



Administering WebSphere Process Server

Note

Before using this information, be sure to read the general information in "Notices" on page 325.

23 June 2006

This edition applies to version 6, release 0, modification 1 of WebSphere Process Server for z/OS (product number 5655-N53) and to all subsequent releases and modifications until otherwise indicated in new editions.

To send us your comments about this document, email doc-comments@us.ibm.com. We look forward to hearing from you.

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.

© Copyright International Business Machines Corporation 2006. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Administering WebSphere Process Server	1
Overview of administering WebSphere Process Server	1
The administrative console for WebSphere Process Server	3
Setting up the WebSphere Process Server administration environment roadmap	10
Setting up the administrative architecture	10
Configuring cell-wide settings	11
Administering WebSphere Process Server servers roadmap	12
Configuring WebSphere Process Server resources roadmap	14
Configuring WebSphere Process Server settings roadmap	15
Managing Events Service	16
Administering the relationship service	16
Creating a clustered environment	19
Preparing a server or cluster to support applications	23
Administering Business Process Choreographer	29
Using the administrative console to administer Business Process Choreographer	29
Using scripts to administer Business Process Choreographer	36
Administering applications and application services	44
Administering business processes and human tasks	44
Business rules	60
Administering enterprise applications	85
Application Scheduler	86
Administering WebSphere Process Server resources	91
Administering adapters	91
Selector component administration overview	135
Considerations for modules containing business rules and selectors	139
Overview of targets	141
Administering extended messaging resources: Overview	144
Administering WebSphere ESB	151
Managing the bus environment	153
Managing service applications	188
Administering relationships	222
Viewing relationship types	222
Viewing relationship details	223
Querying relationship instances	223
Viewing relationship instance details	224
Editing relationship instance details	225
Creating new relationship instances	225
Deleting relationship instances	226
Rolling back relationship instance data	226
Viewing role types associated with a relationship	226
Querying relationship instances based on roles	227
Viewing role details	228
Viewing role instance details	228
Creating new role instances	229
Deleting role instances	229
Editing role instance properties	230
Tutorial: Relationship manager administration	230
Managing WebSphere Process Server failed events	232
Role-based access for failed event manager	233
Finding failed events	233
Working with data in failed events	238
Resubmitting failed events	241
Deleting failed events	242
Using the Common Event Infrastructure	242
Administering the Common Event Infrastructure	243
Troubleshooting WebSphere Process Server administration	296

Troubleshooting the failed event manager	297
Troubleshooting the Business Process Choreographer configuration	298
Troubleshooting business process and human tasks.	302
Troubleshooting business rules manager	321
Troubleshooting the Common Base Event browser	322
Notices	325
Programming interface information	327
Trademarks and service marks	327

Administering WebSphere Process Server

The topics in this section describe how to administer the WebSphere Process Server runtime environment, including the applications and resources deployed in the environment.

Additional resources for learning about WebSphere Process Server administration are available on the IBM Education Assistant web site under the **WebSphere Business Process Management** module.

Overview of administering WebSphere Process Server

Administering WebSphere® Process Server involves preparing, monitoring, and modifying the environment into which applications and resources are deployed, as well as working with the applications and resources themselves.

The administrative interfaces

WebSphere Process Server offers several interfaces for administering the runtime environment:

The administrative console

The administrative console is a browser-based interface that enables users to monitor, update, stop, and start a wide variety of applications, services, and resources. The administrative console can also be used to work with relationships and to locate and resolve failed WebSphere Process Server events.

The administrative console extends to provide administration capabilities for WebSphere Application Server, as well as other customer-defined products.

Business Process Choreographer Explorer

Business Process Choreographer Explorer is a stand-alone Web application that provides a basic set of administration functions for managing business process and human tasks. You can view information about process templates, process instances, task instances, and their associated objects. You can also act on these objects; for example, you can start new process instances, repair and restart failed activities, manage work items, and delete completed process instances and task instances.

Scripting (wsadmin)

The WebSphere administrative (wsadmin) scripting program is a non-graphical command interpreter environment that enables you to run administrative options in a scripting language and to submit scripting language programs for execution. It supports the same tasks as the administrative console. The wsadmin tool is intended for production environments and unattended operations.

See **Reference > Scripting interface** in this information center for details about the scripting tools.

Command-line tools

Command-line tools are simple programs that you run from an operating

system command-line prompt to perform specific tasks. Using these tools, you can start and stop application servers, check server status, add or remove nodes, and other tasks.

The WebSphere Process Server command-line tools include the serviceDeploy command, which processes .jar, .ear, .war and .rar files exported from a WebSphere Integration Developer environment and prepares them for installation to the production server.

Administrative programs

A set of Java classes and methods under the Java Management Extensions (JMX) specification provide support for administering Service Component Architecture (SCA) and business objects. Each programming interface includes a description of its purpose, an example that demonstrates how to use the interface or class, and references to the individual method descriptions.

See **Reference > Programming interfaces** in this information center for details about the programming interfaces.

Configuration information

Most configuration data for WebSphere Process Server is stored in XML files, which are kept in directories in the configuration repository tree (the master repository). The directory in which a configuration file exists determines its scope, or how broadly or narrowly that data applies.

- Files in an individual server directory apply to only that server.
- Files in a node-level directory apply to every server on that node.
- Files in a cell directory apply to every server on every node within the entire cell.

Table 1. WebSphere Process Server configuration files

Configuration file	Description
server-wbi.xml	Identifies a process server and its components, including Adaptive Entity Service, Extended Messaging Service, and WebSphere Business Integration Adapter Service configuration.
resources-wbi.xml	Defines operating environment resources for WebSphere Process Server and is present at the cell, node, and server scopes. This includes Extended Messaging Providers and WebSphere Business Integration Adapters.
cell-wbi.xml	Identifies a cell. This file is used to store the Relationship Service configuration, and is only present at the cell scope.
server-bpc.xml	Identifies a Business Process Choreographer container and its components, which include the Business Flow Manager, Human Task Manager, Staff Service, and Service Reference Service.
resources-bpc.xml	Defines operating environment resources for a Business Process Choreographer container, including configuration information for Staff Plugin Providers. This file is present at the cell, node, and server scopes.

Table 1. WebSphere Process Server configuration files (continued)

Configuration file	Description
deployment-bpc.xml	Configures application deployment settings for a business process container.
server-core.xml	Identifies configuration information for core WebSphere Process Server configurations, including the Artifact Loader Service, Events Service, and Business Context Data Service.

WebSphere Process Server configuration files can be edited through the administrative console, wsadmin, and scripting. No manual editing is required.

See the WebSphere Application Server for z/OS Information Center for complete information about server configuration files.

The administrative console for WebSphere Process Server

The administrative console is a browser-based interface used to administer WebSphere Process Server applications, services, and other resources. It can be used to administer at a cell, node, or server scope, and is available from both stand-alone process servers and deployment managers that manage all servers in a cell in a networked environment.

Note: The WebSphere Process Server administrative console is part of the larger WebSphere Application Server administrative console. As a result, many administrative tasks (for example, setting security, viewing logs, and installing applications) are the same for both WebSphere Process Server and WebSphere Application Server. Those tasks are documented in the WebSphere Application Server for z/OS Information Center.

Understanding the WebSphere Process Server tasks associated with the console

Common WebSphere Process Server tasks that are performed in the console include:

- Setting up the administrative architecture and environment
- Configuring process servers and their settings
- Deploying new applications to a server
- Managing existing applications and configurations
- Managing resource providers for applications
- Managing server resources, such as relationships, business processes, tasks, adapters, and selectors
- Administering the Business Process Choreographer
- Managing failed events on the process server
- Configuring product security
- Collecting data for troubleshooting purposes

Understanding the administrative console interface

The administrative console has three distinct areas:

Task bar

The task bar is located at the very top of the console. It provides options for logging out of the console, accessing product information, and accessing support.

Navigation tree

The navigation tree is on the left side of the console. It provides links to console pages that you use to create and administer servers, applications, and other resources.

Click the plus sign (+) beside an item in the navigation tree to expand it, or click the minus sign (-) to collapse the item. You can also click the item itself to toggle between its expanded and collapsed state.

Workspace

The workspace is located on the right side of the console. It displays pages that you use to create and administer servers, applications, and other resources. You access these pages by clicking the links in the navigation tree, or by clicking links within the workspace pages themselves.

See “Administrative console pages” on page 8 for a discussion of the types of pages that are displayed in the workspace.

On the far right side of the workspace is the help portal. It provides brief information about each field on the current page, as well as a link to more detailed information in the help browser.

Locating WebSphere Process Server-specific areas of the administrative console

WebSphere Process Server resources are grouped into several areas of the administrative console. Use the navigation tree to locate these resources, as follows:

- **Integration Applications**—Provides access to the following:
 - Failed event manager
 - Relationship manager
 - Common Base Event Browser
- **Resources** —Provides access to the following:
 - WebSphere Business Integration Adapters
 - Common Event Infrastructure Provider
 - Staff plug-in provider
 - Extended Messaging Provider
- **Servers > Application servers > *server_name***— Provides access to the following:
 - Container settings for business processes and human tasks
 - Application Scheduler
 - Business rules
 - Events service
 - Extended Messaging Service
 - Selectors
 - Staff service
 - WebSphere Business Integration Adapter Service
 - Web service reference service

Accessing online help from the administrative console

The administrative console provides online help for each page and field. Access the help in one of the following ways:

- Click **Help** from the console task bar to view online help in a new Web browser. From the help browser, you can do the following:
 - Browse for the topic you want to view in the Index tab. Click the link for that topic to open it in the right panel of the browser.
 - Search for a topic by specifying one or more key words in the Search tab. All matching topics are displayed in the navigation tree; click a topic link to view it.
- Place the cursor over a field to view hover help about that field.
- Place the cursor over a field and wait for the question mark (?) icon to appear. When the icon appears, click the field name to display brief help about it in the help portal (the right-most panel in the workspace).

If you want to view extended information about the field, or about the entire page and its associated tasks, click the **More information about this page** link at the bottom of the help portal.

Getting started with the administrative console

The following list of tasks can help you get started using the administrative console to manage and administer WebSphere Process Server resources.

- **Start the server for the administrative console.**

The startServer command reads the configuration file for the specified application server and starts the server. See the WebSphere Application Server for z/OS Information Center for details on using startServer.
- **Start the administrative console.**

See “Starting and stopping the administrative console” on page 6 for details.
- **Specify console preferences.**

Preferences control how data is displayed in the administrative console, as well as how the workspace behaves. See “Setting administrative console preferences” on page 7.
- **Set the console scope.**

The scope specifies the level at which a resource is visible on the administrative console. A resource can be visible in a console collection table at the cell, node, cluster, or server scope. See the WebSphere Application Server for z/OS Information Center for details on setting the scope.
- **Create filters to view information.**

Filters specify which data is shown in a column on a collection page. See “Setting administrative console filters” on page 7.
- **Optional: Set the session timeout for the console.**

By default, a console session times out after 30 minutes of inactivity. You can change this value by editing the deployment.xml configuration file, as described in the WebSphere Application Server for z/OS Information Center.
- **Save your work to the master repository.**

Until you save your changes to the master repository, the console uses a local workspace to track the changes. To save your changes, click **System Administration > Save Changes to Master Repository** to display the Save page, and then click **Save**.

Starting and stopping the administrative console

To access the administrative console, you must start it and then log in. After you finish working in the console, save your work and log out.

Perform the following steps to start and stop the console:

1. Start the administrative console:
 - a. Distributed platforms: Verify that the administrative console runs on the server1 application server for the WebSphere base product.
 - b. Enable cookies in the Web browser that you plan to use to access the administrative console.
 - c. Distributed platforms: In your cookie-enabled Web browser, type the following: `http://your_fully_qualified_server_name:9060/ibm/console` where *your_fully_qualified_server_name* specifies the fully qualified host name for the machine that contains the administrative server. When the administrative console is on the local machine, *your_fully_qualified_server_name* can be `localhost` unless security is enabled. On Windows platforms, use the actual host name if `localhost` is not recognized.

If security is enabled, your request is redirected to `https://your_fully_qualified_server_name:9043/ibm/console`, where *your_fully_qualified_server_name* is the fully qualified host name for the machine that contains the administrative server.

Note: The console uses the ports 9060 and 9043 by default. The port numbers for your actual installation can vary.

The administrative console loads in the browser, displaying a login page.

2. Log into the console:
 - a. In the **User ID** field, enter your user name or user ID.

Note: If you enter an ID that is already in use (and in session) you are prompted to do one of the following:
 - Force the existing user out of the session. The configuration file for the existing user ID is saved in the temporary area.
 - Wait for the existing user ID to log out or time out of the session.
 - Specify a different user ID.

The user ID lasts only for the duration of the session for which it was used to log in. Any changes made to server configurations are saved to the user ID. Server configurations are also saved to the user ID if a session times out.

- b. If the console uses global security, you must also enter a password in the **Password** field.
 - c. Click **OK**.
The administrative console now displays the Welcome page.

3. Stop the console:
 - a. Click **System administration > Save changes to master repository > Save** to save all work you have done during this session.
 - b. Click **OK**.

Your changes are saved to the master repository and the administrative console closes.

Setting administrative console preferences

The display of data on a collection page (a page that lists collections of data or resources in a table) can be customized through administrative console preferences. Preferences are set on a user level, and typically must be set separately for each area of the administrative console.

You can set the following display preferences for collection pages:

- **Maximum rows**—Specifies the maximum number of rows that are displayed when the collection is large. If there are more rows than the specified maximum, they are displayed on subsequent pages. The default value is 20.
- **Retain filter criteria**—Specifies whether the last search criteria entered in the filter function is retained. If this is enabled, the console collection pages initially use the retained filter criteria to display the data in the table following the preferences. See “Setting administrative console filters” for more information.
- **Max result set size**—Specifies the maximum number of resources that a search can return. The default value is 500.
- **Max column width**—Specifies the maximum number of characters viewable in a collection column. The default value is 18.

Perform the following steps to set display preferences for a collection page:

1. From any collection page, click **Preferences**.
The page expands to display the preference fields.
2. Modify the values for the **Maximum rows**, **Retain filter criteria**, **Max result set size**, and **Maximum column width** fields as desired.
3. Click **Apply**.
The collection table is refreshed to display according to the values you specified.

You can also set global administrative console preferences, such as whether the workspace is automatically refreshed and which scope to use by default. To access the Preferences page in the administrative console, click **System administration > Console settings > Preferences**. See the WebSphere Application Server Information Center for documentation on setting these preferences.

Setting administrative console filters

Each table on a collection page in the administrative console displays a list of WebSphere Process Server data or resources. You can use a filter to specify exactly which resources or data to display in a particular column of the table. Filters can be set on a single column only.

1. From the buttons at the top of the table, click **Filter the view**.
The filter dialog box opens above the top row of the table.
2. Use the **Filter** drop-down menu to select the column you want to include in the filter.
3. In the **Search term(s)** field, specify the filter criteria.

The criteria is a string that must be found in the name of a table entry in order for it to be displayed. The string can contain the percent sign (%), asterisk (*), or question mark (?) symbols as wildcard characters. For example, on the Resource Adapters page, you can enter *JMS* as the filter criteria for the Name column to find any resource adapter whose name contains the string JMS.

Prefix each of the following characters that appear as part of the string with a backslash (\) so that the regular expression engine performing the search correctly matches the search criteria: () ^ * % { } \ + & .

For example, if you want to search for all Java DataBase (JDBC) providers containing (XA) in the provider name, specify the following string in the Search term(s) field:

\XA

4. Click **Go**.

The table refreshes, and only those items in the selected column that meet the filter criteria are displayed.

Administrative console pages

Administrative console pages are formatted in one of three ways: Collection, detail, and wizard pages. Understanding the layout and behavior of each type of page can help you use them more effectively.

Collection pages

A collection page manages a collection of existing administrative objects (for example, relationships, failed events, or resource adapters). It contains one or more of the following elements:

Scope and Preferences

The scope and preferences help determine which administrative objects are displayed in the table, and how they should appear.

Table of existing objects

The table displays existing administrative objects of the type specified by the collection page. The table columns summarize the values of the key settings for these objects. If no objects exist yet, the table is empty. Use the available buttons to create a new object.

Buttons for performing actions

The typical buttons are described in “Administrative console buttons” on page 9. In most cases, you need to select one or more objects in the collection table, then click a button. The action is applied to all selected objects.

Sorting toggle buttons

After each column heading in the table are icons to sort the entries in ascending (^) or descending (v) order. By default, items such as object names are sorted in descending order (alphabetically).

Detail pages

A detail page is used to view details about an object and to configure specific objects (such as an application server or a listener port extension). It typically contains one or more of the following elements:

Configuration tabbed page

This tabbed page is used to modify the configuration of an administrative object. Each configuration page has a set of general properties specific to the object. Additional properties can be displayed on the page, depending on the type of administrative object you are configuring.

Runtime tabbed page

This tabbed page displays the configuration that is currently in use for the administrative object. It is most often read-only. Note that some detail pages do not have runtime tabs.

Local topology tabbed page

This tabbed page displays the topology that is currently in use for the

administrative object. View the topology by expanding and collapsing the different levels of the topology. Note that some detail pages do not have local topology tabs.

Buttons for performing actions

Buttons to perform specific actions display only on configuration tabbed pages and runtime tabbed pages. The typical buttons are described in “Administrative console buttons.”

Wizard pages

Wizard pages help you complete a configuration process comprised of several steps. Be aware that wizards can show or hide certain steps, depending on the characteristics of the specific object you are configuring.

Administrative console buttons

The administrative console interface contains a number of buttons, depending on which page you are currently viewing. This topic describes the available console buttons.

The following graphical buttons are located at the top of a table that displays WebSphere Process Server resources:

Button	Resulting action
Check all	Selects each resource (for example, a failed event or a relationship instance) that is listed in the table, in preparation for performing an action against those resources.
Uncheck all	Clears all selected resources so that no action is performed against them.
Show the filter view	Opens a dialog box to set a filter. Filters are used to specify a subset of resources to view in the table. See “Setting administrative console filters” on page 7.
Hide the filter view	Hides the dialog box used to set a filter.
Clear filter value	Clears all changes made to the filter and restores the most recently saved values.

The following buttons appear at the bottom of a WebSphere Process Server administrative console page. Not all buttons appear on all pages.

Add Adds the selected or typed item to a list, or produces a dialog box for adding an item to a list.

Apply Saves your changes to a page without exiting the page.

Back Displays the previous page or item in a sequence. The administrative console does not support using the Back and Forward options in the web browser, which can cause intermittent problems. Use the **Back** or **Cancel** buttons in the console instead.

Cancel Exits the current page or dialog box, discarding all unsaved changes. The administrative console does not support using the Back and Forward options in the web browser, which can cause intermittent problems. Use the **Back** or **Cancel** buttons in the console instead.

Clear Clears your changes and restores the most recently saved values.

Clear selections

Clears any selected cells in the tables on this tabbed page.

Close Exits the dialog.

Delete Removes the selected instance.

OK Saves your changes and exits the page.

Reset Clears your changes on the tab or page and restores the most recently saved values.

Save Saves the changes in your local configuration to the master configuration.

For a complete list of buttons used in the administrative console (for administering both WebSphere Application Server and WebSphere Process Server resources), refer to the WebSphere Application Server for z/OS Information Center.

Setting up the WebSphere Process Server administration environment roadmap

Specific tasks are needed to ensure that the WebSphere Process Server administration environment is set up correctly.

After properly installing the WebSphere Process Server, you must set up your administrative and server environment. Table 2 lists the tasks involved in setting up this environment.

Table 2. Tasks involved in setting up the WebSphere Process Server administration environment

Task	Described in
Configure the product	Configuring the product after installation
Configure ports	Avoiding port conflicts and Port number settings
Set up the administrative architecture	Setting up the administrative architecture
Configure cell-wide settings	Configuring cell-wide settings
Modify the server configuration files	Working with server configuration files
Administer servers and the network deployment environment	Administering process servers
Configure WebSphere Process Server resources	Configuring WebSphere Process Server resources
Deploy and manage applications	Deploying and administering applications and Overview of preparing and installing modules
Administer WebSphere Process Server services	Administering business processes and human tasks, Business rules, Administering enterprise applications, and Configuring application scheduler

Setting up the administrative architecture

How to set up the services, deployment managers, and other portions of the environment that control the workflow through the servers that comprise your processing environment.

This task assumes that you have already:

1. Unloaded the product from the tape.
2. Created the product definitions by running the install script
3. Configured the product
4. Configured ports (see Avoiding port conflicts and Port number settings)

You may want to review the tasks described in Setting up administrative architecture in the WebSphere Application Server for z/OS information center before beginning.

