) script:

   ```
   setupCmdLine
   ```
4. In the same window, start server1 by changing directories to the install_root/bin directory of the original installation and issuing the command. There is no such command in the /bin directory of the configuration instance:

   ```
   startserver server1
   ```

   This works because running the setupCmdLine script from the configuration instance reconfigures the WebSphere Application Server environment. Reconfiguring the environment lets you issue commands from the /bin directory of the original installation as if you were issuing them from the configuration instance. The configuration instance creates a virtual installation that includes the original installation /bin directory. The virtual environment lasts until you close the Linux or UNIX shell or the command window in a Windows environment.

   Even after you close the shell or command window, the server you started is still running in a spawned process that retains the environment settings.

5. See how the environment has changed. Display all the environment variables. You'll notice one that points to the setupCmdLine.sh (.bat) file in the configuration instance. Its name is WAS_USER_SCRIPT. For example, in a Linux bash shell or in the command window on a Windows platform, type:

   ```
   set
   ```

6. See how the environment has changed. Display all the ports for the machine. You'll notice the ports you set for the configuration instance are now present. For example, in a Linux bash shell or in the command window on a Windows platform, type:

   ```
   netstat -a
   ```

7. Open a browser window and point it at your machine at the port you defined for HTTP_TRANSPORT_ADMIN in the `hostname_instanceName_portdef.props` file in the installation root of the new instance. For example, suppose the setting is HTTP_TRANSPORT_ADMIN=20003. Open the administrative console for server1 by pointing your browser at:

   ```
   http://hostname_orIP_address:20003/admin/
   ```

By setting the WAS_USER_SCRIPT environment variable to point to itself, the setupCmdLine script of the configuration instance overrides the setupCmdLine script of the original installation and redirects scripts in the `/bin` directory of the original installation to work on configuration instance objects.

Creating servers in coexistence or multiple instance environments

WebSphere Application Server lets you create multiple servers based on an existing template, or using an existing server as a template. You can generate unique ports for the new server during its creation. Always select the unique port option when creating servers in a coexistence environment of multiple versions, installations or configuration instances, due to the likelihood of conflicting port assignments. Verify port assignments for the newly created server and change them if necessary.

1. Create a server (MyServer, for example) using either the administrative console or wsadmin scripts.
2. Log on to the administrative console.
3. Go to **Servers > Application Servers > MyServer > End Points**.
4. Go into each end point and change the port numbers.
5. Change the HTTP Transport ports.
   a. Go to **Servers > Application Servers > MyServer > Web Container > HTTP Transports**.
   b. Change the HTTP transport port numbers.
   c. Make a record of the new port numbers.
6. Change the JMS server ports, if you create servers from an application server template that has the embedded messaging feature.
   a. Go to **Servers > Application Servers > MyServer > Server Components > JMS Servers > Security Port Endpoint**.
   b. Change the port numbers.
7. Change the JMS server ports, if you create servers from an Application Server template that is not federated into a deployment manager cell, but does have the embedded messaging feature.
   a. Go to **Servers > Application Servers > MyServer > Server Components > JMS Servers > Security Port Endpoint**.

b. Change the port numbers.

To continue, return to one of these topics:
- Installing multiple Version 5 configuration instances
- Migrating and coexisting

Changing HTTP transport ports

This topic describes how to change HTTP transports manually by editing configuration files. Use this procedure when a conflicting HTTP transport setting is preventing an Application Server or deployment manager instance from starting.

Edit the configuration files to resolve conflicting port assignments, as described below.

1. Look for symptoms of port number conflicts. Troubleshooting topics describe symptoms and ways to identify and fix possible port number conflicts.
2. Edit an Application Server configuration.
   a. Open the `server.xml` configuration file for the `server1` application server.

      The file path for the server1 configuration, with a node name of **mynode**, is:

      `*install_root*/config/cells/mynode/nodes/mynode/servers/server1`

   b. Look for `transports xmi:type="applicationserver.webcontainer:HTTPTransport"`.

      The administrative console application uses transport ports 9090 and 9043. The sample applications use transport ports 9080 and 9443. Change the port numbers and save the file. Make a record of the new port numbers.

   c. Open the `virtualhosts.xml` file in the *install_root*`/config/cells/mynode` folder.

      This file contains alias entries for transport ports defined in the `server.xml` file.

   d. Look for `aliases xmi:id` to change port number assignments for any ports you changed.
3. Edit a deployment manager node configuration.

   The deployment manager uses HTTP transport ports for the administrative console application. The default port is 9090.

   a. Open the `server.xml` configuration file for the `dmgr` deployment manager server.

      The file path for the dmgr configuration, with a cell name of **myManager**, and a node name of **mynode**, is:

      `*install_root*/config/cells/myManager/nodes/mynode/servers/dmgr`

   b. Look for `transports xmi:type="applicationserver.webcontainer:HTTPTransport"`.

      The administrative console application uses transport ports 9090 and 9043. Change the port numbers and save the file. Make a record of the new port numbers.

   c. Open the `virtualhosts.xml` file in the *install_root*`/config/cells/mynode` folder.

      This file contains alias entries for transport ports defined in the `server.xml` file.

   d. Look for `aliases xmi:id` to change port number assignments for any ports you changed.

To continue, return to Installing multiple Version 5 configuration instances.

Product version and history information

The WebSphere Application Server product contains structural differences from previous versions. The `/properties/version` directory in the install_root contains important data about the product and its installed components, such as the build version and build date. This information is included in [product].product and [component].component files. The `/properties/version/history` directory in the install_root contains a collection of records for installed interim fixes and fix packs. This information is included in [interim fixID].efixApplied, [interim fixID].efixDriver, [fix packID].ptfApplied, and [fix packID].ptfDriver files. A driver file has useful information about the entire contents of an interim fix or fix pack. The applied file has relevant information about the interim fixes or fix packs that are currently applied. Event.history files are also present. They contain a detailed log about updates you have applied, either successfully or unsuccessfully. Time-stamped, detailed logs record each update process in the `/properties/version/log` directory of the install_root.

This topic describes the XML data files that store product information for Version 5 WebSphere Application Server products. By default, the document type declarations (DTDs) for these files are in the `properties/version/dtd` folder of the install_root, or the server root directory. See the Storage locations section for more information.

This topic includes these sections:
- A list of product information files and file locations
- A list of reports for displaying version and history information
- A description of logs and component backups
- A list of properties you can use to change storage locations
- A description of how the update service makes operational use of the product information
- A data dictionary that describes data in the files

Product information files

XML files in the in the `/properties/version` directory that store version information:

platform.websphere

One file whose existence indicates that a WebSphere Application Server product is installed. An example of the file follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE websphere PUBLIC "websphereId" "websphere.dtd">
<websphere name="IBM WebSphere Application Server" version="5.0"/>
```

The following XML files in the `/properties/version` directory represent installed items and installation events.

<product-id>.**product**

One file whose existence indicates the particular WebSphere Application Server product that is installed. Data in the file indicates the version, build date, and build level. For example, the file might be the `ND.product` file, which indicates that the installed product is WebSphere Application Server Network Deployment. An example of the file follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE product PUBLIC "productId" "product.dtd">
<product name="IBM WebSphere Application Server for Network Deployment">
```

```
    <id>ND</id>
    <version>5.0.0</version>
    <build-info date="10/5/02" level="s0239.28"/>
  </product>
```