After you install and set up WebSphere Process Server, you need to monitor and control incorporated nodes and the resources on those nodes by using the administrative console.

1. Configure administrative services using Administration service settings
2. Install an instance of the database product
3. Create a WebSphere Process Server database in that database instance.
4. Set up tables in the WebSphere Process Server database.
5. Configure cells
6. Configure deployment managers
7. On the administrative console select **Environment > WebSphere Variables > (the scope of node) > DatabaseName_JDBC_DRIVER_PATH** to define where the WebSphere Process Server database resides

Using one of the following variables to define the database:

- For DB2: DB2UNIVERSAL_JDBC_DRIVER_PATH
- For Cloudscape: not supported for distributed environments

For more information, see Configuring WebSphere variables.

8. Manage nodes
9. Manage node agents
10. Manage node groups
11. Configure remote file services

Your WebSphere Process Server environment is now ready to process work.

Install applications.

Configuring cell-wide settings

To assist in handling requests among Web applications, Web containers, and application servers, you can configure cell-wide settings for virtual hosts and variables.

This task assumes you have already set up the administrative architecture.

When you are establishing a network deployment environment or are planning to install applications in a cluster you also have to configure specific settings across the cell.

- Configure virtual hosts
- Configure variables

The variable that defines the WebSphere Process Server database must be configured with a scope of node or cell. Use one of the following for the variable *DatabaseName_JDBC_DRIVER_PATH*:

- For DB2: DB2UNIVERSAL_JDBC_DRIVER_PATH
- For Cloudscape: not supported for distributed environments

Administering WebSphere Process Server servers roadmap

WebSphere Process Server server configuration provides settings that control how a server provides services for running enterprise applications and their components. Administrators can create and configure servers in an existing application server environment.

WebSphere Process Server administrators can configure one or more servers and perform tasks such as the following:

Table 3. Server administration tasks

Task	Described in
Create servers	Creating servers
Manage application servers	Managing application servers
Configure transport chains	Configuring transport chains
Develop custom services	Developing custom services
Define processes for the application server	Defining application server processes
Configure the Java™ Virtual Machine	Configuring the JVM

Creating servers

For WebSphere Process Server, you can create servers using either the wsadmin createApplicationServer command or the **Create New Application Server** wizard in the administrative console.

This task assumes that you have installed and configured the product.

Create servers to provide a processing environment for WebSphere Process Server applications.

Choose the method and complete the steps for that method

Method	Steps to complete
<p>Create New Application Server wizard</p>	<ol style="list-style-type: none"> 1. In the administrative console Application Servers page, click New to open the Create New Application Server page 2. Define your server by doing the following: <ol style="list-style-type: none"> a. Select a node for the server. b. Type a name for the server. The name must be unique within the node. c. Select the defaultProcessServer template to create the server. You can also use an existing application server as a template. The new process server inherits all properties of the template server. 3. Select whether the new server will have unique ports for each HTTP transport. By default, this option is enabled. If you select this option, you might need to update the alias list for the virtual host that you plan to use with this server to contain these new port values. (This includes the host alias that you use for the Business Process Choreographer configuration in step 4. Business Process Choreographer is itself an application and requires host_alias management.) If you clear this option, ensure that the default port values do not conflict with other servers on the same workstation. Note: If you create the server using an existing application server as a model, do not select to map applications from the existing server to the new server. By default, this option is disabled. 4. Optional: Configure Business Process Choreographer, if the server will run applications that contain business processes or human tasks. See <i>Configuring Business Process Choreographer</i> for more information. 5. Install the SchedulerCalendars application on one or more servers as described in <i>Configuring Business Process Choreographer</i>
<p>wsadmin createApplicationServer command</p>	<p>See the createApplicationServer command name in <i>Commands for the AdminTask object</i></p>

Managing application servers roadmap

You can use the **Application Servers** panel of the administrative console, or command line tools such as startServer and stopServer to manage servers. The administrative console interface is described.

To view information about an server, use the **Application Servers** panel on the administrative console.

Note: You can upgrade a portion of the nodes in a cell, while leaving others at the older release level. This means that, for a period of time, you may be managing servers that are at the current release and servers that are running the newer release in the same cell. In this mixed environment, there are some restrictions on what you can do with servers that are running the

older release level. There are no restrictions on what you can do with the servers that are running on the newer release level. For details, see *Creating servers*.

Table 4 lists the tasks to manage an application server.

Table 4. Server management tasks available from the administrative console

Task	How to accomplish
Display the available servers	Click Servers > Application Servers in the console navigation tree. The Application servers page lists the servers and the cell and the nodes holding the servers. It also shows the status of each server. The status indicates whether a server is started, stopped, or unavailable.
Display information for a particular server	Click a specific server name under Name . This accesses the Application server settings page for that server.
Create a process server	Creating servers
Start servers	Starting an application server
Monitor server operation	Detecting and handling problems with runtime components
Stop servers	Stopping an application server
Delete servers	<ol style="list-style-type: none"> 1. Click Servers > Application Servers in the console navigation tree to access the Application Servers page. 2. Select the check box beside a server to delete it. 3. Click Delete. 4. Click OK to confirm the deletion.

Configuring WebSphere Process Server resources roadmap

After installation, you must configure key resources for your WebSphere Process Server environment, such as application scheduler, business processes, Common Event Infrastructure, business rules, selectors and relationships.

An important step in completing your post-installation tasks for setting up the administrative environment is configuring your key resources. Use Table 5 to locate the information to configure the resources.

Table 5. Steps to configure resources

Resource	How to configure
Application scheduler	Configuring application scheduler
Business Process Choreographer	Configuring Business Process Choreographer
Common Event Infrastructure	Configuring the Common Event Infrastructure
Business rules	Configuring business rules
Selector components	Administering selector components
Relationship service	Administering the relationship service

Working with server configuration files

You can change the default locations of configuration files, as needed. Server configuration files define the available application servers, their configurations, and their contents.

To work with server configuration files in WebSphere Process Server, follow the instructions in the Working with server configuration files topic in the WebSphere Application Server Network Deployment, Version 6.0.x information center.

Installing and managing applications

After product installation, you must complete a number of post-installation tasks concerning the administration of applications and their environments.

After installation, you must administer applications and their environment, including customizing tasks, deploying applications and administering server environments.

- For more information, refer to Administering applications and their environment in the WebSphere Application Server Network Deployment, Version 6.0.x information center.
- For more information on installing applications on WebSphere Process Server, refer to Overview of preparing and installing modules.
- For information on stopping, starting and modifying applications, refer to Deploying and administering applications in the WebSphere Application Server Network Deployment, Version 6.0.x information center.

Administering applications and application services

This section describes how to use the administrative interfaces to administer WebSphere Process Server applications and application services, including business processes and tasks, business rules, and schedules.

Configuring WebSphere Process Server settings roadmap

Lists services within WebSphere Process Server that you configure after installing the software.

Upon successful installation of WebSphere Process Server, you must complete configuration by specifying settings for the various services you are using. Table 6 lists the services and the topics that describe how to configure the settings for each service.

Table 6. Services to configure

Service	Described in
Application scheduler	Configuring the Application Scheduler for a standalone server
Events service	Managing Events Service
Extended messaging service	Enabling the extended messaging service
WebSphere Business Integration Adapter service	Working with WebSphere Business Integration Adapters
Relationship service	Administering the relationship service

Managing Events Service

The Events Service configuration panel lists the properties set to ensure that information about WebSphere Process Server is automatically included in each event passed to the Events Infrastructure.

Start the administrative console. If you are unfamiliar with starting the administrative console or the steps to display the application servers, refer to the related topics on the administrative console. This task is performed at the administrative console.

When active, the events service ensures that any event that passes to the events infrastructure automatically includes information about a server. Use the **Events Service configuration panel** to set the events service configuration to on or off.

To configure the **Events Service Startup property** for an application server, complete the following steps:

1. Select the application server to configure.
In the navigation panel, click **Servers > Application servers**. Select the application server from the list.
The server properties display.
2. Display the Events Service configuration panel.
From the Business Integration properties, select **Events service**.
The **Events Service** configuration panel displays.
3. Select or deselect **Enable service** at startup field.
The default value is to enable the service automatically at startup. If this option is deselected and subsequent applications require this service to run, the system administrator must manually start the service. Optionally, the system administrator can restart this service by selecting the option and restarting the server.
4. Verify that the Java Naming and Directory Interface (JNDI) name is correct.
Review the **Events infrastructure emitter factory JNDI name** field. If you have not generated an alternative profile, it is recommended that you accept the default JNDI name.
5. Select any custom properties, if necessary.
Click the **Custom properties** link and select any custom properties from the listing.
6. Save your configuration.
Click **Apply** to save your configuration and remain on the **Events Service configuration** panel. Click **OK** to save your configuration and return to the Business Integration panel.
7. Apply your changes to the server.
Stop and restart the application server to apply your changes.

Administering the relationship service

The relationship service maintains relationships and roles in the system. It manages relationship and role definitions and metadata and enables the need to specify the definition of a relationship and manipulate the instances derived from the definition.

Relationships correlate identifiers from different environments for the same data. Participants in the relationship are distinguished by the roles they serve in the relationship.

Relationships and roles are described in definitions that you design through the graphical interface of WebSphere Integration Developer's relationship editor tool. The relationship definition is a template that describes what the relationship should look like, identifies each role, and specifies how the roles are related. The role definition captures the structure and constraint requirements for the participants. Each relationship definition has, as its constituents, role definitions. Relationship definitions are stored as XML files that are deployed as part of a J2EE application to a particular server.

For detailed background and task information on creating relationships, relationship types, and using the relationship editor, see the WebSphere Integration Developer Information Center.

At run time, when maps or other WebSphere Process Server components run and need a relationship instance, the instances of the relationships are either created or retrieved depending on the scenario. To facilitate the need to create and manipulate the relationship and role runtime instance data, the relationship service exposes a set of application programming interfaces (API's) to WebSphere Process Server's relationship manager tool. You interact with the relationships and roles through the relationship manager graphical interface.

The relationship and role instance data is saved in relationship tables that are stored on a specific server in the default data source that you specify when you configure the relationship service.

The relationship service runs on each server at the cell level. The **Relationship Manager** home page **About** section shows the number of servers in the cell that are running relationship services; the **Relationships** section shows each server name that is running relationship services. Before working with relationship instances, you need to select the server that has the instances of the relationships and roles you want to manage.

For detailed information on using the relationship manager, see the relationship manager and the administrative console in the WebSphere Process Server Information Center.

The following topics describe the configuration tasks to perform for the relationship services for your WebSphere Process Server environment.

Configuring the relationship service

After installing the product, you need to configure parameters for the relationship service.

Security role required: To perform this task, you must be logged in as a configurator or an administrator. Any WebSphere security role can view this configuration.

Perform this task to set the data source and page size parameters for the relationship service.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.

3. Click **Relationship Services configuration**.
The configuration tabbed page displays, showing the name and version (read-only) of the currently installed relationship service.
4. In the **Data Source** field, specify the default data source for the relationship service by entering the JNDI name of a data source defined at the cell level. This is where the tables for the relationship service are stored. All relationship-related schema is created in this data source by default.
5. In the **Page Size** field, specify the maximum number of results returned per page for a relationship manager search of relationship instances. The default value is 500.
6. You then have the following options:
 - Click **Apply** to save your changes and keep you on the page.
 - Click **OK** to save your changes and return to the previous page.
 - Click **Reset** to clear your changes and restore the most recently saved values.
 - Click **Cancel** to discard any unsaved changes on the page and return to the previous page.

Viewing relationships managed by the relationship service

Perform this task to view a list of the existing relationships that this relationship service manages.

Security role required: Any WebSphere security role can view this configuration.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Click **Relationship Services configuration > Relationships**.
The Relationship collection page displays. Each row shows the version and data source for the associated relationship.
4. To set display preferences for this collection page, click **Preferences**. Modify the field values, as desired, and click **Apply**.
5. To see the configuration properties for a relationship, click the relationship name in the relationship collection table.

Viewing relationship properties

Perform this task to view the configuration properties that the relationship service manages at both the relationship service level—as it applies to the relationship service—and at the individual relationship level—as it applies to individual relationships.

Security role required: Any WebSphere security role can view this configuration.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Click **Relationship Services configuration > Relationships**.
4. In the relationship collection table, click the name of the relationship whose properties you want to view.
The configuration tabbed page displays, showing the name, version, and data source currently in use for the relationship (read-only).

Note: The version is used for migration purposes. If the old relationship data needs to coexist in the new system, then the old infrastructure version will be set to the old version. Otherwise, it will be set to the current version.

5. To return to the relationship collection page, click **Back**.

Creating a clustered environment

Creating a clustered environment enhances the failover and scaling characteristics of WebSphere Process Server applications. To establish a clustered environment, you and your team will perform the described steps.

There are a number of things you should do before creating a clustered environment:

- Ensure that you have adequate resources to implement clustering successfully.
- Analyze the service applications you will deploy into the clustered environment. Some of the optional steps you perform depend on the needs of the service applications.
- Ensure that the application logic is tolerant of a clustered environment, for example:
 - The application tolerates partitioned queues when you make provisions for orphaned requests or for requests processed out of order. An application failure may create one or both of these situations.
 - There are no system-wide values that you maintain locally.
- Familiarize yourself with network deployment and clustering as described in the WebSphere Application Server for z/OS information center.
- Familiarize yourself with these instructions before performing any of the steps. An overall idea of the steps can help you proceed efficiently.
- Make sure WebSphere Application Server for z/OS has been installed. See Installing the product and additional software in the WebSphere Application Server for z/OS information center.
- Configure the WebSphere Application Server for z/OS to support a WebSphere Process Server for z/OS clustered environment.

This involves the following tasks

- Creating a Network Deployment cell as described in the WebSphere Application Server for z/OS information center.
- Creating an empty managed node as described in the WebSphere Application Server for z/OS information center.

Note: Do not Federate into Deployment manager. This means do not run the job named BBOWMNAN as part of the process for creating the empty managed node.

- Run the WebSphere Process Server for z/OS install script to create the product definitions.
See Running the install script for a description of this task.
- Create the databases and storage groups.
WebSphere Process Server for z/OS includes a sample file that you can use to create the database and storage groups.
See Creating databases and storage groups for a description of this task.
- Edit the deployment manager response file.

The deployment manager response file contains default property settings that you modify to suit your environment. Also, the values that you used when creating the database and storage groups must match their corresponding values in the response file. For more information, see *Working with response files*.

- Run the WebSphere Process Server for z/OS configuration script with the deployment manager response file.

This results in augmenting the default profile with WebSphere Process Server for z/OS network deployment configuration data.

Also, as a result of running the WebSphere Process Server for z/OS script, the DDL for the WebSphere Process Server for z/OS databases only are created. The databases for WebSphere ESB are not created.

See *Run the configuration script* for a description of this task

- Run the WebSphere Process Server for z/OS install script for the managed node. See *Running the install script* for a description of this task.
- Run the WebSphere Process Server for z/OS configuration script with the managed node response file.

This results in augmenting the default profile with WebSphere Process Server for z/OS managed node configuration data.

See *Run the configuration script* for a description of this task

- Federate the managed node into the Deployment manager.

Run the job BBOWMNAN. This is the job you did not run when you created the empty managed node originally.

Create a clustered environment when your application requires more capacity and availability than a single server provides. The clustered environment provides these additional benefits:

Workload balancing

By running application images on multiple servers, a cluster balances an application workload across the servers in the cluster.

Processing power for the application

You can add processing power to your application by configuring additional server hardware as cluster members supporting the application.

Application availability

When a server fails the application continues to process work on the other servers in the cluster thereby allowing recovery efforts to proceed without impacting application users.

Maintainability

You can stop a server for planned maintenance without stopping application processing.

Flexibility

You can add or remove capacity as needed by using the administrative console.

Note: A list of related tasks is included at the bottom of this page. The title of any related tasks for a particular step is contained as a parenthetical comment for that step. If you are unfamiliar with the task, see the related topic for additional information.

1. Design the clustered environment
 - a. Lay out the topology of the cell. Determine what physical and logical resources are needed for the cell.

- b. Decide whether to use databases or schemas for the various components of the cell.
- c. Decide if there is a need for monitored events and which servers should host the monitored events.

2. Create other databases needed by the cell, such as:

a. Create Environment variables Create the following:

- DB2UNIVERSAL_JDBC_DRIVER_PATH = /db2810/jcc/classes
- DB2UNIVERSAL_JDBC_DRIVER_NATIVEPATH = /db2810/jcc/lib
- Add JDBC JCC Properties file to DMGR/Server Adjunct and servant Custom Property

Application servers > server > Process Definition > Servant > JavaVirtualMachine > Custom Properties > New

db2.jcc.propertiesFile, for example, /shared/db2810/jcc/properties/DB2JccConfiguration.properties

Note: This property will be required for each server configured in the cluster.

- 3. **Optional:** Create the cluster that handles monitored events, if you need monitored events.
 - a. Create the cluster with cluster members using the default WebSphere Process Server template.
 - b. Using the cluster you are configuring as the target and the Common Event Infrastructure (CEI) database created in “Creating a clustered environment” on page 19, install the CEI application, and install the message-driven bean (MDB) application. (Configuring the Common Event Infrastructure)
- 4. **Optional:** Create the cluster that handles administration applications, if the application contains business rules or selectors and the application requires modification to the rules or selectors after you deploy the application.

Note: If you have determined that administration of business rules and selectors will be deployed to clusters that handle other WebSphere Process Server components, you can combine this step with either step 3 or step 6 on page 22.

- a. Create the cluster with cluster members using the default WebSphere Process Server Template.
 - b. Use the Advanced Configuration panel to deploy the business rules manager server.
5. Configure messaging for the cluster.

Note: If you have determined that administration messaging will be handled by clusters that handle other WebSphere Process Server components, you can combine this step with either step 3, step 4, or step 6 on page 22.

Important: This step must be completed before installing the first service application in the cluster.

- a. Create the cluster using the default WebSphere Process Server template.
- b. Select **Default Destination Location** on the Advanced Configuration panel.
- c. Add the cluster as a member of the Event bus.

Use the data source you created in step “Creating a clustered environment” on page 19. If in step 1b on page 21 you decided to use multiple schemas in a single database, select the correct schema when configuring the messaging engine data source.

- d. Add the cluster as a bus member of the Business Process Choreographer bus.

Use the data source you created in step “Creating a clustered environment” on page 19. If in step 1b on page 21 you decided to use multiple schemas in a single database, select the correct schema when configuring the messaging engine data source.

6. Create the cluster for service applications.
 - a. Create the cluster with cluster members using the default WebSphere Process Server template
 - b. Use the Advanced Configuration panel to select the messaging cluster. When combining this step with step 5 on page 21, select the default messaging cluster as stated in that step. Otherwise, select the correct option and choose the cluster you created in 5 on page 21.
 - c. **Optional:** Configure Business Process Choreographer support using the databases or schemas created in step “Creating a clustered environment” on page 19. (Configuring Business Process Choreographer)
Perform this step when an application contains Business Process Execution Language (BPEL) or business state machine components.
 - d. **Optional:** Use the Advanced Configuration panel to select the Java Naming and Directory Interface (JNDI) name of the emitter factory profile to associate the cluster with the emitter factory profile you created in step “Creating a clustered environment” on page 19.

When you start the servers and any installed applications, they will start normally.

You can now install applications on the cluster.

Related concepts

Setting up the application serving environment

Preparing a server or cluster to support service applications

Setting up the application serving environment

Preparing a server or cluster to support service applications

Related tasks

Unload the product code from the install media

Running the install script

Configuring the product

Federating custom nodes to a deployment manager

Configuring the Common Event Infrastructure

Configuring Business Process Choreographer

Installing a module on a production server

Unload the product code

Run the install script

Configure the product

Federating custom nodes to a deployment manager

Configuring Business Process Choreographer

Installing a module on a production server

Related information

Working with server configuration files
Learning about messaging engines
Learning about data stores
Learning about bus destinations
Learning about the default messaging provider
Working with server configuration files
Learning about messaging engines
Learning about data stores
Learning about bus destinations
Learning about the default messaging provider
Configuring the Common Event Infrastructure

Preparing a server or cluster to support applications

A server or cluster can be configured to be a deployment target for WebSphere Process Server Service Component Architecture (SCA) applications, to support WebSphere Process Server applications that have been deployed to another location, or both. Use the Advanced Configuration page in the administrative console to perform the necessary configuration tasks. The Advanced Configuration page is available for both the server scope and the cluster scope.

There are three basic scenarios for configuring a server or cluster to support WebSphere Process Server applications:

- The server or cluster supports applications deployed to another server or cluster—In this configuration, the server or cluster does not host WebSphere Process Server applications, though it can host the business rules manager or service integration bus members. The default configuration option is to select this scenario without configuring service integration bus members. See “Supporting WebSphere Process Server applications deployed on another server or cluster” on page 24 for specific information about this configuration.
- The server or cluster depends upon remote service integration bus members that are already configured with messaging engines—The server or cluster hosts WebSphere Process Server applications; the destinations can be hosted locally or remotely. It can also host the business rules manager. See “Supporting WebSphere Process Server applications without supporting bus members” on page 26 for specific information about this configuration.
- The server or cluster includes service integration bus members—The server or cluster hosts WebSphere Process Server applications and, optionally, the messaging engines and destinations required by those applications. These messaging engines can also host destinations required by applications with other deployment targets. Finally, the server or cluster can also host the business rules manager. See “Supporting WebSphere Process Server applications and bus members” on page 27 for specific information about this configuration.

When configuring any of the scenarios, you must consider where to locate the destinations (Java Message Service (JMS) queues) used by the applications. Destinations can be hosted on the same server or cluster as the applications themselves, or they can be hosted on a remote server or cluster. The Advanced Configuration page provides options for specifying where to host these destinations.

As part of the configuration process, you can also specify the following options:

- Install the business rules manager
The business rules manager is a Web-based tool used to modify and manage business rules on a single server or across all servers in a cluster.

Note: This option is available only if you have installed it as part of WebSphere Process Server.

- Route Common Event Infrastructure (CEI) emitted events
You can use the WebSphere Process Server monitoring capabilities to route any Common Base Events generated at runtime to the correct CEI server application for processing. In order to route CEI emitted events, you must specify the Java Naming and Directory Interface (JNDI) name of the event emitter factory profile. All applications deployed to the cluster or server use the same emitter profile. If you have already configured CEI, an emitter factory profile that points to the installed event server application exists and is listed on the Advanced Configuration page.

Supporting WebSphere Process Server applications deployed on another server or cluster

A cluster or server can be configured to support WebSphere Process Server service applications that have been deployed to another cluster or server. This is the default configuration scenario.

You must be logged in as administrator or configurator.

In this scenario, the cluster or server does not host WebSphere Process Server applications. However, it can host the business rules manager and JMS queues (destinations) used by applications running on remote servers or clusters. The JMS destinations are hosted in the messaging engines that execute in the server or cluster. One messaging engine is created by default when the Advanced Configuration page is used to add the server or cluster as a member of each system integration bus.

1. From within the administrative console, click one of the following, depending on your scope:
 - To configure a server: **Servers > Application Servers > *server_name*> Advanced Configuration**
 - To configure a cluster: **Servers > Clusters > *cluster_name*> Advanced Configuration**
2. If you want to install the business rules manager on this server or cluster, click **Install Business Rules Manager**.
3. Verify the value in the **Emitter Factory Profile JNDI Name** field.
 - If you want to change the default configuration for routing emitted events, select the appropriate emitter factory profile JNDI name from the field. Note that all applications deployed to the server or cluster use the same emitter factory profile.
 - If you want to use the default emitter factory profile, select **None** from the menu.
4. Ensure the **Do not configure to host SCA applications** option is selected.
5. If the server or cluster is going to host destinations for the applications, perform one of the following tasks:

Resource being configured	Steps to perform
Configuring a server	<ol style="list-style-type: none"> 1. Click Default Destination Location. 2. If you want to have the SI Bus service automatically create the data sources and databases using Cloudscape, click the Use default datasource values check box and proceed to Step 6 on page 26. 3. If you want to create a new data source and database, use the JDBC Provider menu to specify the appropriate JDBC provider template. 4. Optionally, in the Application bus schema name field, type the name of the database schema used to contain the tables for the application bus data source. 5. Optionally, in the System bus schema name field, type the name of the database schema used to contain the tables for the system bus data source. 6. In the Data source user name field, type the user name you are using to access the database. 7. In the Data source password field, type the password associated with the user name. 8. In the Application Bus Datasource Properties field, enter the custom properties required by the data source used by the SCA Application Bus. 9. In the System Bus Datasource Properties field, enter the custom properties required by the data source used by the SCA System Bus. 10. If you want the messaging engine to create the database tables for the data source, select Create tables. The tables are created when the server hosting the messaging engine is started.

Resource being configured	Steps to perform
Configuring a cluster	<ol style="list-style-type: none"> 1. Click Default Destination Location. 2. Use the JDBC Provider menu to specify the appropriate JDBC provider template. 3. Optionally, in the Application bus schema name field, type the name of the database schema used to contain the tables for the application bus data source. 4. Optionally, in the System bus schema name field, type the name of the database schema used to contain the tables for the system bus data source. 5. In the Data source user name field, type the user name you are using to access the database. 6. In the Data source password field, type the password associated with the user name. 7. In the Application Bus Datasource Properties field, enter the custom properties required by the data source used by the SCA Application Bus. 8. In the System Bus Datasource Properties field, enter the custom properties required by the data source used by the SCA System Bus. 9. If you want the messaging engine to create the database tables for the data source, select Create tables. The tables are created when the server hosting the messaging engine is started.

6. Click **OK**.

Supporting WebSphere Process Server applications without supporting bus members

A cluster or server can be configured as a deployment target for WebSphere Process Server applications. The destinations for the applications can be hosted on the same cluster or server as the applications or on a remote cluster or server. Note that when you set up a cluster or server to host applications, the SchedulerCalendar application is automatically installed as part of the configuration.

You must be logged in as administrator or configurator to perform the following tasks.

This topic describes how to host WebSphere Process Server applications on one server or cluster while hosting the applications' destinations on a remote server or cluster. For information on how to host destinations on the same cluster or server, see "Supporting WebSphere Process Server applications and bus members" on page 27.

1. From within the administrative console, click one of the following, depending on your scope:

- To configure a server: **Servers > Application Servers > *server_name*> Advanced Configuration**
 - To configure a cluster: **Servers > Clusters > *cluster_name*> Advanced Configuration**
2. If you want to install the business rules manager on this server or cluster, click **Install Business Rules Manager**.
 3. Verify the value in the **Emitter Factory Profile JNDI Name** field.
 - If you want to change the default configuration for routing emitted events, select the appropriate emitter factory profile JNDI name from the field. Note that all applications deployed to the server or cluster use the same emitter factory profile.
 - If you want to use the default emitter factory profile, select **None** from the menu.
 4. Clear the **Do not configure to host SCA applications** option.
 5. Specify the location for the destinations, as follows:
 - a. Click **Remote Destination Location**.
 - b. Use the associated menu to specify the name of the remote server or cluster that is going to host the destinations.
 6. Click **OK**.
 - If you are planning to use business processes defined with the Business Process Execution Language (BPEL) on this cluster or server, you must also use the Business Process Container wizard to perform the necessary configuration.
 - If you are planning to use applications that contain human tasks or if you plan to use the Business Process Choreographer Explorer in this cluster or server, you must also use the Human Task Container wizard to perform the necessary configuration.

Supporting WebSphere Process Server applications and bus members

A cluster or server can be configured to both host WebSphere Process Server applications and support WebSphere Process Server applications that are installed on remote clusters or servers. In this configuration scenario, the application buses are configured for the current cluster or server. In addition, the SchedulerCalendar application is automatically installed to provide support for hosting WebSphere Process Server applications. In this scenario, the destinations for the applications can be hosted on the same cluster or server as the applications themselves or on a remote location. This topic describes how to set up the server or cluster to host the destinations.

You must be logged in as administrator or configurator to perform the following tasks.

1. From within the administrative console, click one of the following, depending on your scope:
 - To configure a server: **Servers > Application Servers > *server_name*> Advanced Configuration**
 - To configure a cluster: **Servers > Clusters > *cluster_name*> Advanced Configuration**
2. If you want to install the business rules manager on this server or cluster, click **Install Business Rules Manager**.
3. Verify the value in the **Emitter Factory Profile JNDI Name** field.

- If you want to change the default configuration for routing emitted events, select the appropriate emitter factory profile JNDI name from the field. Note that all applications deployed to the server or cluster use the same emitter factory profile.
 - If you want to use the default emitter factory profile, select **None** from the menu.
4. Clear the **Do not configure to host SCA applications** option.
 5. To host destinations for the applications, perform one of the following tasks:

Resource being configured	Steps to perform
Configuring a server	<ol style="list-style-type: none"> 1. Click Default Destination Location. 2. If you want to have the SI Bus service automatically create the data sources and databases using Cloudscape, click the Use default datasource values check box and proceed to Step 6 on page 29. 3. If you want to create a new data source and database, use the JDBC Provider menu to specify the appropriate JDBC provider template. 4. Optionally, in the Application bus schema name field, type the name of the database schema used to contain the tables for the application bus data source. 5. Optionally, in the System bus schema name field, type the name of the database schema used to contain the tables for the system bus data source. 6. In the Data source user name field, type the user name you are using to access the database. 7. In the Data source password field, type the password associated with the user name. 8. In the Application Bus Datasource Properties field, enter the custom properties required by the data source used by the SCA Application Bus. 9. In the System Bus Datasource Properties field, enter the custom properties required by the data source used by the SCA System Bus. 10. If you want the messaging engine to create the database tables for the data source, select Create tables. The tables are created when the server hosting the messaging engine is started.

Resource being configured	Steps to perform
Configuring a cluster	<ol style="list-style-type: none"> 1. Click Default Destination Location. 2. Use the JDBC Provider menu to specify the appropriate JDBC provider template. 3. Optionally, in the Application bus schema name field, type the name of the database schema used to contain the tables for the application bus data source. 4. Optionally, in the System bus schema name field, type the name of the database schema used to contain the tables for the system bus data source. 5. In the Data source user name field, type the user name you are using to access the database. 6. In the Data source password field, type the password associated with the user name. 7. In the Application Bus Datasource Properties field, enter the custom properties required by the data source used by the SCA Application Bus. 8. In the System Bus Datasource Properties field, enter the custom properties required by the data source used by the SCA System Bus. 9. If you want the messaging engine to create the database tables for the data source, select Create tables. The tables are created when the server hosting the messaging engine is started.

6. Click **OK**.

- If you are planning to use business processes defined with the Business Process Execution Language (BPEL) on this cluster or server, you must also use the Business Process Container wizard to perform the necessary configuration.
- If you are planning to use applications that contain human tasks or if you plan to use the Business Process Choreographer Explorer in this cluster or server, you must also use the Human Task Container wizard to perform the necessary configuration.

Administering Business Process Choreographer

You can administer Business Process Choreographer using the administrative console or using scripts.

Using the administrative console to administer Business Process Choreographer

Describes the administrative actions that can be performed using the administrative console.

Administering the compensation service for a server

Use the administrative console to start the compensation service automatically, when the application starts, and to specify the location and maximum size of the recovery log.

The compensation service must be started on an application server, when business processes are run on that server. The compensation service is used to manage updates that might be made in a number of transactions before the process completes.

You can use the administrative console to view and change properties of the compensation service for application servers.

1. Display the administrative console.
2. In the navigation pane, click **Servers** → **Application servers** → *server_name*.
3. On the Configuration tab, under Container Settings, click **Container services** → **Compensation service**. This action displays a panel with the compensation service properties.

Enable service at server startup

Specifies that, whenever the application server starts, it automatically tries to start the compensation service.

Make sure that this check box is selected. The compensation service must be enabled when you use business processes. If you run your business processes on a cluster, you must enable the compensation service for each server in the cluster.

Recovery log directory

Specifies the name of a directory for this server where the compensation service stores log files for recovery. When compensation is used, the WebSphere product stores information that is required for compensation.

Recovery log file size

Specifies the maximum size, in megabytes, for compensation log files on this application server.

4. **Optional:** If necessary, change the compensation service properties.
5. Click **OK**.
6. To save your configuration, click **Save** in the Messages box of the administrative console window. Then click **Save** on the Application Servers Save pane.

Querying and replaying failed messages, using the administrative console

This describes how to check for and replay any messages for business processes or human tasks that could not be processed.

When a problem occurs while processing a message, it is moved to the retention queue or hold queue. This task describes how to determine whether any failed messages exist, and to send those messages to the internal queue again.

1. To check how many messages are on the hold and retention queues:
 - a. Click **Servers** → **Application servers** → *server_name*.
 - b. On the **Configuration** tab, in the Container Settings section, click one of the following sequences:

- For business processes: **Business process container settings** → **Runtime Configuration**
- For human tasks: **Human task container settings** → **Runtime Configuration**

The number of messages on the hold queue and retention queue are displayed under **General Properties**.

2. If either the hold queue or the retention queue contains messages, you can move the messages to the internal work queue.

Click one of the following options:

- For business processes: **Replay Hold Queue** or **Replay Retention Queue**
- For human tasks: **Replay Hold Queue**

Note: When security is enabled, the replay buttons are only visible to users who have operator authority.

Business Process Choreographer tries to service all replayed messages again.

Refreshing the failed message counts:

Use the administrative console to refresh the count of failed messages for business processes or human tasks.

The displayed number of messages on the hold queue and on the retention queue, and the number of message exceptions, remain static until refreshed. This task describes how to update and display the number of messages on those queues and the number of message exceptions.

1. Select the appropriate application server.

Click **Servers** → **Application servers** → *server_name*.

2. Refresh the message counts.

On the **Configuration** tab, in the Container Settings section, click one of the following sequences:

- For business processes: **Business process container settings** → **Runtime Configuration** → **Refresh Message Counts**
- For human tasks: **Human task container settings** → **Runtime Configuration** → **Refresh Message Count**

The following updated values are displayed under **General Properties**:

- For business processes: The number of messages on the hold queue and on the retention queue
- For human tasks: The number of messages on the hold queue
- If any exceptions occurred while accessing the queues, the message text is displayed in the Message Exceptions field.

Failed message handling and quiesce mode:

Business Process Choreographer provides a facility for handling temporary infrastructure failures.

This section describes how the business process container handles failed messages. This contrasts with the simpler mechanism used by the human task container, described in “Failed message handling for human tasks” on page 34.

Long-running processes consist of a sequence of transactions. The transactions are separated by Java Message Service (JMS) messages, which the server sends to a message-driven bean. This bean passes the incoming messages to the process server, for processing. Each transaction consists of the following actions:

- Receive a message.
- Navigate, on behalf of the message.
- Send messages that trigger follow-on transactions.

The server might fail to process a message received by the message-driven bean for either of the following reasons:

- A specified number of consecutive messages cannot be processed. The infrastructure is therefore assumed to be unavailable.
- Only some messages can be processed. Any single message that cannot be processed is assumed to be damaged.

The responses to these causes are as follows:

Cause	Response
Unavailable infrastructure	The message-driven bean tries, for a specified time, to recover from that situation. It tries to keep all messages available until the server is again operational. This problem might be caused by a database failure, for example.
Damaged message	After a specified number of retries, the message is put into the hold queue, where it can be manipulated or reviewed. From the hold queue, it can also be moved back to the input queue, to retry the transaction.

The implementation for messages for business processes is as follows:

- If a message fails to be processed, the server puts it into a retention queue, where it is kept available, in case this is an infrastructure problem that is fixed within a specified time.
- When a message is in the retention queue, the options are as follows:
 - When a subsequent message can be processed successfully, all messages from the retention queue are moved back to the input queue of the message-driven bean. For each message, a count is maintained of how often the message has been sent to the retention queue. If this count exceeds the retry limit for a given message, the message is put in the hold queue.
 - If the next message fails to be processed, it is also put in the retention queue. This process continues until the threshold of maximum messages in the retention queue is reached. When this threshold is reached, the message-driven bean moves all messages from the retention queue to the input queue and switches into quiesce mode.

When the message-driven bean operates in quiesce mode, it periodically tries to process a message. Messages that fail to be processed are put back in the input queue, without incrementing either the delivery count or the retention queue traversal count. As soon as a message can be processed successfully, the message-driven bean switches back into normal processing mode.

This facility consists of two numerical limits, two queues, quiesce mode, and the message retry behavior.

Retry limit

The retry limit defines the maximum number of times that a message can be transferred through the retention queue before being put on the hold queue.

To be put on the retention queue, the processing of a message must fail three times.

For example, if the retry limit is 5, a message must go through the retention queue five times (it must fail for $3 * 5 = 15$ times), before the last retry loop is started. If the last retry loop fails two more times, the message is put on the hold queue. This means that a message must fail $(3 * \text{RetryLimit}) + 2$ times before it is put on the hold queue.

In a performance-critical application running in a reliable infrastructure, the retry limit should be small: one or two, for example. This parameter can be found in the administrative console, on the Business Process Container configuration page.

Retention queue message limit

The retention queue message limit defines the maximum number of messages that can be on the retention queue. If the retention queue overflows, the system goes into quiesce mode. To make the system enter quiesce mode as soon as one message fails, set the value to zero. To make the business process container more tolerant of infrastructure failures, increase the value.

This parameter can be found in the administrative console, on the Business Process Container page. (To locate this parameter, click **Servers** → **Application Servers** → *server_name*. Then, under the heading Business Process Container Settings, click **Business Process Container**. The **Retry Limit** field is under the heading General Properties.)

Retention queue

The retention queue holds failed messages that are replayed by moving them back to the business process container's internal work queue. A message is put on the retention queue if it fails three times. If the message fails $(3 * \text{RetryLimit}) + 2$ times, it is put on the hold queue. (For details of the retry limit, see "Retry limit.") If the retention queue is full to the limit defined by the retention queue message limit and another message fails, the queue overflows, and the system goes into quiesce mode. The administrator can move the messages in this queue back to the internal queue performing the task Querying and replaying failed messages.

Hold queue

The hold queue contains messages that have failed $(3 * \text{RetryLimit}) + 2$ times. (For details of the retry limit, see "Retry limit.") The administrator can move the messages in this queue back to the internal queue performing the task Querying and replaying failed messages.

Replay Messages

The administrator can move the messages from the hold or retention queues back to the internal queue. This can be done using the administrative console or using administrative commands.

Quiesce Mode

Quiesce mode is entered when the retention queue overflows. When this happens, it is assumed that there is a serious, though possibly temporary, infrastructure failure. The purpose of quiesce mode is to prevent the system from using a lot of resources, while an infrastructure failure means that most messages will probably fail anyway. In quiesce mode, the system sleeps for two seconds before attempting to process the next message. As soon as a message is successfully processed, the system resumes normal message processing.

Failed message handling for human tasks

The human task container does not have a retention queue, nor retry limits. It only has a hold queue, on which failed messages are placed, and from which, they can be replayed.

Refreshing staff queries, using the administrative console

The results of a staff query are static. Use the administrative console to refresh staff queries.

Business Process Choreographer caches the results of staff assignments evaluated against a staff directory, such as an Lightweight Directory Access Protocol (LDAP) server, in the runtime database. If the staff directory changes, you can force the staff assignments to be evaluated again.

To refresh the staff queries:

1. Click **Servers** → **Application servers** → *server_name*.
2. On the **Configuration** tab, in the Container Settings section, click **Human task container settings** → **Runtime Configuration** → **Refresh Staff Queries**.

Note: When security is enabled, the refresh button is only visible to users who have operator authority.

All staff queries are refreshed.

Enabling Common Base Events and the audit trail

Use this task to enable Business Process Choreographer events to be emitted to the Common Event Infrastructure as Common Base Events or stored in the audit trail.

You can change the state observers settings for the business process container or the human task container, permanently on the Configuration tab, or temporarily on the Runtime tab. Any choices you make on these Configuration or Runtime tabs affect all applications executing in the appropriate container. For changes to affect both the business process container and the human task container, you must change the settings separately for them both.

Changing the configured logging infrastructure:

Use this task to change the state observer logging for the audit log or common event infrastructure logging for the configuration.

Choices made on the Configuration tab are activated the next time the server is started. The chosen settings remain in effect whenever the server is started.

Make changes to the configuration, as follows:

1. Display the Business process container or Human task container pane.
Click **Servers** → **Application servers** → *server_name*. Then, under **Container Settings**, click one of the following sequences:
 - For business processes: **Business process container settings** → **Business process container**
 - For human tasks: **Human task container settings** → **Human task container**
2. In the General Properties section, select the logging to be implemented. The state observers are independent of each other: you can enable or disable either or both of them.

Enable Common Event Infrastructure logging
Select this check box to enable event emission that is based on the Common Event Infrastructure.

Enable audit logging
Select this check box to store the audit log events in the audit trail tables of the relational database.
3. Accept the change.
 - a. Click **Apply**.
 - b. In the Messages box, click **Save**.
 - c. On the Application servers pane, click **Save**.

The state observers are set, as you required. The changes take place after server restart.

Restart the container, to effect the changes.