<*component-name*>**.component**
: Any number of component files that each indicate the presence of an installed component, which is part of the product. Data in the file indicates the component build date, build version, component name, and product version. For example, the file might be the `activity.component` file, which indicates that the ***activity*** component is installed. The activity component is part of the Network Deployment product. An example of the file follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE component PUBLIC "componentId" "component.dtd">
<component build-date="10/5/02" build-version="s0239.28"
   name="activity" spec-version="5.0"/>
```

<*extension.id*>**.extension**
: Any number of extension files that each indicate the presence of an extension that you install as a user extension, as part of a service engagement, or as installed by a third party product. The <*extension.id*>.extension files are not created, logged, or removed by WebSphere Application Server products.

<*fix-id*>**.efix**
: Any number of interim fix files that each indicate the presence of an installed fix.

<*ptf-id*>**.ptf**
: Any number of files, that each indicate the presence of an installed fix pack.

XML files in the `/properties/version/history` directory that store version history information files:

event.history
: One file that lists update events that have occurred. An update event is an operation that installs or uninstalls an interim fix or a fix pack. The file is sorted by the date and time of the events that are listed.

The following XML files in the `/properties/version/history` directory describe fixes and fix packs that are currently installed. These XML files are related to installation items by the primary ID information, which is shown here by <*angle brackets*> and italicized text.

<*fix-id*>**.efixDriver**
: Interim fix-driver defining information

<*fix-id*>**.efixApplied**
: Interim fix installation details

<*ptf-id*>**.ptfDriver**
: Fix pack-driver defining information

<*ptf-id*>**.ptfApplied**
: Fix pack installation details

Reports

You can view product information by examining files in the `properties/version` directory, including the `properties/version/history` directory.

WebSphere Application Server provides the ability to generate two types of reports about the data in the files, ***Version*** reports and ***History*** reports. The following report-generation scripts are available in the install_root `bin` directory.

Product version reports

The following report generation scripts extract data from XML data files in the `properties/version` folder.

versionInfo.bat
Lets you use parameters to create a version report on Windows platforms. Do not use the **versionInfo** command while installing or uninstalling the product, or while installing or uninstalling an interim fix or fix pack.

Parameters:

-? or /? (Windows only)
Displays command syntax.

-components
Adds a list of installed components to the report.

-componentDetail
Adds details about installed components to the report.

-efixes Adds a list of applied interim fixes to the report.

-efixDetail
Adds details about applied interim fixes to the report.

-file <*fileName*>

Specifies the output file name. The report goes to standard output (stdout) by default.

-format
TEXT | HTML

Selects the format of the report. The default is TEXT.

-help or /help (Windows only)
Displays command syntax.

-long Creates the long version of the report.

-ptfDetail
Adds details about applied fix packs to the report.

-ptfs Adds a list of applied fix packs to the report.

-usage Displays command syntax.

versionInfo.sh
Lets you use parameters to create a version report on UNIX-based platforms. Do not use the **versionInfo** command while installing or uninstalling the product, or while installing or uninstalling an interim fix or fix pack. Parameters are the same as for the Windows version.

genVersionReport.bat
Generates the `versionReport.html` report file in the `bin` directory on Windows platforms. The report includes the list of components, fixes, and fix packs.

genVersionReport.sh
Generates the `versionReport.html` report file in the `bin` directory on UNIX-based platforms. The report includes the list of components, fixes, and fix packs.

Product history reports

The following report generation scripts extract data from XML data files in the `properties/version/history` folder.

historyInfo.bat
Lets you use parameters to create a history report of installed and uninstalled fixes and fix packs, on Windows platforms. You can also specify a component name to create a report that shows the history for that component.

Parameters:

-format

TEXT | HTML

Selects the format of the report. The default is TEXT.

-file *<fileName>*

Specifies the output file name. The report goes to standard output (stdout) by default.

-updateID <ID>

Specifies the ID of an interim fix or fix pack update. When specified, the product history report displays events for only the specified update. When not specified, the report displays events for all updates.

-component <componentName>

Specifies the name of a component. When specified, the product history report displays events for only the named component. When not specified, the report displays events for all components.

historyInfo.sh

Lets you use parameters to create a history report on UNIX-based platforms. Parameters are the same as for the Windows version.

genHistoryReport.bat

Generates the `historyReport.html` report file in the `bin` directory on Windows platforms. The report includes all updates and components.

genHistoryReport.sh

Generates the `historyReport.html` report file in the `bin` directory on UNIX-based platforms. The report includes all updates and components.

Logs and component backups

WebSphere Application Server products use two other directories when performing update operations, for logging and backups:

install_root/**properties/version/log**

Product updates log directory

WebSphere Application Server products store log files to document component, interim fix and fix pack operations and updates.

install_root/**properties/version/backup**

Product updates backup directory

WebSphere Application Server products back up components before applying interim fixes and fix packs. If you uninstall an interim fix or fix pack, WebSphere Application Server products restore the backed-up component JAR file.

File naming convention

Time stamp

YYYYMMDD_HHMMSS

For example: `20020924_211832` is 24-Sep-2002, 9:18:32 pm, GMT. All time stamps are in GMT.

ID Interim fix ID or fix pack ID

For example: apar6789c is an interim fix ID; PTF_1 is a fix pack ID.

Operation

install | uninstall

Interim fix log file names

`<timeStamp>_<fixId>_<operation>.log`

For example:`properties/version/log/20020924_211832_apar6789c_install.log` and `properties/version/log/20020924_211912_apar6789c_uninstall.log`

Interim fix component log file names

`<`*`timeStamp`*`>_<`*`fixId`*`>_<`*`componentName`*`>_<`*`operation`*`>.log`

For example: `properties/version/log/20020924_211832_apar6789c_ras_install.log` and `properties/version/log/20020924_211912_apar6789c_ras_uninstall.log`

Fix pack log file names

`<`*`timeStamp`*`>_<`*`ptfId`*`>_<`*`operation`*`>.log`

For example: `properties/version/log/20030324_211832_was50_fp1_install.log` and `properties/version/log/20030325_211912_was50_fp1_uninstall.log`

Fix pack component log file names

`<`*`timeStamp`*`>_<`*`ptfId`*`>_<`*`componentName`*`>_<`*`operation`*`>.log`

For example: `properties/version/log/20030324_211832_was50_fp1_ras_install.log` and `properties/version/log/20030325_211912_was50_fp1_ras_uninstall.log`

Backup JAR file names

`<`*`timeStamp`*`>_<`*`ptfId`*`>_<`*`componentName`*`>_undo.jar` or `<`*`timeStamp`*`>_<`*`fixId`*`>_<`*`componentName`*`>_undo.jar`

For example: `20020924_211832_apar6789c_ras_undo.jar`

Note: Do not delete a backup JAR file. You cannot remove a component update if the corresponding backup JAR file is not present.

Update processing might also use a temporary directory, if necessary. A Java property specifies this directory as described in the next section.

Storage locations

Product information files are located relative to the WebSphere Application Server product install_root, or the server root directory.

Default file paths are:

Version directory

`install_root``/properties/version` or `<`*`server_root`*`>/properties/version`

History directory

`install_root``/properties/version/history`

Updates log directory

Updates backup directory

`install_root``/properties/version/backup`

DTD directory

`install_root``/properties/version/dtd`

Temporary directory

Specified by the java.io.tmpdir Java system property

Operational description

WebSphere Application Server products update the product version history information while performing events that install or uninstall fixes or fix packs. Events that might occur include:

- A WebSphere Application Server product adds an interim fix file (with an extension of `.efix`) to the `version` directory to indicate that an interim fix is currently installed.
- A WebSphere Application Server product removes an interim fix file from the `version` directory when it uninstalls the corresponding fix.
- A WebSphere Application Server product adds an interim fix driver file (with an extension of `.efixDriver`) to the `history` directory when an interim fix is installed. an interim fix driver file contains defining information for a fix.
- A WebSphere Application Server product removes an interim fix driver file when it removes the corresponding fix.
- A WebSphere Application Server product adds an interim fix application file (with an extension of `.efixApplied`) to the `history` directory when it installs an interim fix. An interim fix application file contains information that identifies component updates that have been applied for a fix. The application file also provides links to component log and backup files.
- A WebSphere Application Server product removes an interim fix application file when it removes the corresponding fix.
- A WebSphere Application Server product adds a fix pack, with an extension of `.ptf`, to the `version` directory to indicate than a fix pack is currently installed.
- A WebSphere Application Server product removes a fix pack file from the version directory when it uninstalls the corresponding fix pack.
- A WebSphere Application Server product adds a fix pack driver file (with an extension of `.ptfDriver`) to the `history` directory when it installs a fix pack. A fix pack driver file contains defining information for a fix pack.
- A WebSphere Application Server product adds a fix pack application file (with an extension of `.ptfApplied`) to the `history` directory when it installs a fix pack. A fix pack application file contains information that identifies component updates that have been applied for a fix pack. The application file also provides links to component log and backup files.
- A WebSphere Application Server product makes entries in the history file, `event.history`, when it installs or uninstalls an update.
- A WebSphere Application Server product stores a parent event for each interim fix that it installs or uninstalls.
- A WebSphere Application Server product stores a parent event for each fix pack that it installs or uninstalls.
- A WebSphere Application Server product stores child component events for each component update that it installs or uninstalls, beneath the corresponding interim fix or fix pack parent event.
- A WebSphere Application Server product stores one log file in the `logs/update` directory as it installs or uninstalls one interim fix or fix pack.
- A WebSphere Application Server product stores one log file in the `logs/update` directory as it installs or uninstalls an interim fix or fix pack, in response to each component update that occurs.
- A WebSphere Application Server product stores a component backup file in the backup directory for each component update that it installs.
- A WebSphere Application Server product removes a component backup file from the backup directory for each component update that it uninstalls.

Data dictionary

Type Family: websphere product family

File Types:
websphere

File Type:
websphere

Elements:

```
name                  string     required
version               string     required
```

Persistence:
<versionDir>/platform.websphere

Type Detail:

The websphere file denotes the presence of WebsSphere family products.

Element Detail:

```
websphere.name          The WebSphere product family name.
websphere.version       The WebSphere product family version.


Type Family:      product

File Types:       product
                  component
                  extension

File Type:        product

Persistence:      <versionDir>/<id>.product

Elements:         id                    string     required
                  name                  string     required
                  version               string     required
                  build-info            complex    required

Type Detail:

A product file is placed to denote the presence of a specific
WebSphere family product.
The product ID is embedded in the product file name.

Element Detail:

product.id              The id of the product.
product.name            The name of the product.
product.version         The version of the product.
product.build-info      An element containing build information for
                        the product.

Element Type:     build-info

Elements:         date                  date       required
                  level                 string     required

Type Detail:

A build-info instance details the build of a specific installed
WebSphere family product.

Element Detail:

build-info.date         The date on which the product was build.
build-info.level        The level code of the product's build.

File Type:        component

Persistence:      <versionDir>/<name>.component

Elements:         name                  string     required
```

```
                  spec-version          string     required
                  build-version         string     required
                  build-date            date       required

File Detail:

A component file denotes the presence of a specific component.
The component name is embedded in the component file name.

Element Detail:

component.name           The name of the component.
component.spec-version   The specification version of the component.
component.build-version  The build level of the component.
component.build-date     The build date of the component.

File Type:        extension

Persistence:      <versionDir>/<id>.extension

Elements:         id                    string     required
                  name                  string     required

File Detail:

An extension file denotes the presence of a specific extension.  The
extension's id is embedded in the extension file name.

The elements of an extension file are minimally specified.  The listed
elements are required.  Additional elements may be present as determined
by the actual installed extension.

Element Detail:

extension.id             The id of the extension.
extension.name           The name of the extension.


Type Family:      update

File Types:       efix
                  ptf
                  efix-applied
                  ptf-applied

File Type:        efix

Persistence:      <versionDir>/<id>.efix

Elements:         id                    string     required
                  apar-number           string     optional
                  pmr-number            string     optional
                  short-description     string     required
                  long-description      string     required
                  is-temporary          boolean    required
                  build-version         string     required
                  build-date            date       required
                  component-update      complex    min=1, max=unbounded
                  platform-prereq       complex    min=0, max=unbounded
                  product-prereq        complex    min=0, max=unbounded
                  efix-prereq           complex    min=0, max=unbounded
                  custom-property       complex    min=0, max=unbounded

Type Detail:

An efix file denotes the presence of some portion of a specific fix.
The ID of the interim fix is embedded in the file name.
```

```
An efix file contains all interim fix data, such as description,
a listing of component updates, and prerequisite information.

Almost always, when installing a fix, all of the potential component
updates within the interim fix are required to be installed.

A separate application file must be examined to determine the components
which have been updated for a particular fix.

A list of custom properties may be provided.  These are provided for
future use.

Element Detail:

efix.id                  The id of the fix.

efix.short-description   A short description of the fix.

efix.long-description    A long description of the fix.

efix.is-trial            A flag indicating whether or not an interim fix is
                         considered a trial fix.  Generally, a trial interim
                         fix will be followed up with a more permanent fix.

efix.expiration-date     A date on which the interim fix is obsolete.

efix.build-version       The build version of the fix.  This is distinct from
                         the build version of component updates contained within
                         the fix.

efix-build-date          The build date of the fix.  This is distinct from the
                         build version of the component updates contained within
                         the fix.

efix.apar-info           A list of APAR's which are associated with the fix.

efix.component-update    A list of updates for components.  For a fix, these are
                         usually all required, and are all patch updates. At least
                         one component update must be present.

efix.efix-prereq         A list of prerequisite fixes for the fix. Note that
                         prerequisite fixes may be negative (see below). This
                         list may be (and is often) empty.

efix.plaform-prereq      A list of platforms on which the interim fix may be installed.
                         This list may be empty, in which case the interim fix may be
                         installed on all platforms.

efix.product-prereq      A list of products on which the interim fix may be installed.
                         This list may be empty, in which case the interim fix may be
                         installed on all products.

efix.custom-proprty      A list of properties, provided for future use.


File Type:        ptf

Persistence:      <versionDir>/<id>.ptf

Elements:         id                     string     required
                  short-description      string     required
                  long-description       string     required
                  build-version          string     required
                  build-date             date       required
                  component-update       complex    min=1, max=unbounded
                  product-update         complex    min=0, max=unbounded
                  platform-prereq      complex    min=0, max=unbounded
                  product-prereq       complex    min=0, max=unbounded
                  included-efix        complex    min=0, max=unbounded
                  custom-property      complex    min=0, max=unbounded

Type Detail:

A ptf file denotes the presence of some portion of a specific fix pack.
The id of the fix pack is embedded in the fix pack file name.

A ptf file contains all fix pack data, such as description, a listing of
component updates, and prerequisite information.

Usually, when installing a fix pack, you can omit certain potential component
updates, but only when the corresponding component is not installed.

You must examine a separate application file, to determine which components
a particular fix pack has updated.

A fix pack can include updates for a number of fixes.

A list of custom properties might be provided. These are for future use.

Element Detail:

ptf.id                   The ID of the fix pack.

ptf.short-description    A short description of the fix pack.

ptf.long-description     A long description of the fix pack.

ptf.build-version        The build version of the fix pack.
                         This is distinct from the build version of
                         component updates contained within the fix pack.

ptf-build-date           The build date of the fix pack.  This is distinct
                         from the build version of the component updates
                         contained within the
                         fix pack.

ptf.component-update     A list of updates for components.  For a fix pack,
                         these are usually all required, and are all patch updates.
                         At least one component update must be present.

ptf.plaform-prereq       A list of platforms on which you can install the fix pack.
                         This list might be empty. If so, you can install the fix
                         pack on all platforms.

ptf.product-prereq       A list of products on which you can install the fix pack.
                         This list might be empty. If so, you can install the fix
                         pack on all products.

ptf.included-efix        A list of fixes which are included (fixed) by the fix pack.

ptf.custom-proprty       A list of properties, provided for future use.

Element Type:     component-update

Elements:         component-name       string     required
                  update-type          enum       required [enumUpdateType]
                  is-required          boolean    required
                  is-recommended       boolean    required
                  is-optional          boolean    required
                  is-external          boolean    required
                  root-property-file   anyURL     optional
                  root-property-name   string     optional
                  root-property-value  anyURL     optional
                  is-custom             boolean    required
                  primary-content       string     required
                  component-prereq      complex    min=0, max=unbounded
                  final-version         complex    optional
                  custom-property       complex    min=0, max=unbounded

Type Detail:

A component update represents a potential component update which is packaged
in an update (an interim fix or a fix pack).

An component update may be required, in which case the parent update may not
be installed unless the component update can be installed.  (A component update
can be installed if the corresponding component is installed.)

A component update may be a custom update, in which case the content which
was provided must be an executable file.  Otherwise, the content which is
provided must be an update JAR file.

A component update has a type.  A final version may be required according to
the update type.