Configuring the logging infrastructure for the session:

Use this task to change the state observer logging for the audit log or common event infrastructure logging for the session.

Choices made on the Runtime tab are effective immediately. The chosen settings remain in effect until the next time the server is started.

Make changes to the session infrastructure, as follows:

1. Display the Replay messages pane.
Click **Servers** → **Application servers** → *server_name*. Then, under Container Settings, click one of the following sequences:
 - For business processes: **Business process container settings** → **Runtime Configuration**
 - For human tasks: **Human task container settings** → **Runtime Configuration**
2. In the **State observer logging** field, select the logging to be implemented. The state observers are independent of each other: you can enable or disable either or both of them.

Enable Common Event Infrastructure logging
Select this check box to enable event emission that is based on the Common Event Infrastructure.

Enable audit logging
Select this check box to store the audit log events in the audit trail tables of the relational database.
3. Accept the change.

Click **Update State Observers**.

The state observers are set, as you required.

Event emission and storage:

Events for state changes can be generated for executing business processes, human tasks, or both.

Two infrastructures emit or store the events, such that those events can be retrieved by applications. Applications might use events to monitor business processes and to analyze the history of business processes, or human tasks, or both.

Task events, for example, can be emitted without having a business process involved. These events can be consumed by the audit trail and the Common Event Infrastructure (CEI). This applies to standalone tasks, purely human tasks, and tasks invoked by application components other than business processes.

Because the generation of events impacts system performance, you can select the infrastructure to be used to store and emit events:

Common Event Infrastructure

Events can be both stored and published to subscribing applications. To use this event infrastructure, make sure that the Common Event Infrastructure is installed and configured.

Use event emission that is based on the Common Event Infrastructure to retrieve events in the format of Common Base Events, through the application programming interface (API) of the Common Event Infrastructure. You can connect consuming applications either by subscription or by using the query-oriented interface of the Common Event Infrastructure.

Event emission that is based on the Common Event Infrastructure has considerably greater impact on system performance than have audit log events. However, it provides greater flexibility for consuming applications.

Audit trail

Events are stored as records of a table in a relational database.

This is a fast event-storing infrastructure that has little impact on performance. Consuming applications need Structured Query Language (SQL) queries to retrieve the events from the database.

You can select either, both, or neither of the infrastructures. Selecting an infrastructure does not imply that events are necessarily stored or emitted. The selection enables the infrastructure, while you can control the actual generation of events later, by additional mechanisms. However, the enablement of an infrastructure results in a basic overhead that affects system performance.

Using scripts to administer Business Process Choreographer

Describes the administrative actions that can be performed using scripts.

Deleting audit log entries, using administrative commands

Use the administrative commands to delete some or all audit log entries.

Before you begin this procedure, the following conditions must be met:

- The user ID that you are using must have administrative rights.
- The application server through which audit log entries are to be deleted must be running. That is, the `-conntype none` option of `wsadmin` cannot be used, because a server connection is required.
- When security is enabled, you must have operator authority.

You can use the `deleteAuditLog.jacl` script to delete audit log entries from the database.

1. Change to the Business Process Choreographer utilities directory where the scripts are located.

Enter the following command:

```
cd install_root/ProcessChoreographer/util
```

2. Delete the entries in the audit log table.

Enter one or more of the following commands. The differences between the commands are emphasized:

```
install_root/bin/wsadmin -f deleteAuditLog.jacl
                        -server serverName
                        [-profile profileName]
                        [options]
```

```
install_root/bin/wsadmin -f deleteAuditLog.jacl
                        -node nodeName
                        -server serverName
                        [-profile profileName]
                        [options]
```

```
install_root/bin/wsadmin -f deleteAuditLog.jacl
                        -cluster clusterName
                        [-profile profileName]
                        [options]
```

Where:

cluster *clusterName*

The name of the cluster. Required if the business process container is configured for a WebSphere cluster.

node *nodeName*

Optional when specifying the server name. This name identifies the node. The default is the local node.

server *serverName*

The name of the server. Required if the cluster name is not specified.

profileName *profileName*

The name of a user-defined profile. Specify this option if you are not working with the default profile.

The available options are:

-all

Deletes all the audit log entries in the database. The deletion is done in multiple transactions. Each transaction deletes the number of entries specified in the slice parameter, or the default number.

-time *timestamp*

Deletes all the audit log entries that are older than the time you specify for *timestamp*. The time used is coordinated universal time (UTC). Its format must be: YYYY-MM-DD[^THH:MM:SS]. If you specify only the year, month, and day, the hour, minutes, and seconds are set to 00:00:00.

The *-time* and *-processtime* options are mutually exclusive.

-processtime *timestamp*

Deletes all the audit log entries that belong to a process that finished before the time you specify for *timestamp*. Use the same time format as for the *-time* parameter.

The *-time* and *-processtime* options are mutually exclusive.

-slice *size*

Used with the *-all* parameter, *size* specifies the number of entries included in each transaction. The optimum value depends on the available log size for your database system. Higher values require fewer transactions but you might exceed the database log space. Lower values might cause the script to take longer to complete the deletion. The default size for the slice parameter is 250.

Deleting process templates and task templates that are no longer valid

Use the administrative commands to delete, from the database, no longer valid process templates, or no longer valid task templates, or both.

Before you begin this procedure, the application server on which templates to be deleted must be running. That is, the *-conntype none* option of *wsadmin* cannot be used, because a server connection is required. No special authority is required to run this command, even if security is enabled.

Use the methods described here to remove, from the database, templates and all objects that belong to them if no corresponding valid application in the WebSphere configuration repository contains them. This situation can occur if an application installation was canceled or not stored to the Configuration Repository by the user. These templates usually have no impact. They are not shown in Business Process Choreographer Explorer.

There are rare situations in which these templates cannot be filtered. They must then be removed from the database with the following scripts.

You cannot use the scripts to remove templates of valid applications from the database. This condition is checked and a *ConfigurationException* exception is thrown if the corresponding application is valid.

- Change to the Business Process Choreographer utilities directory where the scripts are located.

Type the following command:

```
cd install_root/ProcessChoreographer/uti1
```

- Delete, from the database, business process templates or human task templates that are no longer valid.

To delete, a business process template that is no longer valid, enter one of the following commands. The differences between the commands are emphasized:

```
install_root/bin/wsadmin.sh -f deleteInvalidProcessTemplate.jacl  
    -server serverName  
    -template templateName  
    -validFrom validFromString  
    [-profileName profileName]
```

```
install_root/bin/wsadmin.sh -f deleteInvalidProcessTemplate.jacl  
    -server serverName  
    -node nodeName  
    -template templateName
```

```
-validFrom validFromString  
[-profileName profileName]
```

```
install_root/bin/wsadmin.sh -f deleteInvalidProcessTemplate.jacl  
-cluster clusterName  
-template templateName  
-validFrom validFromString  
[-profileName profileName]
```

To delete, a human task template that is no longer valid, enter one of the following commands. The differences between the commands are emphasized:

```
install_root/bin/wsadmin.sh -f deleteInvalidTaskTemplate.jacl  
-server serverName  
-template templateName  
-validFrom validFromString  
-nameSpace nameSpace  
[-profileName profileName]
```

```
install_root/bin/wsadmin.sh -f deleteInvalidTaskTemplate.jacl  
-server serverName  
-node nodeName  
-template templateName  
-validFrom validFromString  
-nameSpace nameSpace  
[-profileName profileName]
```

```
install_root/bin/wsadmin.sh -f deleteInvalidTaskTemplate.jacl  
-cluster clusterName  
-template templateName  
-validFrom validFromString  
-nameSpace nameSpace  
[-profileName profileName]
```

Where:

cluster *clusterName*

The name of the cluster. Required if the business process container is configured for a WebSphere cluster. You can specify the cluster name or the server name and node name.

node *nodeName*

Optional when specifying the server name. This name identifies the node. The default is the local node. You can specify the server name and node name or the cluster name.

server *serverName*

The name of the server. Required if the cluster name is not specified. You can specify the server name and node name or the cluster name.

template *templateName*

The name of the process template or task template to be deleted.

validFrom *validFromString*

The date from which the template is valid (in UTC) as displayed in the administrative console. The string should have the following format: 'yyyy-MM-ddThh:mm:ss' (year, month, day, T, hours, minutes, seconds). For example, 2005-01-31T13:40:50

nameSpace *nameSpace*

The target namespace of the task template.

profileName *profileName*

The name of a user-defined profile. Specify this option if you are not working with the default profile.

Querying and replaying failed messages, using administrative commands

Use the administrative commands to determine whether there are any failed messages for business processes or human tasks, and, if there are, to retry processing them.

Before you begin this procedure, the following conditions must be met:

- The user ID that you are using must have administrative rights.
- The application server on which the messages are to be queried or replayed must be running. That is, the `-conntype none` option of the `wsadmin` script cannot be used, because a server connection is required.
- When security is enabled, you must have operator authority.

When a problem occurs while processing an internal message, this message ends up on the retention queue or hold queue. To determine whether any failed messages exist, and to send those messages to the internal queue again:

1. Change to the Business Process Choreographer utilities directory where the scripts are located: Type the following:
`cd install_root/ProcessChoreographer/util`
2. Query the number of failed messages on both the retention and hold queues. Enter one of the following commands. The differences between the commands are emphasized:

```
install_root/bin/wsadmin -f queryNumberOfFailedMessages.jacl  
                        -cluster clusterName  
                        [ -bfm | -htm ]  
                        [-profile profileName]
```

```
install_root/bin/wsadmin -f queryNumberOfFailedMessages.jacl  
                        -node nodeName  
                        -server serverName  
                        [ -bfm | -htm ]  
                        [-profile profileName]
```

Where:

cluster *clusterName*

The name of the cluster. Required if the business process container is configured for a WebSphere cluster.

node *nodeName*

Optional when specifying the server name. This name identifies the node. The default is the local node.

server *serverName*

The name of the server. Required if the cluster name is not specified.

bfm | htm

These keywords are optional. The default, if neither option is specified is to display all failed messages for both business processes and human tasks. If you only want to display the number of messages in the business process container hold and retention queues, specify `bfm`. If you only want to display the number of messages in the human task container hold queue, specify `htm`.

profile *profileName*

The name of a user-defined profile. Specify this option if you are not working with the default profile.

3. Replay all failed messages on the hold queue, retention queue, or both queues. Enter one of the following commands:

```
install_root/bin/wsadmin -f replayFailedMessages.jacl -cluster clusterName -queue replayQueue  
install_root/bin/wsadmin -f replayFailedMessages.jacl -node nodeName -server serverName -queue  
install_root/bin/wsadmin -f replayFailedMessages.jacl -server serverName -queue replayQueue pr
```

Where:

queue *replayQueue*

Must have one of the following values:

holdQueue
retentionQueue
both

cluster *clusterName*

The name of the cluster. Required if the business process container is configured for a WebSphere cluster.

node *nodeName*

Optional when specifying the server name. This name identifies the node. The default is the local node.

server *serverName*

The name of the server. Required if the cluster name is not specified.

bfm|htm

These keywords are optional and mutually exclusive. The default, if neither option is specified is to replay failed messages for both business processes and human tasks. If you only want to replay the messages for business processes, specify bfm. If you only want to replay messages for human tasks, specify htm.

profile *profileName*

The name of a user-defined profile. Specify this option if you are not working with the default profile.

Refreshing staff queries, using administrative commands

The results of a staff query are static. Use the administrative commands to refresh staff queries.

Before you begin this procedure, the following conditions must be met:

- The user ID that you are using must have administrative rights.
- The application server on which the messages are to be queried or replayed must be running. That is, the `-conntype none` option of `wsadmin` cannot be used, because a server connection is required.
- When security is enabled, you must have operator authority.

Business Process Choreographer caches the results of staff assignments evaluated against a staff directory, such as an Lightweight Directory Access Protocol (LDAP) server, in the runtime database. If the staff directory changes, you can force the staff assignments to be evaluated again.

1. Change to the Business Process Choreographer utilities directory where the scripts are located.

Enter the following command:

```
cd install_root/ProcessChoreographer/util
```

2. Force the staff assignment to be evaluated again.

Enter one of the following commands. The differences between the commands are emphasized:

```
install_root/bin/wsadmin -f refreshStaffQuery.jacl
  -server serverName
  [-processTemplate templateName |
  (-taskTemplate templateName [-nameSpace nameSpace]) |
  -userList username{,username}...]
  [-profile profileName]
```

```
install_root/bin/wsadmin -f refreshStaffQuery.jacl
  -node nodeName
  -server serverName
  [-processTemplate templateName |
  (-taskTemplate templateName [-nameSpace nameSpace]) |
  -userList username{,username}...]
  [-profile profileName]
```

```
install_root/bin/wsadmin -f refreshStaffQuery.jacl
  -cluster clusterName
  [-processTemplate templateName |
  (-taskTemplate templateName [-nameSpace nameSpace]) |
  -userList username{,username}...]
  [-profile profileName]
```

Where:

cluster *clusterName*

The name of the cluster. Required if the business process container is configured for a WebSphere cluster.

node *nodeName*

Optional when specifying the server name. This name identifies the node. The default is the local node.

server *serverName*

The name of the server. Required if the cluster name is not specified.

processTemplate *templateName*

The name of the process template. Staff assignments that belong to this process template are refreshed.

taskTemplate *templateName*

The name of the task template. Staff assignments that belong to this process template are refreshed.

nameSpace *nameSpace*

The namespace of the task template.

userList *userName*

A comma-separated list of user names. Staff assignments that contain the specified names are refreshed.

profileName *profileName*

The name of a user-defined profile. Specify this option if you are not working with the default profile.

Note: If you do not specify any *templateName* nor *userList*, all staff queries that are stored in the database are refreshed. You might want to avoid this for performance reasons.

Setting the interval for refreshing staff queries:

Set the interval at which the database refreshes the cached staff queries.

Staff queries are resolved by the specified staff repository. The result is stored in the Business Process Choreographer database. To optimize the staff query

resolution performance, the retrieved query results are cached. The cache content is checked for currency when a new process instance is created or the corresponding staff activity is scheduled. By default, the time after which the shared staff query results expire is one hour. If your staff repository changes infrequently, consider using a large value for this property and refreshing your staff queries on demand by using the `refreshStaffQuery.jacl` script.

You can change the default value for the expiration time for staff query results, in the administrative console.

1. Go to the custom properties page for the human task container.
Click **Servers** → **Application servers** → *Server_Name* → **Human task container** → **Custom properties**.
2. Select **StaffQueryResultValidTimeSeconds** and enter a new value in seconds.
3. Click **OK**.
4. Save the changes and restart the application server to make the changes effective.
The new expiration time value applies only to new staff queries, it does not apply to existing staff queries.

Removing unused staff queries, using administrative commands

Use the administrative commands to remove unused staff queries from the database.

Before you begin this procedure, the following conditions must be met:

- The user ID that you are using must have administrative rights.
- The application server, through which unused staff queries are to be deleted, must be running. That is, the `-conntype none` option of `wsadmin` cannot be used, because a server connection is required.
- When security is enabled, you must have operator authority.

Business Process Choreographer maintains lists of user names in the runtime database for staff expressions that have been evaluated. Although the processes instances that used these staff expressions have finished, the lists of user names are maintained in the database until the corresponding business process application is uninstalled.

If the size of the database is affecting performance, you can remove the unused staff lists that are cached in the database tables.

1. Change to the Business Process Choreographer utilities directory where the scripts are located.

Enter the following command:

```
cd install_root/ProcessChoreographer/util
```

2. Remove the unused staff lists.

Enter one of the following commands. The differences between the commands are emphasized:

```
install_root/bin/wsadmin -f cleanupUnusedStaffQueryInstances.jacl  
-server serverName  
[-profile profileName]
```

```
install_root/bin/wsadmin -f cleanupUnusedStaffQueryInstances.jacl  
-node nodeName  
-server serverName  
[-profile profileName]
```



```
install_root/bin/wsadmin -f cleanupUnusedStaffQueryInstances.jacl  
                        -cluster clusterName  
                        [-profile profileName]
```

Where:

cluster *clusterName*

The name of the cluster. Required if the business process container is configured for a WebSphere cluster.

node *nodeName*

Optional when specifying the server name. This name identifies the node. The default is the local node.

server *serverName*

The name of the server. Required if the cluster name is not specified.

profileName *profileName*

The name of a user-defined profile. Specify this option if you are not working with the default profile.

The number of entries deleted from the database is displayed.

Administering applications and application services

This section describes how to use the administrative interfaces to administer WebSphere Process Server applications and application services, including business processes and tasks, business rules, and schedules.

Administering business processes and human tasks

Business processes and human tasks are deployed and installed as part of an enterprise application. You can use the administrative console or the administrative commands to administer process templates and task templates, and Business Process Choreographer Explorer to work with process instances and task instances.

Administering process templates and process instances

Use the administrative console or the administrative commands to administer process templates. Use Business Process Choreographer Explorer to work with process instances.

Process templates define business processes within an enterprise application. When an enterprise application that contains process templates is installed, deployed, and started, the process templates are put into the started state. You can use the administrative console or the administrative commands to stop and start process templates.

Process templates are shown in Business Process Choreographer Explorer. A process instance can be a long-running process or a microflow. Use Business Process Choreographer Explorer to display information about process templates and process instances, or act on process instances. These actions can be, for example, starting process instances; and for long-running processes other process life cycle actions, such as suspending, resuming, or terminating process instances; or repairing activities.

Authorization roles for business processes:

Actions that you can take on business processes depend on your authorization role. This role can be a J2EE role or an instance-based role.

A role is a group of employees who share the same level of authority. Java 2 Platform, Enterprise Edition (J2EE) roles are set up when the business process container is configured. Instance-based roles are assigned to processes and activities when the process is modeled. Role-based authorization requires that global security is enabled in WebSphere Application Server.

J2EE roles

The following J2EE roles are supported:

- J2EE BPESystemAdministrator. Users assigned to this role have all privileges.
- J2EE BPESystemMonitor. Users assigned to this role can view the properties of all business process objects.

You can use the administrative console to change the assignment of users and groups to these roles.

Setting up Roles using RACF security: These RACF permissions apply when the following security fields are specified:

- **com.ibm.security.SAF.authorization= true**

```
RDEFINE EJBROLE BPESystemAdministrator UACC(NONE)
PERMIT BPESystemAdministrator CLASS(EJBROLE) ID(userid) ACCESS(READ)

RDEFINE EJBROLE BPESystemMonitor UACC(NONE)
PERMIT BPESystemMonitor CLASS(EJBROLE) ID(userid) ACCESS(READ)
```
- **com.ibm.security.SAF.delegation= true**

```
RDEFINE EJBROLE JMSAPIUser UACC(NONE) APPLDATA(' userid')
```

You can use Security Authorization Facility (SAF)-based authorization (for example, using the RACF EJBROLE profile) to control access by a client to Java 2 Platform, Enterprise Edition (J2EE) roles in EJB and Enterprise applications, including the Business process container. For more information on using SAF, see System Authorization Facility for role-based authorization in the WebSphere Application Server for z/OS information center.

Instance-based roles

A process instance or an activity is not assigned directly to a staff member in the process model, instead it is assigned to one of the available roles. Any staff member that is assigned to an instance-based role can perform the actions for that role. The association of users to instance-based roles is determined at runtime using staff resolution.

The following instance-based roles are supported:

- For processes: reader, starter, administrator
- For activities: reader, editor, potential owner, owner, administrator

These roles are authorized to perform the following actions:

Role	Authorized actions
Activity reader	View the properties of the associated activity instance, and its input and output messages.
Activity editor	Actions that are authorized for the activity reader, and write access to messages and other data associated with the activity.

Role	Authorized actions
Potential activity owner	Actions that are authorized for the activity reader. Members of this role can claim the activity, and send messages to receive or pick activities.
Activity owner	Work on and complete an activity. Members of this role can transfer owned work items to an administrator or a potential owner.
Activity administrator	Repair activities that are stopped due to unexpected errors, and force terminate long-running activities.
Process starter	View the properties of the associated process instance, and its input and output messages.
Process reader	View the properties of the associated process instance, its input and output messages, and everything that the activity reader supports for all of the contained activities but not those of the subprocesses.
Process administrator	Members of this role can administer process instances and intervene in a process that has started; create, delete, and transfer work items. Members of this role also have activity administrator authorization.

Do not delete the user ID of the process starter from your user registry if the process instance still exists. If you do, the navigation of this process cannot continue. You receive the following exception in the system log file:

```
no unique ID for: <user ID>
```

Business process administration—frequently asked questions:

Answers to a set of frequently asked questions about administering business processes.