Element Detail:

component-update.component-name   The name the component which is to be updated.

component-update.update-type      The type of the component update, one of 'add',
                                  'replace', 'remove', or 'patch'.  Final version
                                  information must be provided when the update type
                                  is 'add' or 'replace'.

component-update.is-required      A flag which, when true, specifies that the parent
                                  update may not be applied unless this component
                                  update is applied.

component-update.is-recommended   A flag which, when true, specifies that this
                                  component update, although optional, should be
                                  installed.

component-update.is-optional      A flag which, when true, specifies that this update
                                  may be omitted even if its corresponding component
                                  is installed.

component-update.is-external          A flag which, when true, specifies that this
                                      component update may live outside of the usual
                                      install root.


component-update.root-property-file   For a component with an external root, this
                                      properties file provides the root value.


component-update.root-property-name   For a component with an external root,
                                      this named property provides the root value.


component-update.root-property-value  For a component with an external root, this value
                                      provides the default root value.


component-update.is-custom        A flag which, when true, specifies that the update
                                  is a custom update.  When true, the content must
                                  be an executable program.  When false, the content
                                  must be an update JAR.


component-update.primary-content  The name of the content which is provided for the
```

```
                                  update.  This will be an entry which is packaged
                                  in the 'components' directory of the update.


component-update.component-prereq A list of component versions, one of which must
                                  be present for this update to be installed.
                                  When this list is empty, any component version
                                  is allowed.


component-update.final-version    Final version information for the component.
                                  A final version is required when the update
                                  operation is 'add' or 'replace'.


component-update.custom-property  A list of properties, provided for future use.


Element Type:     apar-info

Elements:         number                string     required
                  date                  date       required
                  short-description     string     required
                  long-description      string     optional

Type Detail:

An apar-info object provides information about an APAR which is associated
with a fix, usually indicating that the interim fix provides an interim fix
for the APAR.

Element Detail:

apar-info.number             The number of the associated APAR.

apar-info.date               The date of the APAR.

apar-info.short-description  A short description of the APAR.

apar-info.long-description   An optional long description of the APAR.

Element Type:     efix-prereq

Elements:         efix-id               string     required
                  is-negative           boolean    required
                  install-index         int        optional

Type Detail:

An efix-prereq instance denotes an interim fix that must be present
(or, if negative, must be absent) for the parent interim fix to be installed.

efix-prereq instances can specify a cycle, in which case the prerequisite
specification is treated as a corequisite specification.

The following chart summarizes the interpretation of prerequisite information
for two fixes:

       fix1       fix2
        -          -         Install the fixes without regard to each other.
       fix2        -         fix1 must be installed after fix2 is installed.
        -         fix1       fix2 must be installed after fix1 is installed.
       fix2       fix1       fix1 and fix2 must be installed together.
      !fix2        -         fix1 may not be installed after fix2 is installed.
        -        !fix1       fix2 may not be installed after fix1 is installed.
      !fix2      !fix1       fix1 and fix2 may not ever be installed together.
      !fix2       fix1       This is an erroneous specification.
```

```
       fix2      !fix1        This is an erroneous specification.

The installation index element provides ordering information for corequisite
fixes that must be installed in a particular order.

Element Detail:

fix-prereq.efix-id         The id of the prerequisite fix.

fix-prereq.is-negative     A flag which indicates if the prerequisite
                           interim fix is required or prohibited.  If false,
                           you must install the interim fix before installing
                           the parent fix.  If true, you must not install
                           the interim fix before you install the parent fix.

fix-prereq.install-index   An optional index number which is used to
                           order corequisite fixes.

Element Type:     product-update

Elements:         product-id            string     required
                  product-name          string     required
                  build-version         string     required
                  build-date            date       required
                  build-level           string     required

Type Detail:

A product update specifies a replacement to a product file.

The product update information matches the information in product files.

Multiple product updates may be present, in which case each matching
product is updated.

Element Detail:

product-update.product-id     The id of the product that is updated.

product-update.product-name   The name of the product.

product-update.build-version  The build version of the product.

product-update.build-date     The build date of the product.

product-update.build-level    The build level of the product.

Element Type:     component-prereq

Elements:         component-name        string     required
                  spec-version          string     required
                  build-version         string     required
                  build-date            date       required

Element Type:     platform-prereq

Elements:         architecture          string     required
                  os-platform           string     optional
                  os-version            string     optional

Type Detail:

A platform prerequisite instance denotes a platform which must be present
for an update to be installed.  The element values are according to the
values supplied for the matching java properties.

Note that when multiple platform prerequisites are specified, these
```

```
prerequisites have an OR relationship: At least one of the platform
prerequisites must be satisfied.

Element Detail:

platform-prereq.architecture  The name of an architecture which must be
                              present.

platform-prereq.os-platform   The name of an operating system which
                              must be present.  This element is optional.
                              When absent, the architecture is checked,
                              but the os-platform and os-version are not.

platform-prereq.os-version    The version of a the operating system which
                              must be present.  This element is optional.
                              When absent, the architecture and os-platform
                              are checked, but os-version is not.  (When
                              os-platform is absent, os-version should not
                              be set.)

Element Type:     product-prereq

Elements:         product-id           string     required
                  build-version        string     optional
                  build-date           date       optional
                  build-level          string     optional

Type Detail:

A product prerequisite specifies that a particular product must be present
for an update to be installed.

Note that when multiple product prerequisites are specified, these
prerequisites have an OR relationship: At least one of the product
prerequisites must be satisfied.

Note that all of the elements are required.  When multiple products having
the same id are supported by an update, multiple product prerequisites must
be specified.

Element Detail:

product-prereq.product-id     The id of the product which must be present.

product-prereq.build-version  The version of the product which must be present.

product-prereq.build-date     The build date of the product which must be present.

product-prereq.build-level    The level date of the product which must be present.

Element Type:     component-prereq

Elements:         component-name       string     required
                  spec-version         string     required
                  build-version        string     required
                  build-date           date       required

Type Detail:

A version prerequisite specifies that a particular component version must be
present for an update to be installed.

Note that when multiple version prerequisites are specified, these
prerequisites have an OR relationship: At least one of the version
prerequisites must be satisfied.

Element Detail:
```

```
version-prereq.component-name  The name of the component which must be present.

version-prereq.spec-version    The specification version of the component which
                               must be present.

version-prereq.build-version   The version of the component which must be present.

version-prereq.build-date      The build date of the component which must be
                               present.


Element Type:     included-efix

Elements:         efix-id              string     required

Type Detail:

An included-efix identifies an interim fix by ID and indicates that the fix
is included in the fix pack.

Element Detail:

included-efix.efix-id    The ID of the interim fix that the fix pack includes.

Element Type:     custom-property

Elements:         property-name        string     required
                  property-type        string     optional
                  property-value       string     optional

Type Detail:

A custom property encodes a key-value pair, with an optional type
element.  Custom properties are provided for future use.

Element Detail:

custom-property.property-name   The name of the custom property.

custom-property.property-type   An optional type of the custom property.
                                The semantics of this type are defined
                                by user of the property value.

custom-property.property-value  The value of the custom property.

File Type:        efix-applied

Persistence:      <versionDir>/<id>.efixApplied

Elements:         efix-id              string     required
                  component-applied    complex    min=0, max=unbounded

Type Detail:

An efix-applied collection specifies what components have been updated for
the interim fix as specified by the efix-id.

Element Detail:

efix-applied.efix-id            The id of the interim fix for which applieds
                                are recorded.

efix-applied.component-applied  The list of recorded applications.

File Type:        ptf-applied
```

```
Persistence:      <versionDir>/<id>.ptfApplied

Elements:         ptf-id                string     required
                  component-applied     complex    min=0, max=unbounded

Type Detail:

A ptf-applied collection specified what components have been updated for
the fix pack as specified by the fix pack ID.

Element Detail:

ptf-applied.efix-id            The ID of the fix pack for which applieds are
                               recorded.

ptf-applied.component-applied  The list of recorded applications.