- “What happens if a process template is in the started state, but the application it belongs to is in the stopped state?”
- “How do I stop new process instances being created?”
- “What happens to running instances when a newer process template becomes valid?” on page 47
- “What happens to a running instance if the template it was created from is stopped?” on page 47
- “How can I tell if any process instances are still running?” on page 47
- “Why can’t I stop a business process application if it has any process instances?” on page 47

What happens if a process template is in the started state, but the application it belongs to is in the stopped state?

If a currently valid process template is in the started state, but the application is in the stopped state, no new process instances are created from the template. Existing process instances cannot be navigated while the application is in the stopped state.

How do I stop new process instances being created?

Using the administrative console, select a process template, and click **Stop**. This action put the process template into the stopped state, and no more instances are created from the template. After the template stops, any attempts to create a

process instance from the template result in an `EngineProcessModelStoppedException` error.

What happens to running instances when a newer process template becomes valid?

If a process template is no longer valid, this fact has no effect on running instances that were instantiated from the template. Existing process instances continue to run to completion. Old and new instances run in parallel until all of the old instances have finished, or until they have been terminated.

What happens to a running instance if the template it was created from is stopped?

Changing the state of a process template to 'stopped' only stops new instances being created. Existing process instances continue running until completion in an orderly way.

How can I tell if any process instances are still running?

Log on to the Business Process Choreographer Explorer as a process administrator, and go to the Process Instances Administered By Me page, this displays any running process instances. If necessary, you can terminate and delete these process instances.

Why can't I stop a business process application if it has any process instances?

For a process instance to run, its corresponding application must also be running. If the application is stopped, the navigation of the process instance cannot continue. For this reason, you can only stop a business process application if it has no process instances.

Life cycle management and versioning behavior of subprocesses:

A process that is started by another process is known as a *subprocess*. The way in which the life cycle of subprocesses can be managed and the versioning behavior of subprocesses depend on how these processes are modeled.

For modularity and reuse, it often makes sense to apply the programming concept of encapsulation to business process modeling, that is to implement one or more steps of the business logic as a separate process and to invoke this process from the main process. A subprocess can also start another process. This can lead to an arbitrarily deep hierarchy of process instances. When these processes are deployed, all of the process templates in the process-to-process relationship must be deployed to the same Business Process Choreographer database.

Life cycle management

A subprocess can have a peer-to-peer relationship or a parent-child relationship with the calling process. This relationship determines the behavior of a subprocess when an action that manages the process life cycle is invoked for the calling process. The life cycle actions comprise suspend, resume, terminate, delete, and compensation. Actions that manage the process life cycle can be taken only on top-level process instances.

The calling process-subprocess relationship is determined by the autonomy attribute of the subprocess. This attribute can have one of the following values:

Peer A peer process is considered to be a *top-level process*. A top-level process is a process instance that either is not invoked by another process instance or is invoked by another process instance, but it has peer autonomy. If the subprocess is part of a peer-to-peer relationship, life cycle actions on the calling process instance are not applied to the subprocess instance.

However, for long-running processes with a creating operation that implements a one-way interface, the value of the autonomy attribute is automatically set to peer during runtime. If the autonomy attribute is set to child, this value is ignored at runtime.

Child If the subprocess is part of a parent-child relationship, life cycle actions on the parent process instance are applied to the subprocess instance. For example, if the parent process instance is suspended, all of the subprocess instances with child autonomy are suspended, too.

A microflow always runs as a child process. However, if there is another component between the two processes, it might prevent a parent-child relationship from being established, for example, an interface map component that is wired between the two process components.

Versioning behavior

The version of a process that is used is determined by whether the process is used in an *early-binding* scenario or a *late-binding* scenario.

Early binding

In an early-binding scenario, the decision on which version of the subprocess is invoked is made during deployment. The calling process invokes a dedicated, statically-bound subprocess according to the Service Component Architecture (SCA) wiring. The versioning of the process is ignored.

An example of early-binding is an SCA wire. For example, if you wire a stand-alone reference to a process component, every invocation of the process using this reference is targeted to the specific version that is represented by the process component.

Late binding

In a late-binding scenario, the decision on which subprocess template is invoked happens when the calling process instance needs to invoke the subprocess. In this case, the version of the subprocess that is currently valid is used. A newer version of a process supersedes all of the previous versions of the process. Existing process instances continue to run with the process template with which they were associated when they started. This leads to the following categories of process templates:

- Process templates that are no longer current might still be valid for existing long-running process instances
- Current process templates are used for new process instances
- Process templates that become valid in the future according to their valid-from date and time.

To apply late-binding when a subprocess is invoked, the parent process must specify the name of the subprocess template from which the valid subprocess is to be chosen at the reference partner. The valid-from attribute

of the process is used to determine the subprocess template that is currently valid. Any SCA wiring is ignored.

An example of late-binding is when a new process is invoked in Business Process Choreographer Explorer. The instance that is created is always based on the most recent version of the process template with a valid-from date that is not in the future.

When a new version of a process model is created and the existing process model is used in late-binding scenarios, you must avoid making changes that will lead to compatibility problems when the new version of the process becomes valid and, for example, a parent process invokes an instance of the new version of the subprocess. The following are incompatible changes that you must avoid:

- Modifying the correlation sets
- Changing any interface that is used by the parent process to communicate with the subprocess

Stopping and starting process templates with the administrative console:

You can use the administrative console to start and stop each installed process template individually.

If global security is enabled, verify that your user ID has operator authorization. The server on which the application is installed must be running.

You must stop a process template, for example, before you can uninstall the business process application to which it belongs. The following steps describe how to use the administrative console to administer process templates.

1. Select the application that you want to manage.
In the navigation pane of the administrative console, click **Applications** → **Enterprise Applications**, and then the application that you want to manage.
2. Select an EJB module.
Under Related Items, click **EJB Modules** and then an EJB module.
3. Select the process template that you want to manage.
Under Additional Properties, click **Business Processes** and then a process template.
4. Stop the process template.
Existing instances of the process templates continue to run until they end normally. However, you cannot create process instances from a stopped template.
5. Start the process template that is in the stopped state.

Stopping and starting process templates with administrative commands:

Administrative commands provide an alternative to the administrative console for stopping and starting process templates.

If global security is enabled, verify that your user ID has operator authorization.

You must stop a business template, for example, before you can uninstall the business process application to which it belongs. The following steps describe how to use the administrative commands to administer process templates.

1. Change to the Business Process Choreographer samples directory. Type the following:

```
cd install_root/ProcessChoreographer/sample
```

2. Stop the process template.

```
install_root/bin/wsadmin -f bpcTemplates.jacl  
                        -stop application_name
```

Where *application_name* is the name of the application to which the template belongs.

Existing instances of the process templates continue to run until they end normally. When the application stops, you cannot create process instances from the stopped templates.

3. Start the process template.

```
install_root/bin/wsadmin -f bpcTemplates.jacl  
                        -start application_name
```

The process template starts. You can use Business Process Choreographer Explorer to start process instances from the process template.

Starting a new process instance:

You can start a new process instance from any of the process templates that you are authorized to use.

All of the installed process templates are shown in the list of process templates in Business Process Explorer. To start a new process instance, complete the following steps.

1. Display the process templates that you are authorized to use.

Click **My Process Templates** under Process Templates in the navigation pane.

2. Select a process template from the list and click **Start Instance**.

This action displays the Process Input Message page. Here you can provide the input data that is needed to start an instance of the business process from this page.

If the process has more than one operation, this action displays a page that contains all of the available operations. Select the operation that is to start the process instance.

3. Provide the input data to start the process instance.

If the process is a long-running process, you can type in a process instance name. If you do not specify a name, a system-generated name is assigned to the new process instance.

Complete the input for the process input message.

4. To start the process, click **Submit**.

The process instance is started. If the business process contains an activity that requires human interaction, tasks are generated for all of the potential owners. If you are one of these potential owners, this task appears in the list on the My Tasks page.

If the process is a long-running process, a process output message is displayed immediately after the process finishes. Not all processes have output messages, for example, if the process implements a one-way operation, an output message is not displayed.

Suspending and resuming process instances:

You can suspend a running process instance and then resume it again later.

To suspend and resume process instances, you must have process administrator authorization.

To suspend a process instance, the process instance must be in either the running or failing state. To resume a process, the process instance must be in the suspended state.

You can suspend a long-running, top-level process instance. You might want to do this, for example, so that you can configure access to a back-end system that is used later in the process, or to fix a problem that is causing the process instance to fail. When the prerequisites for the process are met, you can resume running the process instance.

To suspend or resume a process instance, complete the following steps in Business Process Choreographer Explorer.

1. Display a list of process instances.

For example, click **Administered By Me** under Process Instances in the navigation pane.

2. Suspend the process.

Select a process instance and click **Suspend**.

This action suspends the specified top-level process instance. The process instance is put into the suspended state. Subprocesses with the autonomy attribute set to `child` are also suspended if they are in the running, failing, terminating, or compensating state. However, you can still complete any active activities and tasks that belong to the process instance.

3. Resume the process instance.

Select a process instance that is in the suspended state and click **Resume**. The process instance and its subprocess are put into the states they had before they were suspended, for example, running. The process instance and its subprocesses resume.

Administering compensation for microflows:

If compensation fails for a microflow, there are a number of administrative actions you can take.

For microflows to be compensated, the compensation service must be started in the administrative console.

When a microflow runs, it can encounter problems. For these situations, compensation might have been defined for the process in the process model. Compensation allows you to undo previous completed steps, for example, to reset data and states so that you can recover from these problems.

However, the compensation processing might also fail. When a compensation for a microflow fails, the process administrator must intervene to resolve the problems.

In Business Process Choreographer Explorer, complete the following steps to administer failed compensation actions.

1. Display a list of the compensation actions that failed.

Click **Failed Compensations** under Process Instances in the navigation pane.

The Failed Compensations page is displayed. This page contains information about why the named compensation action failed. This information can help you to decide what actions to take to correct the failed compensation.

2. Select an activity and then click one of the available actions.

The following administrative actions are available:

- Skip** Skips the current compensating action and continues with compensating the microflow. This action might result in a non-compensated activity.
- Retry** If you have taken action to correct the failed compensation action, click **Retry** to try the compensation action again.
- Stop** Stops the compensation processing.

Compensation in business processes:

Compensation is the means by which operations in a process that have successfully completed can be undone.

Compensation processing starts because an error occurs in a running process instance for which compensation is defined in the process model. Compensation reverses the effects of operations that were committed up to when the error occurred to get back to a consistent state.

You can define compensation for long-running processes and for microflows in your process model.

Compensation for long-running processes

Compensation for long-running processes is also known as *business-level compensation*. This type of compensation is defined on the scope level. This means that either part of the process, or the entire process can be compensated.

Compensation is triggered by fault handlers or the compensation handler of a scope or a process; compensation is another navigation path of the process.

A long-running process automatically compensates child processes that have successfully completed when the enclosing parent scope is compensated. Within a process, only invoke and scope activities that complete successfully are compensated.

Compensation for microflows

Compensation for microflows is also known as *technical compensation*. This type of compensation is triggered when the work unit (the transaction or the activity session) that contains the microflow is rolled back. Therefore, undo actions are typically specified for activities that cannot be reversed by rolling back the unit of work. When a process instance runs, undo actions for compensable activities are registered with the enclosing unit of work. Depending on the outcome of this unit of work (rollback or commit), compensation starts.

If the microflow is a child of a compensable, long-running process, the undo actions of the microflow are made available to the parent process when the microflow completes. It can, therefore, potentially participate in the compensation of the parent process. For these types of microflows, it is a good practice to specify undo actions for all of the activities in the process when you define your process model.

If an error occurs during compensation processing, the compensation action requires manual resolution to overcome the error. You can use Business Process Choreographer Explorer to repair these compensation actions.

Terminating process instances:

To terminate a process instance, you must have process administrator authorization.

You might want to terminate a process instance, for example, if the work or documents it represents are no longer needed, if no one is available to complete the process instance, if you have encountered problems with the process template and it needs to be redesigned, and so on.

If compensation is defined for the business process model, you can choose to terminate the process instance with compensation.

In Business Process Choreographer Explorer, complete the following steps to terminate a process instance.

1. Display the process instances that you can administer.
Click **Administered By Me** under Process Instances in the navigation pane.
2. Select the process instance that you want to stop.
 - To terminate the process instance with compensation, click **Compensate**. This action terminates the process instance and starts compensation processing.
 - To terminate the process instance without compensation, click **Terminate**. This action stops the process instance immediately without waiting for any outstanding activities or tasks. Process instances that are terminated are not compensated.

Deleting process instances:

Not all process instances are automatically deleted when they complete. You can explicitly delete process instances that have completed.

To delete a process instance, you must have process administrator authorization. The process instance must be in the finished or terminated state.

Completed processes instances are automatically deleted from the Business Process Choreographer database if the corresponding property is set for the process template in the process model.

You might want to keep process instances in your database, for example, to query data from process instances that are not written to the audit log, or if you want to defer the deletion of processes to off-peak times. However, old process instance data that is no longer needed can impact disk space and performance. Therefore, you should regularly delete process instance data that you no longer need or want to maintain. Make sure that you run this maintenance task at off-peak times.

In Business Process Choreographer Explorer, complete the following steps to delete a process instance.

1. Display the process instances that you administer.
Click **Administered By Me** under Process Instances in the navigation pane.
2. Select the process instance that you want to delete and click **Delete**.

This action deletes the selected process instance from the database.

Administering task templates and task instances

Use the administrative console or the administrative commands to administer task templates. Use Business Process Choreographer Explorer to work with task instances.

Authorization roles for human tasks:

Actions that you can take on human tasks depend on your authorization role. This role can be a J2EE role or an instance-based role.

A role is a group of employees who share the same level of authority. Java 2 Platform, Enterprise Edition (J2EE) roles are set up when the human task container is configured. Instance-based roles are assigned to human tasks and escalations when the task is modeled. Role-based authorization requires that global security is enabled in WebSphere Application Server.

J2EE roles

The following J2EE roles are supported:

- J2EE TaskSystemAdministrator. Users assigned to this role have all privileges.
- J2EE TaskSystemMonitor. Users assigned to this role can view the properties of all of the task objects.

You can use the administrative console to change the assignment of users and groups to these roles.

Setting up Roles using RACF security: These RACF permissions apply when the following security fields are specified:

- **com.ibm.security.SAF.authorization= true**
RDEFINE EJBROLE TaskSystemAdministrator UACC(NONE)
PERMIT TaskSystemAdministrator CLASS(EJBROLE) ID(userid) ACCESS(READ)
RDEFINE EJBROLE TaskSystemMonitor UACC(NONE)
PERMIT TaskSystemMonitor CLASS(EJBROLE) ID(userid) ACCESS(READ)
- **com.ibm.security.SAF.delegation= true**
RDEFINE EJBROLE JMSAPIUser UACC(NONE) APPLDATA(' userid')

You can use Security Authorization Facility (SAF)-based authorization (for example, using the RACF EJBROLE profile) to control access by a client to Java 2 Platform, Enterprise Edition (J2EE) roles in EJB and Web applications, including the WebSphere Application Server administrative console application. For more information, see System Authorization Facility for role-based authorization in the WebSphere Application Server for z/OS information center.

Instance-based roles

A task instance or an escalation instance is not assigned directly to a staff member in the task model, instead it is assigned to one of the available roles. Any staff member that is assigned to an instance-based role can perform the actions for that role. The association of users to instance-based roles is determined at runtime using staff resolution.

The following instance-based roles are supported:

- For tasks: potential instance creator, originator, potential starter, starter, potential owner, owner, reader, editor, administrator
- For escalations: escalation receiver

These roles are authorized to perform the following actions:

Role	Authorized actions
Potential instance creator	Members of this role can create an instance of the task. If no potential instance creator is defined for the task template or the application components, then all users are considered to be a member of this role.
Originator	Members of this role have administrative rights until the task starts. When the task starts, the originator has the authority of a reader and can perform some administrative actions, such as suspending and resuming tasks, and transferring work items.
Potential starter	Members of this role can start an existing task instance. If a potential starter is not specified, the originator becomes the potential starter. For inline tasks without a potential starter, the default is everybody.
Starter	Members of this role have the authority of a reader and can perform some administrative actions, such as transferring work items.
Potential owner	Members of this role can claim a task. If no potential owner is defined for the task template or the application components, then all users are considered to be a member of this role.
Owner	Work on and complete a task.
Reader	View the properties of all of the task objects, but cannot work on them.
Editor	Members of this role can work with the content of a task, but cannot claim or complete it
Administrator	Members of this role can administer tasks, task templates, and escalations.
Escalation receiver	Members of this role have the authority of a reader.

Stopping and starting task templates with the administrative console:

Use the administrative console to start and stop task templates.

If global security is enabled, verify that user ID has operator authorization.

Task templates define Service Component Architecture (SCA) services that are represented as stand-alone tasks within an enterprise application. When an enterprise application that contains task templates is installed, deployed, and started, the task templates are put into the start state.

1. Select the application that you want to manage.
In the navigation pane of the administrative console, click **Applications** → **Enterprise Applications**, and then the application that you want to manage.
2. Select an EJB module.
Under Related Items, click **EJB Modules** and then an EJB module.
3. Select the task template that you want to manage.
Under Additional Properties, click **Human Tasks** and then a task template.

4. To stop the task template, click **Stop**.
5. To start the task template, click **Start**.

Stopping and starting task templates with the administrative commands:

Administrative commands provide an alternative to the administrative console for stopping and starting task templates.

If global security is enabled, verify that you are logged with a user ID that has operator authorization.

Task templates define Service Component Architecture (SCA) services that are represented as stand-alone tasks within an enterprise application. When an enterprise application that contains task templates is installed, deployed, and started, the task templates are put into the start state.

1. Change to the Business Process Choreographer samples directory. Type the following:

```
cd install_root/ProcessChoreographer/sample
```

2. Stop the task template.

```
install_root/bin/wsadmin -f bpcTemplates.jacl  
                        -stop application_name
```

Where *application_name* is the name of the application to which the template belongs. Existing instances of the task template continue to run until they end normally.

3. Start the task template.

```
install_root/bin/wsadmin -f bpcTemplates.jacl  
                        -start application_name
```

The task template starts. You can use Business Process Choreographer Explorer to work with task instances associated with the task template.

Creating and starting a task instance:

You can create and start a task instance from any of the task templates that you are authorized to use.

All of the installed task templates are shown in the list of task templates in Business Process Choreographer Explorer. To create and start a task instance from a task template, complete the following steps.

1. Display the task templates that you are authorized to use.

Click **My Task Templates** under Task Templates in the navigation pane.

2. Select a task template from the list and click **Create Instance**.

3. Start the task instance.

Click **Task Instances Initiated By Me** under Task Instances in the navigation pane, select the task instance, and click **Start Instance**.

This action displays the Task Input Message page. Here you can provide the input data that is needed to start an instance of the task template from this page.

4. Provide the input data to start the task instance.
5. To start the task instance, click **Submit**.

The task instance is ready to be worked on.

Working on your tasks:

To work on a task, you must claim the task and then perform the actions needed to complete it.

You can claim a task that is in the ready state if you are a potential owner or the administrator of that task. If you claim a task, you become the owner of that task and are responsible for completing it.

Tasks for which you have the role of reader or editor also appear on your list of tasks.

To claim and complete a task with Business Process Choreographer Explorer, complete the following steps.

1. Display the tasks that have been assigned to you.

Click **Task Instances** → **My Tasks**.

This action displays the My Tasks page, which lists the tasks that have been assigned to you.

2. Claim the task on which you want to work.

Select the check box next to the task and click **Work on**.

This action displays the Task Message page.

3. Provide the information to complete the task.

If you need to interrupt your work, for example, because you need more information from a co-worker to complete the task, click **Save** to save the changes you made.

4. Click **Complete** to complete the task with the information that you provide.

The task that you completed is in the finished state. If you leave the task without completing it, the task remains in the claimed state.

Managing task assignments:

After a task has started, you might need to manage task assignments for the task.

A *work item* is the assignment of a business entity, such as a task or a process instance, to a person or a group of people for a particular reason. The assignment reason allows a person to play various roles in the business process scenario, for example, for example, potential owner, editor, or administrator.

A task instance can have several work items associated with it because different people can have different roles. For example, John, Sarah, and Mike are all potential owners of a task instance and Anne is the administrator; work items are generated for all four people. John, Sarah, and Mike see only their own work items as tasks on their list of tasks. Because Anne is the administrator, she gets her own work item for the task and she can manage the work items generated for John, Sarah, and Mike.

Sometimes, you might need to change a task assignment after a task has been started, for example, to transfer a work item from the original owner to someone else. You might also need to create additional work items or delete work items that are not needed anymore.

Transferring work items:

You might want to transfer a work item to another user, for example, if the work-item owner is on vacation and the work item must be completed before this person returns.

To transfer a work item you must have one of the following roles.

Role	Task state	Work items can be transferred to the following user roles:
Owner	Claimed	Potential owner, administrator.
Starter	Terminated, expired, finished, failed, or running	Potential starter, administrator.
Originator	Any task state	Potential instance creator, administrator. If the task is in the active state, it can be transferred to any user role.
Administrator	Ready, claimed, terminated, expired, finished, failed, or running	Any user role.

In Business Process Choreographer Explorer, complete the following steps to transfer a work item.

1. Display the task instances that you administer.
Click **Administered By Me** under Task Instances in the navigation pane.
2. Display the work items for a task instance.
In the Task Instances Administered By Me page, select a task instance and click **Work Items**.
3. Transfer the work item.
 - a. In the **New Owner** field, specify the user ID of the new work-item owner.
If you are the task administrator, you can transfer the work item to anyone. If you are the current owner of the work item, you can transfer the work item only to another potential owner of the work item or the task administrator.
 - b. Select one or more roles from the **Reason** list. These roles determine the actions that the assigned person can perform on the transferred work item.
 - c. Click **Transfer**.

The transferred work item appears on the list of tasks belonging to the new work-item owner.

Creating work items:

You might want to create work items for new potential owners, for example, when none of the current potential owners can accept any additional work. You might also want to create work items if the query against the staff repository does not return any potential owners. This might happen, for example, in a long-running process if the organization has changed since the process started.

To create a work item for a task instance, you must have the appropriate role for the task. If you are the task administrator, you can create work items for the task instance if it is in one of the following states: ready, claimed, running, finished, or

failed. If the task instance is derived from a task template, you can also create work items if the task is in the terminated or expired state.

In Business Process Choreographer Explorer, complete the following steps to create a work item.

1. Display the task instances that you administer.
Click **Administered By Me** under Task Instances in the navigation pane.
2. Select the task instance for which you want to create a work item and click **Create Work Items**. The Create Work Items page is displayed.
3. Create the work items.
 - a. In the **New Owner** field, specify the user ID of the new work-item owner.
 - b. Select one or more roles from the **Reason** list.
These roles determine the actions that the assigned person can perform on the new work item.
 - c. Click **Create**.

A work item is created for each role that you specify for the new work-item owner. The new task appears on the list of tasks assigned to this person.

Deleting work items:

You might want to delete work items, for example, if you created work items in error or if work items are generated for someone who no longer works for the company.

To delete a work item for a task instance, you must have the appropriate role for the task. If you are the task administrator, you can delete the task instance if it is in one of the following states: ready, claimed, running, finished, or failed. If the task instance was derived from a task template, you can also delete the task in the terminated or expired state.

In Business Process Choreographer Explorer, complete the following steps to delete a work item.

1. Display the task instances that you administer.
Click **Administered By Me** under Task Instances in the navigation pane.
2. Display the work items for a task instance.
In the Task Instances Administered By Me page, select a task instance and click **Work Items**.
3. Delete the work items.
Select a work item and click **Delete**.

The work items are deleted.

Viewing task escalations:

An escalation notifies the escalation receiver that a user might have problems completing their assigned task on time.

When a task becomes overdue, it might result in an escalation. An escalation can result in the following actions:

- A new work item is created, for example, for a manager to take action to support the resolution of the problem.

- If you specified e-mail settings when you configured the human task container, an e-mail is sent to a designated person to inform them about the escalated task.
- An event notification handler is called.

To view escalations, click **My Escalations** under Task Instances.

- To view information about an escalation, click the escalation ID.
- To view information about an escalated task, click the task name.

Business rules

Use business rules to control the behavior of a business practice. (For more information about building and deploying business rules, see the WebSphere Integration Developer information center.)

What is a business rule?

A business rule is anything that imposes structure upon, or controls the behavior of a business practice. A rule can enforce business policy, establish common guidelines within an organization, or control access in a business environment. (For more information about building and deploying business rules, see the WebSphere Integration Developer information center.)

When to use a business rule

Use business rules to officiate over frequently changing business practices that can come from within a business or mandated from outside a business, such as regulatory agencies. Some typical uses for business rules are:

- Determining current interest rates
- Calculating discounts for products
- Calculating sales tax
- Determining special groups such as senior citizens or preferred customers

(For more information about building and deploying business rules, see the WebSphere Integration Developer information center.)

How to use business rules

Create and modify business rules using an eclipse-based WebSphere Integration Developer tool. (For more information about building and deploying business rules, see the WebSphere Integration Developer information center.) Manage and modify business rule values using the Web-based business rules manager tool.

Installing the business rules dynamic repository for a standalone server

When you run the installation and configuration script, the dynamic repository is installed in a standalone server..

If you select a database that does not support business rules, the system uses the default database, Cloudscape.

DB2 for z/OS version 7 limits the size of the primary key to 255 bytes. The dynamic artifact repository for business rule and selector artifacts uses the target name space/name/type to form the primary key. If you configured your system to use DB2 for z/OS version, you must limit the names as follows:

- target name space = 170 bytes

- maximum name = 75 bytes
- maximum type = 10 bytes (this is set by the runtime system)

Note: DB2 for z/OS version 8 does not share this limitation.

If you manually install the business rules dynamic repository, use the provided `configureArtifactRepository` command for DB2.

```
wsadmin -f configureArtifactRepository.jacl properties file -profile default wsadmin
-f configureArtifactRepository.jacl -profile default
```

where:

properties file is the file containing the database properties

The database properties you specify in the properties file are:

cellName

The name of the cell.

Example: `cellName=T40Cell04`

nodeName

The name of the node

Example: `nodeName=T40CellManager04`

profilePath

The profile root name

Example: `profilePath=install_root/AppServer/profiles/profile_name`

profileName

the profile name

Example: `profileName=default`

WBI_HOME

install_root = the installation directory of WebSphere Process Server

Example: `/usr/lpp/zWPS/V6R0`

WAS_HOME

install_root = the installation directory of WebSphere Application Server for z/OS

Example: `/WebSphere/V6R0M0/AppServer`

dbName

database name

Example: `dbName=WPSDB`

dbType

DB2_Universal, DB2UDBOS390_V7_1, DB2UDBOS390_V8_1, DB2_CLI or CLOUDSCAPE

If Cloudscape is specified for the `dbtype`, the following properties must also be provided:

dbServerPort

The Cloudscape server port

Example: `dbServerPort=50000`

dbServerName

The Cloudscape server name

Example: dbServerName=dbserver

dbHostName

The Cloudscape host name

Example: dbHostName=dbserver

dbPassword

The Cloudscape server password

Example: db2passwd

dbClassPath

The path to the Cloudscape JDBC driver

Example: dbClassPath=Install_root/IBM/SQLLIB/java

Note: The Cloudscape JDBC driver shipped with WebSphere Process Server is located in `WPS_INSTALL_ROOT/Cloudscape/lib`. Where `WPS_INSTALL_ROOT` is the directory where you installed WebSphere Process Server.

The following is an example of the `configureArtifactRepository` command.

```
wsadmin -f configureArtifactRepository.jacl install_root/AppServer/profiles/default
```

Installing the business rules dynamic repository for network deployment

Before installing any applications containing business rules, you must install the dynamic repository for business rules.

The dynamic repository supports a centralized configuration that allows you to configure all servers to use the same repository thus allowing all applications to use the same data. This is important for those users who will be updating this data dynamically using the business rule manager tool. The central repository allows:

- changes to be made once
- changes to take effect across all server installations

DB2 for z/OS version 7 limits the size of the primary key to 256 bytes. The dynamic artifact repository for business rule and selector artifacts uses the target name space/name/type to form the primary key. If you configured your system to use DB2 for z/OS version, you must limit the names as follows:

- target name space = 171 bytes
- maximum name = 75 bytes
- maximum type = 10 bytes (this is set by the runtime system)

Note: DB2 for z/OS version 8 does not share this limitation.

If you configure the WPSDB database to be a non-DB2 database as part of the product configuration, you must configure the dynamic repository manually using the `configureArtifactRepository` command. Manual configuration is limited to DB2.

Here is the syntax of the `configureArtifactRepository` command.

```
wsadmin -f configureArtifactRepository.jacl property file -profile {default
| dmgr}
```

where:

properties file = a file containing the database properties
-profile {default | dmgr} = the profile type

The database properties you specify in the properties file are:

cellName

The name of the cell

Example: cellName=T40Cell04

nodeName

The name of the node

Example: nodeName=T40CellManager04

profilePath

The profile root name

Example: profilePath=*install_root*/AppServer/profiles/*profile_name*

profileName

The profile name

Example: profileName=default

WBI_HOME

install_root = the installation directory of WebSphere Process Server

Example: /usr/lpp/zWPS/V6R0

WAS_HOME

install_root = the installation directory of WebSphere Application Server Network Deployment

Example: /WebSphere/V6R0M0/AppServer

dbName

database name

Example: dbName=dynamicDB

dbType

DB2_Universal, DB2UDBOS390_V7_1, DB2UDBOS390_V8_1, or DB2_CLI

Here is an example of the configureArtifactRepository command.

```
wsadmin -f configureArtifactRepository.jacl install_root/AppServer/profiles/default
```

Accessing business rule components

Displaying business rule components is the first step in administering a business rule group. From the display you can export any or all of the business rule groups or display the tables that comprise the business rule groups.

You must be at the administrative console for WebSphere Process Server to perform this task.

Perform this task to determine what business rule groups exist in your server.

1. From the administrative console, select **Servers > Application Servers**.
2. Click *servername* to select the server from the server list that displays business rules.
3. Click **Business rules** under Business Integration.

The console displays a list of all the business rule components defined with a description of each group.

Exporting business rules using the administrative console:

Exporting business rule components creates a file that you import into your development environment, thereby keeping the development artifacts synchronized with the actual production system artifacts.

Before starting this task, you should already have displayed your business rule components as described in “Accessing business rule components” on page 63.

Export business rule components when you have made changes to the business rule tables to synchronize your development environment with your production environment. This task begins with the business rule component display.

1. Choose the business rule groups to export.
Click the check boxes next to the business rule groups and then click **Export**. The browser displays a list of HTML links to the business rule groups you chose. (This is the Business rules to export panel.) Each business rule group has a file extension of `.zip`.
2. Download the files.
Click on each filename and the system prompts you to save the file. When prompted, click **OK** to place the file in your file system.

Note: If you choose to, you can rename the file as you download it.
3. Return to the business rules display panel.
Click **Back** to return to the list of business rule groups.

The system saves file where you specified. You can then copy it to your test system.

You must import this file into your WebSphere Integration Developer environment. See the information center for WebSphere Integration Developer for more information.

Exporting business rules using the command line:

You can also export business rule components using the `exportBusinessRuleArtifacts` command.

Purpose

Use the `exportBusinessRuleArtifacts` command to export business rule components from the command line.

Syntax

```
wsadmin -f exportBusinessRuleArtifacts.jacl target namespace business rule  
group component name<user> -zipf
```

Parameters

where the arguments are:

target namespace = the name of the target namespace of the business rule group component to export

business rule group component name = the name of the business rule group component to export

user = user

-zipf (optional) = the name of a zip file to export the business rule group component ; if not specified, it defaults to *business rule group component name.zip*

Example

```
wsadmin -f exportBusinessRuleArtifacts.jacl  
http://test.oo.brules OnlineOrder admin -zipf c:/artifacts/onlineorder.zip
```

The business rules manager

The business rules manager is a Web-based tool that assists the business analyst in modifying business rule values. This tool is an option of WebSphere Process Server that you install after the initial installation of the server. The business rules manager uses a Web-based interface for you to browse and edit business rules.

How business rules manager works:

The business rules manager component is the main WebSphere Process Server tool that a business analyst uses to manage the rules that run their business.

Use the business rules manager to perform the following tasks

- Open a copy of a business rule from the repository
- Browse and edit a business rule
- Publish a business rule to the repository

Figure 1 shows how the business rule manager calls and publishes rules.

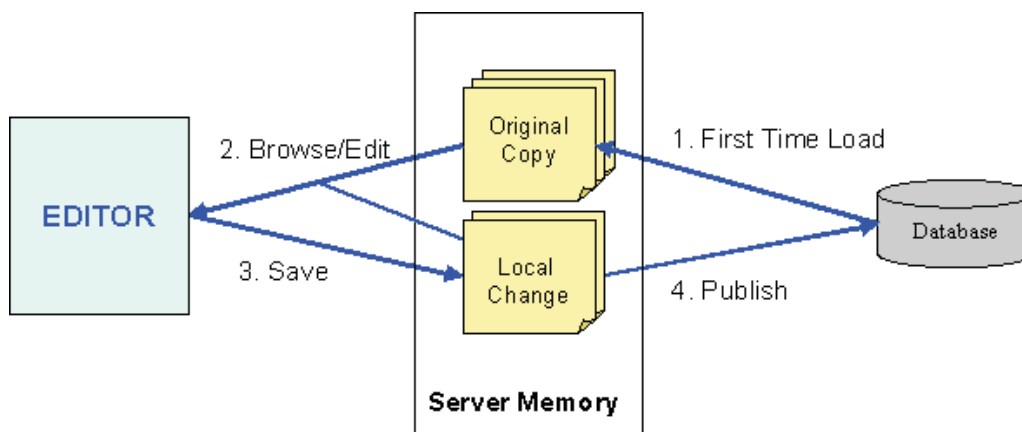


Figure 1. Business rules manager sequence of events

After you log on to the business rules manager, the following events occur when you modify a business rule.

1. When you select a business rule, the business rules manager accesses a rule from the database and stores it in the server memory as an original copy.

2. The rule is available for browsing or editing.
3. You save the business rule as a copy in the local server memory.
4. You publish local copy back to the database.

Business rule templates:

Templates provide the mechanism for Web-based rule authoring in WebSphere Process Server. Business rule templates provide a constrained way of allowing the business analyst to author rules in real-time on an application server. Create templates in WebSphere Integration Developer to provide the means for a business analyst to edit a business rule using the business rule manager. (For more information about building and deploying business rules, see the WebSphere Integration Developer information center.) A template is created for a table cell in a decision table or a rule in a rule set. You must place all rules, condition cases or actions into a template.

Use a template to:

- Modify the values (within the constraints of the rule) of a business rule.
- Create a rule within a decision table or rule set.

Installing the business rules manager:

The business rules manager is installed as a plug-in for WebSphere Process Server. You can either install it using the administrative console or use a `.jacl` script .

To install business rules manager manually from the command prompt, you must execute an installation script in `WAS_HOME/bin` after starting process server.

Use the following command:

```
- run shell/command prompt and change directory to WAS_HOME/bin  
  
/wsadmin.sh -f ./installBRManager.jacl -s servername  
-n nodename -c cellname -r rootname  
-a <applicationname>
```

where:

servername

The name of the application server.

This pair of arguments is required in the Network Deployment configuration.

nodename

The name of the installation node.

This pair of arguments is required in the Network Deployment configuration.

cellname

The name of the installation cell.

This pair of arguments is optional.

rootname

The name of the application root.

This pair of arguments is optional. If missing, the default value of *rootname* is "br/webclient."

applicationname

The name of the application.

If missing, the default value of *applicationname* is "BusinessRulesManager".

Note: If WebSphere Process Server is configured in a single-server environment , all of these pairs of arguments are optional. If WebSphere Process Server is configured for a Network Deployment environment, the only two argument pairs required are:

- -s *servername*
- -n *nodename*

the other argument pairs are optional.

Note: Missing arguments will take the following default values:

- "server1" = *servername*
- "br/webclient" = *rootname*
- "BusinessRulesManager" = *applicationname*

Configuring a server:

You must configure the server that is using the business rules manager.

These are the areas that are addressed when configuring your server.

Note: The next three items need to be addressed only if you are installing business rules manager from the administrative console.

- Prepare userids
You must enable global security when creating userids. Create userids and map them to the role BusinessRuleUser or business analyst. You must assign a role to each userid.
- Set the session tracking mechanism
Use cookies to track sessions.
- Set other parameters
Minimally, set an appropriate session timeout value.
- **Optional:** Enable security on your sever
If you have different roles or userids, you must enable global security when configuring your server.

Configuring a client:

The server configures a client automatically while installing business rule manager.

These two areas are configured in client workstations.

If you encounter problems using business rules manager, examine the following:

- Scripting is enabled in the Web Browser
Business rules manager requires scripting to function.
- Cookies are enabled
When necessary, cookies are used to track the session when you are using business rules manager. Therefore, enable cookies on your browser when tracking sessions. Contact your system administrator if you enable cookies.

Invoking the business rule manager:

Make sure that the server and client are configured correctly for business rules manager.

Contact your system administrator for the URL to start business rules manager.

An example is: `https://hostname:9443/br/webclient` where `hostname:` is the name of the server for example, `server1`.

Accessing business rules manager:

Business rules manager is accessed using a Web browser.

Make sure that both the server and client are configured correctly. See “Installing the business rules manager” on page 66 for more information.

The default URLs for accessing the business rules manager are:

- `https://hostname.9443/br/webclient` (if security is enabled)
- `https://hostname.9080/br/webclient` (if security is not enabled)

where `hostname` is the name of the host.

Note: URLs may vary according to the environment.

Note: The Login page opens only when you have enabled global security on the server. See “Configuring a server” on page 67 for more information. If the global security is not enabled, the Rule books page opens when business rules manager is accessed.

If global security is enabled, follow these steps to login.

1. At the Login page, type your **User ID**.
2. Type your **Password**.
3. Click **Login**. The business rules manager Initial page opens.

The Initial page opens with the existing rule books listed in the Navigation Area. You can now make changes to any business rule listed.

Business rules manager page layout: The page layout on the business rules manager is described below.

Toolbar

The toolbar contents are:

Welcome

Shows the user’s name.

User identification

Provides the current user’s name preceded with **Welcome User Name**.

Logout

Opens the Login page if the global security is enabled.

Note: If you logout without publishing, all unpublished changes are lost.

Help Provides access to information center for the business tools editor. (For more information, see the WebSphere Integration Developer information center.)

Left pane

The left pane of the page contains the navigation tree. This tree enables you to drill-down to the rule level you need. You can expand or collapse a rule book by clicking either the plus (+) or minus (-) beside the resource.

Note: This area is not displayed in any page that is in the edit mode.

Publish and Revert

Opens the Publish and Revert page where you can publish changes to the database or revert back to the original copy that was on the database.

Rule books

Opens the Rule books page.

Note: Listed beneath Rule books is a list of business rule groups. It is the top level of browsing.

Right pane

The right pane of the page is divided into top and bottom sections.

Top section

The top section contains the following elements.

Path information

Provides the name of the rule book and rule page shown in this example.

BusinessRuleGroup01 > Table1_operation1

Rule title

Includes the name of the resource and type of the business rule shown in this example.

Ruleset1 12 - Ruleset

Buttons

The buttons that appear on a page depend on the function of that particular page. This table shows a list of all the possible push buttons that can appear on a page.

Table 7. Button List

Button Name	Function
Back	Returns to the previous page
Edit	Enables editing of the Rule page, decision table, or rule set
Save	Saves the changes and returns to the previous page
Cancel	Discards and changes to the resource and returns to the previous page
Copy	Copies either a decision table, or rule set in order to create a new decision table, or rule set

Table 7. Button List (continued)

Button Name	Function
Publish	Publishes the Rule Page to the server
Revert	Erases all changes and reverts back to the original copies that reside in the database

Message field

Shows either an action that has been taken to the rule or an error that has occurred. This is an example of a status message.

"calculateDiscount" has been temporarily saved.

You may publish the changes from the "Publish and Revert" page.

Bottom section

The bottom section provides the main viewing and editing area of the business rules manager. This section contains the following information.

Note: The information depends on depending on the level of the rule you are viewing.

Rule group

Lists all the rules contained within a rule book.

Rule page

Shows a rule and all the information concerning a particular business rule record in table form. Here is a list of the column headings for a Rule page.

- **Effective date and time**
- **Actions**
- **Description**
- **Rule sets or decision tables**

Publish and Revert

Is a link to the Publish page where you can either publish a rule to the database or revert a rule to the original state. See "How business rules manager works" on page 65 for more information.

Rule books page:

The Rule books page lists the rule groups created for a particular application.

The Navigation tree in the left pane shows the list of rule groups. You can expand the tree by clicking the plus (+) next to the rule books to show all rules. Select a rule book in the left pane navigation tree and all the children rules are listed in the right pane.

The content in the right pane is in a table format has the following column headings.

Business Rule Resources

Lists the name of the business rules, decision tables, and rule sets.