Element Type:     component-applied

Elements:         component-name        string     required
                  update-type           enum       required [enumUpdateType]
                  is-required           boolean    required
                  is-optional           boolean    required
                  is-external           boolean    required
                  root-property-file    anyURL     optional
                  root-property-name    string     optional
                  root-property-value   string     optional
                  is-custom             boolean    required
                  log-name              anyURL     required
                  backup-name           anyURL     required
                  time-stamp            date       required
                  initial-version       complex    optional
                  final-version         complex    optional

Type Detail:

An applied instance is present to indicate the application of an
update for a particular interim fix or fix pack to a particular component.
(The particular interim fix or fix pack is as specified by the applied's
parent.)  An applied provides sufficient information
to undo itself.

The elements of an applied are copies of values from update
events.

Element Detail:

component-applied.component-name   The name of the component which was updated.

component-applied.update-type      The type of the component update.

component-applied.is-required      A true value specifies that the parent
                                   update requires this component update.

component-applied.is-optional      A flag which, when true, specifies that the parent
                                   update does not require this component update,
                                   even if the component is installed.

component-applied.is-external          A flag which, when true, specifies that this
                                       component update was applied to a location
                                       different than the usual install_root.

component-applied.root-property-file   For an update against a component having an
                                       external root, this properties file provides
                                       the root value.

component-applied.root-property-name   For an update against a component having an
```

```
                                       external root, this named property provides
                                       the root value.

component-applied.root-property-value  For an update against a component having an
                                       external root, this is a record of the
                                       actual external root.

component-applied.is-custom        A flag which, when true, specifies that the
                                   application was a custom update.  When true, an
                                   executable program was applied.  When false, the
                                   contents of an update JAR were applied.

component-applied.log-name         The name of the log file that was generated by
                                   this application.

component-applied.backup-name      The name of the backup file which was generated by
                                   this application.

component-applied.time-stamp       The time of this application (the ending time of
                                   the corresponding update event).

component-applied.initial-version  The version of the component before the
                                   application.  This version will be null if
                                   the application was an add.

component-applied.final-version    The version of the component after application.
                                   This will be null if the update was a removal.


Element Type:     initial-version

Elements:         component-name        string     required
                  spec-version          string     required
                  build-version         string     required
                  build-date            string     required

Type Detail:

A initial-version instance is used to describe a component level as
the initial version of a component.

Element Detail:

initial-version.component-name The name of the component.

initial-version.spec-version   The new specification version for the
                               component following the update.

initial-version.build-version  The new build version for the component.

initial-version.build-date     The new build date for the component.

Element Type:     final-version

Elements:         component-name        string     required
                  spec-version          string     required
                  build-version         string     required
                  build-date            string     required

Type Detail:

A final-version instance is used to supply a component level for a
component which has been added or replaced.

Element Detail:

final-version.component-name The name of the new component.
```

```
final-version.spec-version   The new specification version for the
                             component following the update.

final-version.build-version  The new build version for the component.

final-version.build-date     The new build date for the component.


Enum Type:        enumUpdateType

Values:           0 add
                  1 replace
                  2 remove
                  3 patch

Type Detail:

An update type instance specifies the type of an update.  An 'add' update adds
a component into an installation.  A 'replace' update replaces a particular
version of a component with a different version of that component.  A 'remove'
update removes a component.  A 'patch' update performs a limited update to a
component, in particular, without changing the version of the component.

When adding a component, that component may not already be present.
When replacing or removing a component, that component must be present.
When patching a component, that component must be present.

When replacing or removing a component, or when patching a component, usually,
at least one version prerequisite will be specified for the component update.

Value Detail:

enumUpdateType.add       Specifies that an update adds a component.

enumUpdateType.replace   Specifies that an update replaces a component.

enumUpdateType.remove    Specifies that an update removes a component.

enumUpdateType.patch     Specifies that an update modifies a component, but
                         does not change its version.


Type Family:      history

File Type:        event-history

Persistence:      <historyDir>/event.history

Elements:         update-event          complex    min=0, max=unbounded

Type Detail:

One event history is provided for a websphere product family installation.
This event history contains history of update events, corresponding with
the actual update events for that product family.

Element Detail:

event-history.update-event  The list of update events for the websphere
                            product family.  The top level events are fix
                            and fix pack events, each containing one or more
                            component events.

Element Type:     update-event

Elements:         event-type            enum       required [enumEventType]
```

```
                  parent-id             string     required
                  id                    string     required

                  update-type           enum       required [enumUpdateType]

                  is-required           boolean    required
                  is-optional           boolean    required

                  is-external           boolean    required
                  root-property-file    anyURL     optional
                  root-property-name    string     optional
                  root-property-value   string     optional

                  is-custom             boolean    required
                  primary-content       anyURI     required

                  event-action          enum       required [enumEventAction]
                  log-name              anyURI     required
                  backup-name           anyURI     required
                  start-time-stamp      dateTime   required
                  end-time-stamp        dateTime   optional

                  status                enum       optional [enumEventResult]
                  status-message        string     optional

                  initial-version       complex    optional
                  final-version         complex    optional
                  update-event          complex    optional

Type Detail:

An update event denotes a single update action, applying to either a
fix, a fix pack, or to a component, according to the set event type.

Interim fix (efix) and fix pack (ptf) type events each have a
collection of component events.

Currently, component events have no child events.

Element Detail:

update-event.event-type        The type of this event, either an interim fix or
                               fix pack (ptf) type event, or a component type event.

update-event.parent-id         This element is present only for component
                               events. The ID of the parent interim fix or fix
                               pack of this event.

update-event.id                The ID of the fix, fix pack, or component that
                               was updated, interpreted according to the type
                               of the event.

update-event.update-type       The type of update for component events.

update-event.is-required       A flag which, when true, specifies that this
                               component update is required.

update-event.is-optional       A flag which, when true, specifies that this
                               component update is optional, even if the
                               component is installed.

update-event.is-external          A flag which, when true, specifies that this
                                  update used an external root.

update-event.root-property-file   For an update of an external component, this
                                  properties file contains the external root value.
```

```
update-event.root-property-name   For an update of an external component, the
                                  property having this name specifies the external
                                  root value.

update-event.root-property-value  For an update of an external component,
                                  the root value.

update-event.is-custom         A flag that, when true, specifies that the
                               application was a custom update.  When true,
                               an executable program was applied.  When false,
                               the contents of an update JAR file were applied.

update-event.primary-content   The URL of the primary content for the update.

update-event.event-action      The type of action for this event.

update-event.log-name          The name of the log file that was generated
                               for this event.

update-event.backup-name       The name of the backup file that was generated
                               for this event.

update-event.start-time-stamp  The XML timestamp of the starting time of the
                               event.  This timestamp follows the XML timestamp
                               format, meaning that time zone information is
                               included.

update-event.end-time-stamp    The XML timestamp of the ending time of the
                               event.  This timestamp follows the XML timestamp
                               format, meaning that time zone information is
                               included.  When absent, the update operation
                               corresponding to the parent event failed with
                               a non-recoverable exception.

update-event.status            The result of the update.

update-event.status-message    Message text provided in addition to the basic
                               status code.  Exception text is provided through
                               the status-message when an update fails.

update-event.initial-version   This element is not used unless the update is
                               a component type update.  The initial version of
                               the component which was updated.    This element
                               is absent when the update is an add type update.

update-event.final-version     This element is not used unless the update is a
                               component type update.  The final version of the
                               component which was updated.  This element is absent
                               when the update is a remove type update.

update-event.update-event      A collection of child events.  This collection is
                               used for interim fix and fix pack type events.
                               This collection is empty for component type events.

Element Type:     initial-version

Elements:         spec-version          string     required
                  build-version         string     required
                  build-date            string     required

Type Detail:

A initial-version instance is used to describe a component level as
the initial version of a component.

Element Detail:
```

```
initial-version.spec-version   The new specification version for the
                               component following the update.

initial-version.build-version  The new build version for the component.

initial-version.build-date     The new build date for the component.

Element Type:     final-version

Elements:         spec-version          string     required
                  build-version         string     required
                  build-date            string     required

Type Detail:

A final-version instance is used to supply a component level for a
component which has been added or replaced.

Element Detail:

final-version.spec-version   The new specification version for the
                             component following the update.

final-version.build-version  The new build version for the component.

final-version.build-date     The new build date for the component.