Description

Provides either a brief description or name of the rule, decision table, and rule set.

Action

Is initially empty but when the group is expanded an **Edit** button appears beside each rule.

Rule book page:

When a rule book is selected, the Rule book page opens and lists all the rules in a rule group.

The navigation tree in the left pane shows the list of rule books. You can expand the tree by clicking the plus (+) next to the rule book to show all rules. Select a rule book in the navigation tree and all the children rules are listed in the right pane.

The content in the right pane is in a table format has the following column headings.

Business Rule Resources

Lists the name of the business rules, decision tables, and rule sets.

Description

Provides either a brief description or name of the rule, decision table, and rule set.

Action

Is initially empty but when the group is expanded an **Edit** button appears beside each rule.

Rule page:

The Rule page lists a set of rules and business rule records (decision tables and rule sets) in table form that belong to each rule.

The content in the right pane contains the following elements.

Title Section**Rule Title**

Shows the rule name with the following format, Rule Name-Rule Page. Here is an example of a rule title:

calculateDiscount-Rule Page

Function buttons

Provide specific functionality for that page. Here is a list of the buttons.

- **Back** - returns to the previous page
- **Edit** - opens the edit mode for the selected rule

Messages

Provides a message that signifies a particular error such as incorrect date or time or relating to the status of the rule such as the message shown here.

"calculateDiscount" has been temporarily saved.

You may publish the changes from the "Publish and Revert" page.

General Information section

This section contains the following fields:

Last Published

Shows the date that the rule was last published, the status.

Status Lists the status of the rule, whether it has been published or not.

Description

Provides a brief description of the rule.

Business Rule Selection Records section

Provides a list of effective business rules that are the building blocks of that rule and have the following information:

Start date

Provides the options of either a specific date or "no start date."

Note: The "no start date" signifies that the target rule logic is effective for any date before the end date.

End date

Provides the option of either a specific date or "no end date."

Note: The "no end date" signifies that the rule logic is effective for the start date and any date after it

Effective Business Rule

Provides a decision table or rule set for that rule logic record.

Default Business Rule

Provides the default values for the business rule if no other rule logic is applicable. It is selected when the date does not match any of the other selection records.

Note: The date used for the rule logic selection is either supplied by the application program or the current date of the server. In the latter case, the date is converted to **UTC** before its use and time zone of the server does not matter.

Decision table page:

The Decision table page is opened from the Rule page.

The content in the right pane of the decision table page contains the following elements in table format.

Title section**Table Title**

Shows the rule name with the following format, Decision Table Name-Decision Table (table). Here is an example of a table title:

Table2212-Decision Table

Function buttons

Provide specific functionality for that page. The buttons are:

- **Back** - returns to the previous page.
- **Edit** - opens the edit mode for the selected rule.
- **Copy** - creates a copy of an the rule in order to create a new rule.

Note: You cannot create a new rule in the business rule manager. You must copy an existing rule and then modify the values in order to make a new rule. See *Creating a decision table* for more information.

General Information section

This section contains the following fields:

Last Published

Shows the date that the table was last published, the status.

Status Lists the status of the table, whether it has been published or not.

Description

Provides a brief description of the table.

Decision Table section

This section provides a list of rules in a table format that vary according to the rule.

Columns can have an **orientation** icon that switches orientation from horizontal to vertical. See *“Decision tables”* on page 76

Rule set page:

The Rule set page provides a list of rules for a business rule.

The content in the right pane of the Rule set page contains the following elements in table format.

Top section

This section provides the following fields and information:

Rule Title

Shows the rule name with the following format, “Ruleset Name-Ruleset”. Here is an example of a rule set title:

Ruleset112-Ruleset

Function buttons

Provide the following functionality:

- **Back** - returns to the previous page.
- **Edit** - opens the edit mode for the selected rule.
- **Copy** - creates a copy of an the rule in order to create a new rule.

Note: You cannot create a new rule in the business rules manager. You must copy an existing rule and then modify the values in order to make a new rule. See *Creating a rule set* for more information.

General Information section

This section contains the following fields:

Last Published

Shows the date that the rule set was last published, the status.

Status Lists the status of the rule set, whether it has been published or not.

Description

Provides a brief description of the rule set.

Rules section

Provides a list of rules that are the building blocks of that rule and have the following information.

Name Provides the name of the rule.

Rule Lists the variables, constraints, range, and enumeration that defines the rule.

Working with business rule records

Use business rules manager, to create, modify or delete business rule records.

Creating a business rule record:

You must create business rule records from existing records.

Make sure you are in the **Edit** mode for the rule you want to create.

To create a new business rule record, do the following:

1. Click the **Add Selection Record** button. A new business rule record appears at the bottom of the list with the **Start Date Time** field set a **Jan 1**. A message appears in the **Messages** field stating that the date is invalid.
2. Setting the Start Date/Time:
 - a. Select the **Month**
 - b. Select or type in the **Day**
 - c. Type in the **Year**
 - d. Type in a **time** (in 24-hour format)
3. Setting the End Date/Time.
 - a. Select or type in the **Day**
 - b. Type in the **Year**
 - c. Type in a **time** (in 24-hour format)

Note: Only one rule can be in effect at any one point in time. Rule dates cannot have date/time ranges that overlap.

Note: Gaps in date/time ranges are allowed. If a default business rule is declared, it is used during the gap.

4. Select the **Effective Rule Logic**
5. Click the **Save** button

A message appears in the message field stating that the record has been temporarily saved and you can publish the changes in the **Publish and Revert** page. (See “Publishing business rules” on page 80 for more information.)

Modifying a business rule record:

Modify the date and time values of existing business rule selection records by clicking the **Edit** button for that rule.

Make sure you are in the **Edit** mode in the rule you want to modify by clicking the **Edit** button in the action column. To add, modify or delete a business rule record see “Decision tables” on page 76 or “Rule sets” on page 79 for more information.

To modify a business rule, do the following:

1. Edit the **Start Date/Time** of the selection record:
 - a. Select the **Month**
 - b. Select or type in the **Day**
 - c. Type in the **Year**
 - d. Type in a **time** (in 24-hour format)
2. Edit the **End Date/Time** of the selection record:
 - a. Select or type in the **Day** for the *End Date/Time*
 - b. Type in the **Year** for the *End Date/Time*
 - c. Type in a **time** (in 24-hour format)

Note: Only one rule can be in effect at any one point in time. Rule dates cannot have date/time ranges that overlap.

Note: Gaps in date/time ranges are allowed. If a default business rule is declared, it is used during the gap.

3. Click the **Save** button.

Note: If the **Time/Date** fields are invalid, the fields will turn **red** and a message appears in the message field that the time/dates are invalid.

The record is saved locally and is ready to be published to the database. (See “Publishing business rules” on page 80 for more information.)

See “Splitting dates in business rules” on page 76 for more information on setting business rule dates.

Date/Time selections:

There are four Date/Time selections you can use.

Note: The continuous date selection is only available on the **End Date/Time** field. Using this selection automatically sets the end date to the earliest start date that is later than the selection record.

Specify Date/Time

This selection specifies a date manually.

Continuous

This selection uses an automatic date calculation that sets the end date to the earliest start date that is later than the selection record.

Note: The continuous selection is used when date ranges of two business rule selection records are contiguous. A continuous attribute is set to the end date to the first selection record. When this attribute is set, the start date of the second selection record is set to the end date of the first selection record so that you do not have to specify both dates.

No Start Date or No End Date

The selection does not set a starting or ending boundary, depending on which is selected

Close This selection closes the **End Date/Time** menu.

Splitting dates in business rules:

Splitting a date in a business rule provides a shortcut for modifying a business rule for another purpose.

Select the business rule to be modified and make sure you are in the **edit** mode.

To split a business rule record, do the following:

1. Click the **Split** button for the record to be modified. A new record is created with a start date of Jan 1. The fields will be in red and a message appears in the Message field that there is an invalid date/time field.
2. Select the **Start date/time** for the new record. The **End date/time** for the original record changes from continuous to the **start date** of the previous record and the **End date/time** of the new record changes to the **start date/time** of the next record.
3. Modify the date/times of the record that followed the original record.
4. Modify the **Effective Rule Logic** to fit the needs of the new rule.

The business rule date has been modified.

Decision tables

Modify the values in a decision table using the business rule manager.

A decision table is a business rule record in table format as shown in this figure.

Purchase amount →	<= 1000 dollars	> 1000 dollars
Member Type ↓	Discount ↓	Discount ↓
Gold	8%	10%
Silver	3%	5%

Figure 2. Decision table

The business rules manager provides a Web interface that presents decision tables in tabular form with each variable in **bold** characters. The orientation of conditions and actions are shown by arrows like the ones shown in Figure 1. A decision table functions as a condition with a corresponding action. If a condition is met, then the corresponding action or actions are performed.

Creating a decision table record:


To create a new decision table record, you must first copy an existing decision table.

1. Select a decision table and copy it.
 - a. Click **Copy** in the selected business rule record.
 - b. Click **OK** to copy the record. The **Edit** screen opens with a title, Edit Mode:Copy_of_TableName-Decision Table
 - c. Click **OK** to copy the record.
2. Type in the **name** of the new business rule record in the **Title field**.

3. Type in a short **Description** of the new record.
4. Modify the **values** in each condition. **Note:** To display the parameter settings for each value, place your cursor over a field. A rollover message appears showing the type of variable and its range.
5. Click on the **Up arrow** to place the rule in the correct sequence.
6. Click **Save**.

A message appears in the message field stating that the record has been temporarily saved and you can publish the changes in the Publish and Revert page. See “Publishing business rules” on page 80 for more information.

Modifying a decision table record:

Edit a decision table by directly typing in the values into the input fields or selecting a value from the field’s list box. There is also a special **Edit** menu which appears by clicking on the **Page** icon. 

Note: Reordering the columns or rows only effects the visual presentation of the table and has no effect on the order in which the conditions and actions are processed.

Menu		Condition	Action
Add below	Adds a new condition value below the cell (orientation is vertical)	Yes	
Add to the right	Adds a new condition value on the right of the cell (orientation is horizontal)	Yes	
Change template	Changes the template of a cell	Yes	Yes
Move up	Moves the condition value up one row (orientation is vertical)	Yes	
Move down	Moves the condition value down (orientation is horizontal)	Yes	
Move left	Moves the condition value to the left (orientation is horizontal)	Yes	
Move right	Moves the condition value to the right (orientation is vertical)	Yes	
Delete	Deletes the condition value	Yes	
Close menu	Close the menu	Yes	yes

To modify the values of a decision table do the following:


1. From either the **Navigation** or **Content** sections, select the **Table** to be modified.
2. Click the **Edit** button.
3. Modify the appropriate values.
Either directly type in the values using the field's list box or click the page icon beside the field.
4. Click the **Save** button.

The rule is modified locally and is ready to be published to the server. See "Publishing business rules" on page 80 for more information.

Using special actions menu in a decision table template:

The Decision Table page has a special actions menu to modify a template structure and values.


You need to be in the edit mode for the decision table and need to click the

Special Actions icon () to show the special actions menu. Any variable with the special actions icon can be changed so you can make multiple changes to a template.

The *Special Actions* menu has the following options.

Selection	Action
Add Below	Creates a new row below the present one
Change Template	Allows modifications to the cell values
Move Down	Moves the variable to the row below
Delete	Deletes the variable
Close Menu	Closes the menu and returns to the edit mode.

Modifying a template value of a decision table:

You must be in the edit mode for the decision table and click the special actions icon () to show the special actions menu. You can change any variable with the special actions icon can be changed and you can make multiple changes to a template.

To change the template values, , do the following:

1. Change the template values.
 - a. Select **Change Template** from the menu.
 - b. Type in the new **values** for the that template.
2. Click **Close**.

Note: If you only change one value, you can also click **Change** in the *Action* column.

3. Click **Save**.

The Decision Table template has been modified and is now ready for publishing. See "Publishing business rules" on page 80 for more information.

Rule sets

Create and modify rule sets in the business rules manager.

A rule set is a group of if/then rules that the server executes sequentially. The sequence is from top to bottom of the set. Therefore, when you modify or add a rule, make sure that it is in the correct sequence.

A rule set is created using templates. The template provides the structure that determines how the rule set functions.

Creating a rule set record:

To create a new rule set, you must first copy an existing rule set.

1. Copy the rule set you have selected.
 - a. Click **Copy** in the rule logic record.
 - b. Click **OK** to copy the record. The **Edit** screen opens with a title, Edit Mode:Copy_of_TableName-Ruleset.
 - c. Click **OK** to copy the record.
2. Type in the **name** of the new business rule record in the title field.
3. Type in a short **Description** of the new record.
4. Modify the **values** in each condition.

Note: To display the parameter settings for each value, place your cursor over a field. A rollover message appears showing the type of variable and its range.

5. Click on the **Up** or **Down** arrow to place the rule in the correct sequence.
6. Click **Save**.

A message appears in the message field notifying stating that the record has been temporarily saved and you can publish the changes in the Publish and Revert page.

The rule set is ready for publishing. See “Publishing business rules” on page 80 for more information.

Creating a rule within a rule set from a template:

Create a new rule within a rule set using the rule templates associated with that rule set.

Open the rule set in the edit mode.

To create a new rule from a template, do the following:

1. Click the **New Rule from Template** button. A list of templates appears at the bottom of the page.
2. Create a new rule using an existing template.
 - a. Type **Name** of the new rule in the Name field.
 - b. Type or select the appropriate **variable(s)** or **constraint(s)** for the rule.
 - c. Click **Add**.
3. Click the **Up** or **Down** arrows to place the rule in the proper order.

Note: For a rule set to function correctly, all rules must be in the right order.

4. Click **Save**.

The rule set is ready for publishing. See “Publishing business rules” for more information.

Modifying a rule within a rule set using a template:

Modify a rule in a rule set using templates associated with that rule set.

Open the rule set.

To edit a rule using an existing template, do the following:

1. Edit the **rule set**.
2. Click **Edit** button.
3. Change the **value** in the rule by typing over the existing value or selecting the appropriate value from the pull down list.
4. If necessary, click the **Up** or **Down** arrows to place the rule in the proper order.

Note: For a rule set to function correctly, all rules must be in the right order.

5. Click **Save**.

The modified rule set is ready for publishing. See “Publishing business rules” for more information.

Deleting a business rule, decision table or rule set record

You must be in the edit mode.

Note: Each operation on a business rule group must have at least one business rule associated with it. Attempting to delete all business rules will result in an error.

There are times when you will need to delete a business rule, decision table, or rule set record. To delete a record, do the following:

1. Select the **record** from the business rule list.
2. Click **Delete** button. You are warned that you are about to delete a record.
3. Click **OK**.
4. Click the **Save** button.

The rule is modified locally and is ready to be published to the server. See “Publishing business rules” for more information.

Publishing business rules

Once you have saved any part of a business rule group, you are reminded that the changes have been saved locally and that you need to store the changes to the database. The server publishes changes at the Rule page level. You can publish multiple rule pages at the same time as a single transaction. An error can occur when another user updates the same rule logic or rule book that you are updating. See “Access conflict errors” on page 82 for more information.

To publish a business rule from the business rule group editor, do the following:

1. Click the **Publish and Revert** option in any screen or page that has a *Navigation* area . The Publish and Revert page opens.

2. Select the pages to send to the database by clicking the check box on the left-hand column of the *Content* area. The selected Rule Pages do not appear when you refresh the screen, signifying that they have been written to the database.

The edited rules that have been published are stored on the database that resides on the application server. The business rule is ready to be exported to the server. (See “Exporting business rules using the administrative console” on page 64 for more information.)

Troubleshooting business rules manager

Some areas to examine if you experience problems with business rules manager are: login error, login conflict, and access conflict.

Login error:

Upon logging in, you receive a login error message.

The login error message:

Unable to process login. Please check User ID and password and try again.

This error occurs when global security is enabled and either the userid, the password, or both, are incorrect.

Note: Login errors occur only when global security is enabled.

1. Click **OK** on the error message.
You return to the login page.
2. Enter valid **User ID** and **Password**.
Make sure that Caps Lock key is not on, if passwords are case sensitive.
Make sure the userid and password are spelled correctly.
Check with the system administrator to see that the userid and password are correct.
3. Click the **Login** button.

If you resolve the login error, you will now be able to login to the business rules manager. If the error is not resolved, contact your system administrator.

Login conflict error:

This event occurs when another user with the same userid is already logged in to the application.

The login conflict message is:

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

Usually this error occurs when a user closed the browser without logging out. When this condition occurs, the next attempted login before the session timeout expires results in a login conflict.

Note: Login conflict occurs only when global security is enabled.

There are three options that you can choose.

- Return to the login page.
Use this option if you want to open the application with a different userid.
- Logout the other user with the same userid.
Use this option to logout the other user and start a new session.

Note: Any unpublished local changes made in the other session are lost.

- Inherit the context of the other user with the same userid and logout that user.
Use this option to continue work already in progress. All unpublished local changes in the previous session that have been saved are not lost. The business rules manager opens to the last page displayed in the previous session.

Access conflict errors:

Access conflicts occur when a business rule is updated in the database by one user at the same time another user is updating the same rule.

This error is reported when you publish your local changes to the database.

These are the actions to correct access conflict errors.

- Publish the Rule page.
- Find the source of the business rule that is causing the error and check if your changes on the local machine are still valid. Your change may no longer be required after the changes done by another user.
- If you choose to continue working in the business rule manager, you must reload Rule Pages in the error from the database as your local changes of Rule pages in error are no longer usable. You can still use local changes in other Rule pages that are not in error.
- Reload a Rule page, by clicking **Reload** in the Publish and Revert page of the rule for which the error was reported.

Uninstalling and reinstalling applications using business rules manager

Business rules and selectors add flexibility to your modules. The added flexibility affects how you install or delete a module because the server saves business rules and selectors in a central repository.

Access the business rules manager through the administrative console which provides the ability to dynamically modify business rules and their effective dates. You can also use the administrative console to export business rule data from the repository to the file system.

In addition to the dynamic capabilities of the business rules manager, application installation and removal also affects the data in the repository. For more information on installing and removing applications that use business rules and selectors see the related topics.

Related concepts

Considerations for modules containing business rules and selectors

This topic contains information to consider when you install or delete modules that contain business rules and selectors

Related tasks

Removing business rule and selector data from the repository

When you uninstall an application that uses business rules or selectors, the

server does not remove these artifacts from the repository. This task removes unneeded business rule and selector artifacts from the repository.

Considerations for modules containing business rules and selectors:

This topic contains information to consider when you install or delete modules that contain business rules and selectors

Business rules and selectors add flexibility to your modules. The added flexibility affects how you install or delete a module because the server saves business rules and selectors in a central repository.

Considerations for changing business rules or selectors

You can change business rules and selectors in your production environment without reassembling and reinstalling the affected modules. These changes are made directly to the repository and are not copied into any of the files that contain the business rules or the selectors. After making a change to business rules or selectors, export the business rules or selectors and reimport them into your development environment. If you are unfamiliar with exporting business rules and selectors, see the topics that describe those tasks in this information center.

Considerations for replacing a module containing business rules or selectors

When you replace a module that contains business rules or selectors, the server overwrites the copies of the business rules and selectors in the repository. When you replace a module, any changes that you made dynamically are lost. To prevent that loss, export the business rules and selectors used by the module, reimport them into your development environment, and rebuild the module before replacing the module on your production system.

If you have made changes to the business rules or selectors implemented by one module, other modules running in the server need the current copies of the business rules or selectors. If this is the case, you will have to configure different repositories so that the updated module has no effect on the other modules when you install that module in the server. The topic “Configuring the environment” describes configuring the databases.

Considerations for deleting a module containing business rules or selectors

When you delete a module that contains business rules or selectors from the server, the server does not remove the business rules and selectors from the repository. It keeps these artifacts because it cannot determine if another application or module requires the rules.

If you determine that there is no requirement for a business rule or selector, remove it from the repository. “Removing business rule and selector data from the repository” describes how to clear out unneeded business rules or selectors.

Considerations for database configuration

The dynamic artifact repository for business rule and selector artifacts uses the target name space/name/type to form the primary key. DB2 for z/OS version 7 limits the size of the primary key to 255 bytes.

If you configured your system to use DB2 for z/OS version, you must limit the names as follows:

- target name space = 170 bytes
- maximum name = 75 bytes
- maximum type = 10 bytes

Note: DB2 for z/OS version 8 does not share this limitation.

“Removing business rule and selector data from the repository”

When you uninstall an application that uses business rules or selectors, the server does not remove these artifacts from the repository. This task removes unneeded business rule and selector artifacts from the repository.