Enum Type         enumEventType

Values:           0 Interim fix (efix)
                  1 Fix pack (ptf)
                  2 Component

Type Detail:

An event type instance specifies the type of an update event, which is either
an interim fix (efix) event, a fix pack (ptf) event or a component event. The
interpretation of particular event elements depends on the set event type.

Value Detail:

enumEventType.efix       Specifies that an event is for an interim fix update.

enumEventType.ptf        Specifies that an event is for a fix pack update.

enumEventType.component  Specifies that an event is for a component update.

Enum Type:        enumEventAction

Values:           0 Install
                  1 Uninstall
                  2 Selective install
                  3 Selective uninstall

Type Detail:

An event action instance specified the operation performed by an update, which
can be an install or uninstall operation, and which may be a selective operation.
Component operations are always either install or uninstall type operations, only
interim fix and fix pack operations may be selective operations.

A selective operation is an installation which is applied to a preset list of
components.  In particular, potential component updates may be skipped, and
component updates which were already applied may be reapplied.

A selective uninstall operation is used to back out an update which was cancelled
by the user.
```

```
Value Detail:

enumEventAction.install             Specifies that an event is an install
                                    operation.

enumEventAction.uninstall           Specifies that an event is an uninstall
                                    operation.

enumEventAction.selective-install   Specifies that an event is an install
                                    operation with a preset list of components
                                    that are updated.

enumEventAction.selective-uninstall  Specifies that an event is an uninstall
                                    operation with a preset list of components
                                    that are updated.

Enum Type:        enumUpdateType

Values:           0 Add
                  1 Replace
                  2 Remove
                  3 Patch

Type Detail:

An update type instance specifies the type of a component update.
An 'add' update adds a component into an installation.
A 'replace' update replaces a particular version of a component with
a different version of that component.
A 'remove' update removes a component.
A 'patch' update performs a limited update to a component,
in particular, without changing the version of the component.

To add a new component, the component must not exist.
To replace or remove a component, the component must exist.
To patch a component, the component must exist.

When replacing or removing a component, or when patching a component,
usually, at least one version prerequisite is specified for the
component update.

Value Detail:

enumUpdateType.add       Specifies that an update adds a component.

enumUpdateType.replace   Specifies that an update replaces a component.

enumUpdateType.remove    Specifies that an update removes a component.

enumUpdateType.patch     Specifies that an update modifies a component, but
                         does not change its version.

Enum Type:        enumEventResult

Values:           0 Succeeded
                  1 Failed
                  2 Cancelled

Type Detail:

An event result instance denotes a particular result for an update event.
The result indicates success, failure, or cancellation.

Value Detail:

enumEventResult.succeeded  Specifies that the operation was successful.
```

`enumEventResult.failed` Specifies that the operation failed.

`enumEventResult.cancelled` Specifies that the operation was cancelled.

Installation: Resources for learning

Use the following links to find relevant supplemental information about installation. 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 about:
- Planning, business scenarios, and IT architecture
- Programming model and decisions
- Programming instructions and examples
- Programming specifications
- Administration
- Support

Planning, business scenarios, and IT architecture
- IBM WebSphere Application Server supported hardware, software, and APIs

  The official site for determining product prerequisites for hardware, software and APIs for all WebSphere Application Server products.
- IBM WebSphere Developer Domain

  The home of technical information for developers working with WebSphere products. You can download WebSphere software, take a fast path to developer domain 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.
- IBM WebSphere Application Server library and InfoCenters Web site

  The IBM WebSphere Application Server Library Web site contains links to all WebSphere Application Server InfoCenters, for all versions. It also lets you access each InfoCenter in your native language.
- IBM WebSphere Application Server home page

  The IBM WebSphere Application Server home page contains useful information, including support links and downloads for fixes, APARs, tools, and trials.
- IBM WebSphere software platform home page

  The IBM WebSphere software platform home page introduces WebSphere products and describes how companies can easily transform to an e-business, with software that can grow as fast as the business it supports.
- Migrating to WebSphere V5.0: An End-to-End Migration Guide, SG24-6910-00

  This IBM Redbook is the definitive migration guide for migrating earlier versions of WebSphere Application Server to Version 5. The Redbook adds a broader scope, including planning for application migration and WebSphere Studio Application Developer tooling and samples. Read this book to formulate an optimal migration strategy.
- Just announced: IBM WebSphere Application Server Version 5!

The IBM WebSphere Application Server Version 5 product announcement.

- What's new in IBM WebSphere Application Server Version 5?

  The official list of new features at the What's new in IBM WebSphere Application Server Version 5? Web page.
- IBM WebSphere Application Server Version 5 fact sheet

  This fact sheet describes IBM WebSphere Application Server Version 5.
- The power of Edge of Network technology in IBM WebSphere Application Server Version 5

  A description of WebSphere Application Server Edge Components Version 5.
- WebSphere Application Server - Express, V5

  A description of WebSphere Application Server Express, Version 5.
- WebSphere Application Server, Version 5

  A description of the base product, WebSphere Application Server, Version 5.
- WebSphere Application Server Enterprise, Version 5

  A description of WebSphere Application Server Enterprise, Version 5.
- IBM WebSphere Application Server for z/OS, Version 5

  A description of WebSphere Application Server for z/OS, Version 5.
- InfoCenter for WebSphere Application Server Edge components

  The InfoCenter for WebSphere Application Server Edge components contains complete documentation for the Caching Proxy and the Load Balancer in these PDF online books, *WebSphere Application Server Concepts, Planning, and Installation for Edge Components*, the *WebSphere Application Server Caching Proxy Administration Guide*, and the *WebSphere Application Server Programming Guide for Edge Components*.
- IBM's Web Services architecture debuts

  Introducing IBM Web Services, a distributed software architecture of service components. This brief overview and in-depth interview on IBM developerWorks covers the fundamental concepts of Web Services architecture and what they mean for developers. The interview with Rod Smith, Vice President of Emerging Technologies at IBM, explores which types of developers Web Services targets, how Web services reduce development time, what developers can do with Web Services now, and takes a glance at the economics of dynamically discoverable services.
- developerWorks: Patterns for e-business: Redbooks listing

  This Web page lists links to pattern resources under these categories:
  - Current patterns Redbooks
  - Superseded patterns Redbooks (valid for back-level product versions)
  - Independent analyst reports
  - Patterns CD order offer
  - Back-level version patterns Web site (zip downloads and old Flash tutorial)
  - Customer references
  - White papers
  - Multimedia presentations and screen cams
  - Webcasts
  - Patterns development kit
  - WebSphere technical exchange presentations
- developerWorks: IBM Patterns for e-business

  The IBM developerWorks site is the source for IBM patterns for e-business, a set of tested, reusable intellectual assets that you can use to design and implement your e-business network and architecture!
- Self-Service: Select application pattern

  This Web page describes the self-service business pattern, also known as the User-to-Business or U2B pattern. The self-service e-business pattern captures the essence of direct interactions between interested parties and an e-business. Interested parties include customers, business partners, stakeholders, employees, and all other individuals with whom the business intends to interact.

- Self-Service: Stand-alone single channel application pattern: Run-time patterns

  This Web page describes alternative run-time solution patterns for the self-service business pattern. These run-time patterns are important concepts that any e-business designer should become familiar with.
- Patterns for e-business: A Strategy for Reuse

  Get an inside look at how successful businesses build their e-business architectures. In this book, four IBM e-business experts capture years of experience into easy-to-follow guidelines. Deliberately focusing on Business patterns, integration patterns, and application patterns, the authors share with you proven architectural patterns that can help get you up and running quickly, while at the same time reducing your risks. Because today's economy demands that e-business initiatives emphasize profitability and return on investment, the authors also offer guidance on methods to minimize cost, yet ensure quality.
- Patterns and Web Services

  Patterns architects have reviewed the impact emerging Web Services technologies have on each of the asset layers of the Patterns for e-business designs. Their findings are summarized in this new White paper, in PDF Format.
- developerWorks: Facilitating the application development process using the IBM Patterns for e-business

  This is the most recent White paper from John Lord, a well known IBM consulting IT architect. The paper analyzes the successful use of the Patterns for e-business in an application development scenario.
- Self-Service Patterns using WebSphere Application Server, Version 4.0, SG24-6175-00

  This Redbook discusses the Stand-alone Single Channel application pattern of the Self-Service business patterns. This application pattern describes a situation where you are building an application that has no need to connect to backend or legacy data. The Directly Integrated Single Channel application pattern extends this discussion to describe the situation where you need to access existing data on legacy or third-party systems. Although we do not implement the Directly Integrated Single Channel application pattern in this project, the discussions here are relevant.
- Patterns: Connecting Self-Service Applications to the Enterprise, SG24-6572-00

  This Redbook discusses the Self-Service::Directly Integrated Single Channel application pattern, which covers Web applications needing one or more point-to-point connections with backend applications.
- Self-Service Applications using IBM WebSphere Application Server V4.0 and IBM MQSeries Integrator, SG24-6160-01

  This Redbook focuses on the task of designing and implementing a self-service application using the Router application pattern and the Decomposition application pattern, as defined by the IBM Patterns for e-business.

  The Router application pattern provides intelligent routing from multiple clients to multiple backend applications using a hub-and-spoke architecture. The interaction between the user and the backend application is a one-to-one relation, meaning the user interacts with applications one at a time. The primary business logic resides in the backend tier. This book shows how to use IBM MQSeries Integrator and IBM WebSphere Application Server to implement a router type application.

  The Decomposition application pattern expands on the router pattern, providing all the features and functions of that pattern and adding recomposition/decomposition capability. It provides the ability to take a user request and decompose it into multiple requests to route to multiple backend applications. The responses are recomposed into a single response for the user. This action moves some of the business logic into the decomposition tier, but the primary business logic still resides in the back-end application tier. The decomposition and recomposition functions are illustrated in this book using IBM WebSphere Application Server, IBM MQSeries Integrator and the IBM MQSI Aggregator Plug-In. The JMS listener provided by WebSphere Application Server Enterprise Services is also illustrated in this example.

- Applying the Patterns for e-business to Domino and WebSphere Scenarios, SG24-6255-00

  This Redbook describes Application Integration patterns, and how they form Composite Patterns, together with one or more of the other Patterns for e-business. It looks at run-time patterns for Domino and WebSphere Application Server integration, and identifies Composite Patterns in action. Some of the patterns include Lotus Sametime and Tivoli Policy Director.

  A major part of the book describes three real-life scenarios where Patterns for e-business are applied, with Domino and WebSphere Application Server as part of the run-time topology. Starting from the business requirements phase, the book identifies and applies Business, Application, and Run-time patterns to get to the final run-time topology. It starts with a simple scenario, which becomes increasingly complex in later scenarios. It discusses technology options, as well as design and development guidelines.

- User-to-Business Pattern Using WebSphere Personalization Patterns for e-business Series, SG24-6213-00

  This Redbook describes what was once referred to as the User-to-Business Topology 7 Personalization Pattern. At the time the book was written, the pattern was an emerging pattern. It focuses on direct interactions between users and a business. The pattern helps guide the design of systems with a consolidated customer-centric view that you can exploit for sophisticated personalization and cross-selling opportunities.

  This Redbook provides examples and guidelines for the User-to-Business Topology 7 Personalization Pattern. It shows how the pattern works and documents the tasks required to build an example of the pattern.

- Mobile Applications with IBM WebSphere Everyplace Access Design and Development, SG24-6259-00

  This Redbook provides application designers and developers with a broad overview of mobile e-business application design and development using the WebSphere Everyplace Access V1R1 offering.

  The book gives an overview of the Patterns for e-business and shows how to use the Patterns in the mobile e-business environment. It also discusses the design and development guidelines for mobile e-business applications using the products bundled in the WebSphere Studio and Visual Age for Java offerings.

  This book provides detailed information about the Sample application, by discussing scenarios and implementing mobile applications exercising different techniques for several type of clients. It also provides detailed instructions for setting up the development and run-time environment for WebSphere Application Server, WebSphere Transcoding Publisher and WebSphere Voice Server together with the Sample shipped with the Redbook.

- Access Integration Pattern using IBM WebSphere Portal Server, SG24-6267-00

  This Redbook describes the Access Integration pattern, which is an emerging pattern that describes services and components commonly required to provide users with consistent, seamless, device-independent access to relevant applications and information.

This Redbook provides an example and guidelines for the Access Integration pattern. It shows how the Pattern works and documents the tasks required to build an example.

- WebSphere Commerce Suite V5.1 for iSeries, Implementation and Deployment Guide, REDP0159

  This Redpaper describes the great benefit of the IBM Framework for e-business, to develop applications on the Windows platform and then deploy the application to one of the many supported WebSphere Application Server platforms, without changing the application.

  This Redpaper introduces the Patterns for e-business and the Electronic Commerce composite pattern used for building e-commerce Web sites. The focus of the Redpaper is on the iSeries unique implementation, deployment and development considerations when using IBM WebSphere Commerce Suite V5.1, Pro Edition for iSeries.

  It includes detailed procedures for implementing WebSphere Commerce Suite V5.1, Pro Edition for iSeries in single-tier and multiple tier run-time environments. Advanced configuration instructions are provided for integrating WebSphere Payment Manager, enabling Secure Sockets Layer (SSL) for the HTTP Server, and configuring a secure VPN connection for Open Servlet Engine (OSE) Remote. Once the run-time environment is configured, there is a detailed description of the steps necessary to deploy a WebSphere Commerce Suite store to a WebSphere Commerce Suite V5.1, Pro Edition for iSeries run-time environment.
- e-commerce Patterns for z/Linux Using WebSphere Commerce Suite V5.1 Patterns for e-business series, REDP0411

  This Redpaper describes the installation, configuration and customization of IBM WebSphere Commerce Suite Pro Edition for Linux for e-server z900 and S/390. It is intended for both technicians who need to install and administer Commerce Suite, and for developers who need to design and customize e-commerce sites for deployment on IBM WebSphere Commerce Suite Pro Edition for Linux for e-server z900 and S/390.

  This Redpaper is part of the Patterns for e-business series and reuses information developed in the Redbook B2C e-commerce Composite pattern using WebSphere Commerce Suite V5.1, SG24-6180.
- Integrating WebSphere Commerce Suite With a backend Order Management Application, REDP0514

  This Redpaper describes a scenario that presents a fictitious manufacturing company called MANCO, which produces bicycle parts and accessories. MANCO uses the J.D. Edwards OneWorld application on an iSeries server for the Enterprise Resource Planning (ERP) system. MANCO wanted to take advantage of the Internet global connectivity to give its business customers a new way to buy products and to check order status while augmenting the manual sales processes with direct electronic sales to its business customers. MANCO requirements best fit the user-to-online buying business pattern, which is a subset of the user-to-business pattern.
- Connect for iSeries with WebSphere Commerce Suite: BtoB Enabling a WebSphere Commerce Suite Web Site, REDP0127

  This Redpaper describes another MANCO scenario, from another iSeries development team in the IBM Rochester laboratory.

  This scenario presents a fictitious manufacturing company called MANCO, which produces bicycle parts and accessories.

  The e-marketplace has emerged as a means of connecting buyers and suppliers. Suppliers look for opportunities to grow their business and expand their customer base. An e-marketplace provides that ability. Many suppliers have already invested in e-commerce solutions and would like to leverage their current solution to enter into the e-marketplace arena.

This Redpaper details the migration of the existing MANCO e-Commerce Web site to an e-marketplace-enabled Web site, by utilizing the WebSphere Commerce Suite and Connect for iSeries products.

- e-Marketplace Pattern using WebSphere Commerce Suite, MarketPlace Edition Patterns for e-business Series, SG24-6158-00

  This Redbook describes the Business to Business e-Marketplace Pattern, which supports the development of e-Marketplace hub applications that bring multiple buyers and sellers together for efficient electronic trading of goods and services. Subsets of the application topologies for the Business to Business e-Marketplace Pattern are used to describe different parts of the full marketplace topology, and they represent increasing levels of complexity, functionality and integration in the topology, ranging from a simple e-Marketplace to a fully integrated e-Marketplace.
- Business-to-Business Integration Using MQSeries and MQSI, Patterns for e-business Series, SG24-6010-00

  This Redbook describes what was known as Business-to-Business Integration patterns two and three, which form the basis for many complex and more fully functioned B2B patterns. It is relevant to all enterprises dealing with partner integration issues over the Internet.

  Application topology 2 describes a scenario in which messages are being passed between two enterprise applications and no routing is performed. Topology 3 extends topology 2 to describe the scenario where routing is required for multiple cross enterprise applications to communicate.
- Business Process Management using MQSeries and Partner Agreement Manager, SG24-6166-00

  This Redbook describes business process management.

  Business process management is the answer for addressing the business issues of the economic world. The velocity of change, driven by the dynamics of commerce and e-business, force you to have an IT system that is ready to change rapidly, and continually. Another issue is the integration of the Internet within business processes and more general multichannel delivery. Companies must be able to provide a consistent service across all channels. To achieve this, you need to have an integrated business view. Business Process Management is the externalization and formalization of knowledge and expertise within applications and minds. That externalization makes it possible to stay in control of your business.

  This Redbook introduces the concepts of business process management and its relationship with business-to-business technologies. In the second part of the book, it explores an example of a business process built in MQSeries Workflow. The third part of the book extends this intra-enterprise business process to include collaboration with external companies using WebSphere Business-to-Business Partner Agreement Manager.

  The final part of the book takes one step back and looks in more general terms at the patterns of developing and deploying business-to-business solutions.
- Design for Scalability - An Update

  This White paper is from the IBM High Volume Web Sites team. The White paper describes component selection and management techniques you can use to make your Web site ready to adapt to increasing traffic. These techniques are the product of IBM experiences while working with customers seeking to improve the performance and availability of some of the largest Web sites in the world.

Abstract: Optimizing for scalability remains a significant challenge for e-businesses as they balance the demands for availability, reliability, security, and high performance. Vendors are responding with infrastructure options and supporting hardware and software platforms that address these requirements. This update identifies current products and emerging trends that are most likely to improve the scalability of your e-business infrastructure.

- IBM WebSphere V4.0 Advanced Edition Handbook

  This Redbook describes base application topologies and product mappings for WebSphere Application Server. Refer to the IBM Redbooks Web site for the latest update.
- The User centered design (UCD) for different project types, part 1

  This Web page is the first of two articles posted to the IBM develperWorks domain that describes useful application design activities for different types of projects.
- The User centered design (UCD) for different project types, part 2

  This Web page is the latest of two articles that describes design activities that IBM scientists have found most useful in various types of projects. This article defines user interface design elements, including the design prototype, use case model, and design specification document.

Programming model and decisions

- Designing e-business Solutions for Performance

  This White paper describes how the design or implementation of an e-business application can affect performance.
- Managing Web Site Performance

  This White paper contains tips and techniques for developers building applications that use session persistence. It also helps administrators to tune the WebSphere Application Server product appropriately for these applications.

Programming instructions and examples

- IBM developerWorks

  IBM developerWorks contains many excellent resources for developers, including tutorials on Web development-related topics. There is an excellent tutorial on the JDBC API.
- IBM Redbooks

  The IBM Redbooks site contains many WebSphere Application Server related documents.
- Servlets and JavaServer Pages - A Tutorial

  Tutorial from the author of Core Servlets and JavaServer Pages.

Programming specifications

- J2EE information

  For more information about J2EE specifications, visit the Sun site.
- sun.net.inetaddr.ttl property

  The following Java 2 SDK, Standard Edition 1.4 Web site describes the private sun.net.inetaddr.ttl property, which works in both Java 2 SDK, Standard Edition 1.3 (WebSphere Application Server V5.0.0, V5.0.1, and V5.0.2) and Java 2 SDK, Standard Edition 1.4.
- java.net.URLConnection class

  The *Networking* section of this Java 2 SDK, Standard Edition 1.4 Web site describes a change in the behavior of the java.net.URLConnection class.

Administration

- Best Practices Zone on WSDD

  The WebSphere Best Practices Zone is a collection of best practices for administering WebSphere Application Server. Over time, the zone is intended to grow to include best practices for using other WebSphere software products, and to cover more topics. Use the feedback mechanism to submit your best practice suggestions.
- The IBM Glossary of Computing Terms

  This glossary defines technical terms used in many IBM products. It is not a comprehensive resource of all IBM computing terms. This resource is provided for information purposes only and is updated periodically. IBM takes no responsibility for the accuracy of the information it contains.

Support

- AIX Fix Distribution Service Web site

  A Web facility for downloading AIX Version 4 and AIX Version 3 fixes, with a limited search engine designed with the assumption that you know what fix you need. If you do not know what fix you need, there is a pointer at the Web site to the APAR Database Facility. You can also contact your authorized IBM business partner or IBM Support Center.
- Ten Steps to Getting Support for WebSphere Application Server

  If you are new to a product, you might have difficulty finding all the information you need. And if you come across a problem, where do you go for help? Whether you are a new user looking for introductory information, or an experienced user looking for a workaround for a specific defect, you can benefit immediately from extensive Web-based support from IBM. It enables you to download fix packs, search on keywords, look up FAQs, Hints and Tips, and so forth. Always use this Web resource before contacting IBM Support directly.
- 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.

Chapter 2. Samples Gallery

The Samples Gallery offer a set of samples that demonstrates common Web application tasks. The samples are installed as a by-product of the install process, specifically the Customization Dialog, which generates the batch jobs that ultimately install the samples

The Samples Gallery includes the following samples:

- The Plants by WebSphere application, which demonstrates several J2EE functions, using an online store that specializes in plant and garden tool sales.
- Technology Samples, which showcase enterprise beans, servlets, JavaServer Pages technology, message-driven beans, and J2EE application client.
- The Java Pet Store Application, which demonstrates J2EE technology, using an online pet store.
- The message-driven beans Samples demonstrate message-driven beans receiving messages from the Point-to-Point and Publish Subscribe messaging models. It also demonstrates Java Message Service (JMS) inside the client container.

Finding the Samples Gallery. Once the Samples are installed on your local machine, they are available to try out. You can locate them by pointing a browser at http://hostname:9080/WSsamples/. The default port is 9080. However, if you specified a different port assignment you will need to modify your URL. If you do not find the Samples on your localhost, confirm their installation and the port number for the internal HTTP server.

Client Samples. A separate Samples Gallery is available for the client Samples except on z/OS. There is a line mode client on z/OS which is used for the calculator client sample and the MDBSamples client. To view the client Samples, install the WebSphere Application Server client. The Client Samples Gallery demonstrates the following:

- J2EE application client.
- Java thin client.
- Applet client.
- ActiveX to EJB Bridge client.
- CORBA C++ SDK Client.

Code Examples. In addition to the Samples in the Samples Gallery, you can find other code examples in the InfoCenter by clicking **Quick reference > Examples** in the InfoCenter navigation.

Note: The Samples are for demonstration purposes only. The code provided is not intended to run in a secured production environment. The Samples support Java 2 Security, therefore the Samples implement policy-based access control that checks for permissions on protected system resources, such as file I/O. The Samples also support global security.

The Samples are not supported in a multi-server, clustered environment. Many of the Samples use Cloudscape as a persistent data store on the server. Only one instance of Cloudscape is supported per Java virtual machine (JVM). As a result, the second server in the node will fail to start the Sample applications, since an instance of Cloudscape has already been created with the first server in the node.

Additional WebSphere Application Server Samples are available on the IBM WebSphere Developer Domain.

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 Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, New York 10594 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
522 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

The following terms are trademarks of IBM Corporation in the United States, other countries, or both:

- AIX
- CICS
- Cloudscape
- DB2
- DFSMS
- Everyplace
- iSeries
- IBM
- IMS
- Informix
- iSeries
- Language Environment
- MQSeries
- MVS
- OS/390
- RACF
- Redbooks
- RMF
- SecureWay
- SupportPac
- ViaVoice
- VisualAge
- VTAM
- WebSphere
- z/OS
- zSeries

The term CORBA used throughout this book refers to Common Object Request Broker Architecture standards promulgated by the Object Management Group, Inc.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

The Duke logo is a trademark or registered trademark of Sun Microsystems, Inc. in the United States and other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product and service names may be trademarks or service marks of others.