Removing business rule and selector data from the repository:

When you uninstall an application that uses business rules or selectors, the server does not remove these artifacts from the repository. This task removes unneeded business rule and selector artifacts from the repository.

Make sure that you uninstall all copies of applications that use the business rules or selectors to be removed from all servers.

When you install an application containing business rule or selector artifacts, the server stores these artifacts in database tables so that you can dynamically update them without changing the application. This also allows other servers to share these artifacts. When you uninstall an application, the server does not automatically remove these artifacts from the database tables because the application may still be installed and running on another server. Deleting the artifacts from the database causes the other running copies of the application to fail when they try to use business rules or selectors.

To delete the unused artifacts from the database, you must do so manually after you uninstall all applications that use them. Remove artifacts using the tools supplied by the database platform of your repository.

1. Locate the database.

Locating the database depends on the database platform.

Database platform

Cloudscape™

Location

WASHOME\cloudscape\ /
databases\RepositoryDB

Other databases

Depends on the location configured during installation and configuration of the server. For example, if you configured the server automatically and selected the default database name, the name of the database is WPSDB.

2. Locate the following database tables from which you will delete rows:

BYTESTORE

The main table that contains the business rule and selector artifacts

BYTESTOREOVERFLOW

The overflow table for the main table

APTIMESTAMP

The installed applications that contain business rule and selector artifacts

3. Delete the artifacts for an application.

Using the tools for your database platform, follow these steps to delete all business rule and selector artifacts for a given application:

 - a. Find all of the rows in the BYTESTORE table where the **APPNAME** column is the same as the name of the application.
 - b. Record the values of the primary key columns for all the rows found. The primary key columns for the BYTESTORE table are **ARTIFACTTNS**, **ARTIFACTNAME**, and **ARTIFACTTYPE**.
 - c. Delete the rows found in step 3a from the BYTESTORE table.
 - d. For each set of primary key values recorded in step 3b, find rows in the BYTESTOREOVERFLOW table that have the same values in the corresponding columns.

Note: For a given set of primary key values, there may be zero, one, or more than one row in the BYTESTOREOVERFLOW table.
 - e. Delete rows found in step 3d from the BYTESTOREOVERFLOW table.
 - f. Delete the row in the APPTIMESTAMP table where the **APPNAME** column equals the name of the application.

You have removed the unneeded business rules and selector artifacts from the database tables.

Administering enterprise applications

Use the console's Enterprise Application page (viewed by clicking **Applications > Enterprise Applications**) to view and administer enterprise applications installed on the server.

To view the values specified for an application's configuration, click the application name from the list. The application details page opens and displays the application's configuration properties and, if appropriate, local topology. From this page, you can modify existing values and link to additional console pages for configuring the application.

To administer an enterprise application, select it by clicking the check box next to its name and then use one of the following buttons:

Table 8. Buttons for administering enterprise applications

Button	Resulting action
Start	<p>Attempts to run the application. After the application starts successfully, the state of the application changes to one of the following:</p> <ul style="list-style-type: none"> • Started—The application has started on all deployment targets • Partial Start—The application is still starting on one or more of the deployment targets
Stop	<p>Attempts to stop the processing of the application. After the application stops successfully, the state of the application changes to one of the following:</p> <ul style="list-style-type: none"> • Stopped—The application has stopped on all deployment targets • Partial Stop—The application is still stopping on one or more of the deployment targets

Table 8. Buttons for administering enterprise applications (continued)

Button	Resulting action
Install	Opens a wizard to help you deploy an enterprise application or module (such as a .jar, .war, or .ear file) onto a server.
Uninstall	Deletes the application from the WebSphere Application Server configuration repository and deletes the application binaries from the file system of all nodes where the application modules are installed after the configuration is saved.
Update	Opens a wizard to help you update application files deployed on a server. You can update the full application, a single module, a single file, or part of the application. If a new file or module has the same name as a file or module already on the server, the new file or module replaces the existing one. Otherwise, it is added to the deployed application.
Remove File	Deletes a file from the deployed application or module. This button deletes the file from the configuration repository and from the file system of all nodes where the file is installed.
Export	Opens the Export Application EAR files page so you can export an enterprise application to an EAR file. Use the Export action to back up a deployed application and to preserve its binding information.
Export DDL	Opens the Export Application DDL files page so you can export DDL files in the EJB modules of an enterprise application.

For more information on administering applications, see the WebSphere Application Server for z/OS Information Center.

Application Scheduler

Application Scheduler allows an administrator to schedule the starting and stopping of applications that are installed on WebSphere Process Server. Use the Application Scheduler panel in the administrative console to control the scheduling of any installed application. Use the Application Scheduler panel in the administrative console to administer these migrated scheduler entries as well.

In a Network Deployment environment, the Application Scheduler is automatically installed for every managed server and cluster member created - no additional action is needed. See Planning for a Network Deployment cell in WebSphere Application Server for z/OS information center for instructions on creating new managed servers and cluster members.

In a standalone server environment, Application Scheduler is optional. While creating the standalone server profile, you select a check box to configure and install Application Scheduler on that server.

Configuring the Application Scheduler for a standalone server

To use Application Scheduler, you must make sure that it is installed. See Working with response files for information on configuring the Application Scheduler.

You need to augment the profile first.

This is an optional component and you must configure Application Scheduler to migrate WebSphere InterChange Server schedule entries to WebSphere Process Server. Complete these steps to install the application server for a standalone server.

See the values for Application Scheduler in the response file and set the property values required in order to use Application Scheduler. For information on properties that pertain to Application Scheduler, see Sample response files.

The installation is complete.

Application Scheduler is ready to use.

Accessing the Application Scheduler

Access Application Scheduler either programmatically using the Application Scheduler Mbean interface, or through the Application Scheduler panel of the administrative console.

For more information on accessing Application Scheduler see:

- “Accessing Application Scheduler using Application Scheduler MBean interface”
- “Displaying scheduler entries using the administrative console” on page 88

Accessing Application Scheduler using Application Scheduler MBean interface

Use the command line to invoke the Application Scheduler MBean

Perform the following to invoke Application Scheduler MBean.

1. Set the properties SOAP_HOSTNAME and SOAP_PORT in the class `com.ibm.wbiserver.migration.ics.Parameters`. This class is in the `migration-wbi-ics.jar` file in the `WAS_HOME\lib` directory. SOAP_HOSTNAME is the name of the host where Application Scheduler is running. SOAP_PORT is the port where the Application Scheduler is running.

```
Parameters.instance.setProperty(Parameters.SOAP_HOSTNAME, "localhost");
Parameters.instance.setProperty(Parameters.SOAP_PORT, "8880");
```

Note: If security is turned on, you must specify a userid and password in the soap properties file found at the location in `WAS_HOME\profiles\profiles\properties\soap.client.props`.

This properties file name must be set in the Parameters instance shown here.

```
Parameters.instance.setProperty(Parameters.SOAP_PROPERTIES,
"WAS_HOME\profiles\profiles\properties\soap.client.props");
```

2. Create an instance of the class `com.ibm.wbiserver.migration.ics.utils.MBeanUtil` that implements calls to the AppScheduler Mbean.

To instantiate an MBeanUtil, you must pass this query string to its constructor that invokes the correct Mbean based on its name, type, server name and node name.

```
protected static final String WEBSHERE_MB_QUERY_CONSTANT = "WebSphere:*";
protected static final String NAME_QUERY_CONSTANT = ",name=";
protected static final String WBI_SCHED_MB_NAME = "WBISchedulerMB1";
protected static final String TYPE_QUERY_CONSTANT = ",type=";
protected static final String WBI_SCHED_MB_TYPE = "WBIScheduler";
protected static final String SERVER_QUERY_CONSTANT = ",process=";
protected static final String NODE_QUERY_CONSTANT = ",node=";
```

```

serverName = "server1";
nodeName = "myNode";

String queryString = new StringBuffer(WEBSPHERE_MB_QUERY_CONSTANT)
    .append(NAME_QUERY_CONSTANT).append(WBI_SCHED_MB_NAME).append(
        TYPE_QUERY_CONSTANT).append(WBI_SCHED_MB_TYPE).append(
        SERVER_QUERY_CONSTANT).append(serverName).append(
        NODE_QUERY_CONSTANT).append(nodeName).toString();

MBeanUtil mbs = new MBeanUtil(queryString.toString());

```

3. Call Mbean methods using the invoke() method of the MbeanUtil instance and pass it the name of the method.

Here is an example of invoking the createSchedulerEntry method of the Scheduler Mbean. The first step is to create a SchedulerEntry and to set various parameters like name, type, version, transition, entry status, recurrence type, recurrence week, recurrence period, initial date, repeat interval and component id.

```

try
{
//First we set up the Schedule entry

ScheduleEntry entry1 = new ScheduleEntry();
entry1.setCName("BPEWebClient_localhost_server1");
entry1.setCType("Application");
entry1.setCVersion("ver1");
entry1.setCTransition("startApplication");
entry1.setSchedulerNumberOfRepeats(3); // Fire Three times
entry1.setScheduleEntryStatus(TaskStatus.SCHEDULED);
entry1.setRType(Recurrence.MINUTES);
entry1.setRWeekNumber(-1);
entry1.setRPeriod(2);
entry1.setInitialDate(new Date(System.currentTimeMillis()+SIXTY_SECOND_OFFSET));
entry1.setRepeatInterval(entry1.getInitialDate(), entry1.getRType(),
    entry1.getRWeekNumber(),
    entry1.getRPeriod());
entry1.setComponentID(entry1.getCName(), entry1.getCType(), entry1.getCVersion(),
    entry1.getCTransition());

```

Then invoke the Mbean's createSchedulerEntry method. We pass it the scheduler entry entry1 as a parameter along with the name of the ScheduleEntry class.

Then invoke the MBean's createScheduleEntry method:

```

mbs.invoke(schedulerExtMBName, "createScheduleEntry", new Object[]{entry1},
    new String[]{"com.ibm.wbiserver.scheduler.common.ScheduleEntry"});

```

Finally, read all the Schedule entries including the one that was just added by calling the readAllScheduleEntries method.

```

result = mbs.invoke("readAllScheduleEntries", null, null);
}
catch (MigrationException e)
{ e.printStackTrace();
}

```

Displaying scheduler entries using the administrative console

Use the Application Scheduler panel of the administrative console to create, modify, or delete scheduler events.

You must be at the administrative console for the server to perform this task.

To display this panel and view existing scheduler events, follow these steps.

1. Select **Servers > Application Servers > ServerName**.
2. Select **Application Scheduler** from **Business Integration**.
3. Select the **scope** (cell, node, server) of the entries to display. The existing scheduled events for that scope are listed.

You can now edit the scheduler event, create a new scheduler event, or delete an existing event.

Creating a scheduled event

Administrative console provides a panel for creating new scheduled events.

To create a new scheduled event, you must be at the Application Scheduler collection panel in the administrative console for the server. See “Displaying scheduler entries using the administrative console” on page 88 for more information.

There are times that you will have to create an event to fit a specific need. To create a new scheduled event, follow these steps.

Note: The fields with an “*” on the panel are required fields.

1. Click **New**. The Add panel opens.
2. Configure the scheduled event.
 - a. Select **Group Application**
 - b. Select **Status**
 - c. Type in the **Initial Date** with the following format *Abbrev month, dd, yyyy* for example, type in **Apr 15, 2005** for April 15, 2005.
 - d. Type in the **Initial Time** using a 12-hour format *hh:mm*

Note: You must also signify either **am** or **pm** and **time zone**.

Note: After you have moved from this field, the **Next Fire Time** is automatically calculated.

- e. Select the **Action**.

Optional: You can also fill in the **Recurrence** parameters.

 - Start-by-period
 - Whether the schedule entry should recur at a specified time.
 - One or more times a minute, hour, day, month or year.
 - A certain day (Sunday thru Saturday) of a certain week (first, second, third, fourth or last) of every one or more months.
 - The last day of every one or more months.
3. Click **Apply** or **OK** to set the event.

Note: To create another event, click **Reset** to clear the panel.

Application Scheduler creates and displays a new scheduled event in the Application Scheduler panel.

Event status and action descriptions:

Each event must have a status and an action.

Status

The status field shows what state the event is in for monitoring purposes. This table lists each status.

Status	Description
Scheduled	A task is to fire at a predetermined date, time and interval. Each subsequent firing time is calculated.
Suspended	A task is suspended and will not fire until its status is changed to scheduled.
Complete	A task is completed.
Cancelled	A task has been cancelled. The task will not fire and it cannot be resumed, but it can be purged.
Invalid	Normally the reason that a task has a status of invalid is because either the task has been purged or the information used to query for that task is invalid.
Running	A task is in the midst of firing. Note: This status should be rarely seen since it just monitors the event for the very short duration that the event is firing.

Action

Each event must have an action associated with it. The action signifies what to do with the event. There are only two actions available for an event:

- **Start Application** - starts all applications that are under the system's deployment manager.
- **Stop Application** - stops all applications that are under the system's deployment manager.

Modifying a scheduled event

Modify migrated or existing scheduled events from the administrative console.

To modify a scheduled event, you must be at the Application Scheduler collection panel in the administrative console for the server. See "Displaying scheduler entries using the administrative console" on page 88 for more information.

There are times that you need to modify an event to fit a need. To modify an event, follow these steps.

1. Click the **Event** you want to modify. The Event panel opens.
2. Modify any of the following fields:

Note: Since all applications on the server are listed, you must be careful when changing the status of an existing event. You may stop an application that is running on the server.

- **Group Application**
- **Status**

- **Initial Date** with the following format *Abbrv month, dd, yyyy*
- **Initial Time** using a 12-hour format *hh:mm*
- **Action**

Optional: You can also fill in the **Recurrence** parameters.

3. Click **Apply** or **OK** to set the modifications for the event.

Note: If you modify a scheduled event, the server assigns a new Schedule Entry ID. The server deletes the currently scheduled event and schedules a new event with the new ID.

The panel displays the modified event with the new ID in the Application Scheduler collection panel.

Deleting a scheduled event

Application Scheduler provides a panel for deleting scheduled events.

To delete a scheduled event, you must be at the Application Scheduler collection panel in the administrative console for the server. See “Displaying scheduler entries using the administrative console” on page 88 for more information.

As events become obsolete, you can delete them from the list of events in the collection panel. Follow these steps to delete a scheduled event.

1. Select the event to be deleted from the **Select** column.
2. Click **Delete**.
3. Click **OK** at the prompt.

The event is deleted.

Administering WebSphere Process Server resources

The administrative interfaces enable you to administer the resources associated with WebSphere

Process Server, including selectors, target components, adapters, and the Extended Messaging Service.

Administering adapters

WebSphere Adapters, version 6.0, and WebSphere Business Integration Adapters (based on WebSphere Business Integration Framework, version 2.6) provide an approach to enterprise information systems (EIS) integration which is application module oriented.

WebSphere Adapters (sometimes referred to as JCA Adapters or J2C Adapters) are compliant with J2EE Connector Architecture (JCA 1.5). JCA is the J2EE standard for EIS connectivity. EIS Import and EIS Export provide SCA application modules with a consistent view of those services outside the module. This allows components to communicate with the variety of EIS systems using the consistent SCA programming model.

WebSphere Adapters are assembled in WebSphere Integration Developer from imported RAR files and then exported as an Enterprise Application Archive (EAR) file and deployed on WebSphere Process Server.

Importing and exporting capabilities within the Service Component Architecture define the external interfaces or access points that a service module has in WebSphere Process Server. Imports and exports can be either to other modules within the same application, or to other applications on EISs. Imports identify services outside a module, making them callable from within the module. Exports allow components in a module to provide their services to external clients. A module level import or export permits modules to access other modules. A system level import or export permits your applications to access applications on EISs as if they were local components, which allows your applications to work with WebSphere Adapters and WebSphere Business Integration Adapters.

WebSphere Business Integration Adapters consist of a collection of software, Application Programming Interfaces (APIs), and tools to enable applications to exchange business data through an integration broker. Each business application requires its own application-specific adapter to participate in the business integration process. You can install, configure, and test the adapter using current WebSphere Business Integration Adapter Framework and Development Kit System Manager tools. You can use WebSphere Integration Developer to import existing business objects and connector configuration files, to generate artifacts, and to assemble the solution for WebSphere Process Server. Operational commands for the WebSphere Business Integration Adapters are part of the WebSphere Process Server administrative console. For more information about working with these adapters and WebSphere Process Server, see the WebSphere Business Integration Adapter information center.

Working with WebSphere Adapters

WebSphere Adapters can be installed and administered using the administrative console.

To perform this task you must have access to the administrative console, and have security permission to alter server settings within the console.

This task should be performed if you have a WebSphere Adapter, that may or may not be embedded within an application. Use the administrative console to work with this adapter.

1. Install the WebSphere Adapter

The process of installing WebSphere Adapters depends on whether they are embedded within an application:

Option	Description
Standalone adapters	Use the administrative console to install the adapter. Note: Standalone WebSphere Adapters are not supported in WebSphere Process Server v6.0.
Embedded adapters	The adapter will be installed as part of the application installation.

2. Administer the adapter

3. Configure J2EE connection factories.

Overview of adapters

There are two types of adapter that can be used in WebSphere Process Server: WebSphere Business Integration Adapters and WebSphere Adapters. An overview of the functionality of these adapters is provided.

Adapters provide connectivity to access data, technologies and protocols that enhance integration infrastructure. They extract data and transaction information from cross-industry and industry-specific packaged applications and connect them to a central server.

Two types of adapter are supported in WebSphere Process Server:

- **WebSphere Business Integration Adapters** consist of a collection of software, application program interfaces (APIs) and tools that enable applications to exchange business data through an integration broker.
- **WebSphere Adapters** (sometimes referred to as Resource Adapters) are the preferred technology. They enable managed, bidirectional connectivity between enterprise information systems (EISs) and J2EE components supported by WebSphere Process Server.

Each business application requires its own application-specific adapter to participate in the business integration process. Adapters allow communication from various Enterprise Applications, such as Seibel or PeopleSoft to WebSphere Process Server.

WebSphere Business Integration Adapters allow multiple applications to communicate to fully integrate your system. Users are encouraged to use WebSphere Adapters which are fully compliant with the Java 2 Enterprise Edition (J2EE) Connector Architecture (JCA) version 1.5. The WebSphere Business Integration Adapters are not JCA-compliant, and run outside the application server.

The WebSphere Business Integration Adapters are the appropriate choice if a WebSphere Adapter does not yet exist for your application. If you have purchased WebSphere Business Integration Adapter licenses you may continue to use them, but you are encouraged to migrate to WebSphere Adapters as these will provide an updated way to connect to popular packaged software.

For information on developing applications using adapters see the WebSphere Integration Developer Information Center.

For information on installing, deploying and configuring WebSphere Adapters see the WebSphere Adapters Information Center.

For information on developing WebSphere Adapters for WebSphere Process Server, see (DeveloperWorks with the WebSphere Adapter Toolkit Guide).

Introduction to WebSphere Adapters: WebSphere Adapters implement the Java 2 Enterprise Edition (J2EE) Connector Architecture (JCA) version 1.5. They are referred to as WebSphere Adapters or Resource Adapters. They manage bidirectional connectivity between enterprise information systems (EISs) and J2EE components supported by WebSphere Process Server.

JCA is designed to facilitate data sharing and to integrate new J2EE applications with legacy and other EISs. JCA stipulates how to develop a WebSphere Adapter that can:

- Plug into any J2EE-compliant application server.
- Connect an application running on that server with an EIS.
- Enable data exchange between the J2EE application and the EIS.

The JCA standard accomplishes this by defining a series of contracts that govern interactions between an EIS and J2EE components within an application server. Fully compliant with the JCA standard, WebSphere Adapters have been developed to run on WebSphere Process Server. Compliance with JCA has several advantages:

- JCA is an open standard,
- JCA is the J2EE standard for EIS connectivity,
- JCA provides a managed framework.

Each WebSphere Adapter is made up of the following:

- **Foundation classes** These implement a generic set of contracts that WebSphere Process Server uses to manage interactions between J2EE applications and all WebSphere Adapters. These quality of service and life cycle management contracts, also known as system contracts, define the service provider interface (SPI). For example, system contracts specify security credential management, connection pooling and transaction management parameters.
- **EIS subclasses** These generic and EIS-specific subclasses define the Common Client Interface (CCI) and EIS API contracts. For example, Activation and Connection Specs allow WebSphere Process Server to manage incoming and outgoing events for the WebSphere Adapter.
- **Enterprise Metadata Discovery** This utility introspects the EIS to generate service data objects (SDOs) and other artifacts that are compiled in a standard Enterprise Application Archive (EAR) file.

A simplified version of the operation of a WebSphere Adapter is shown schematically in figure 1:

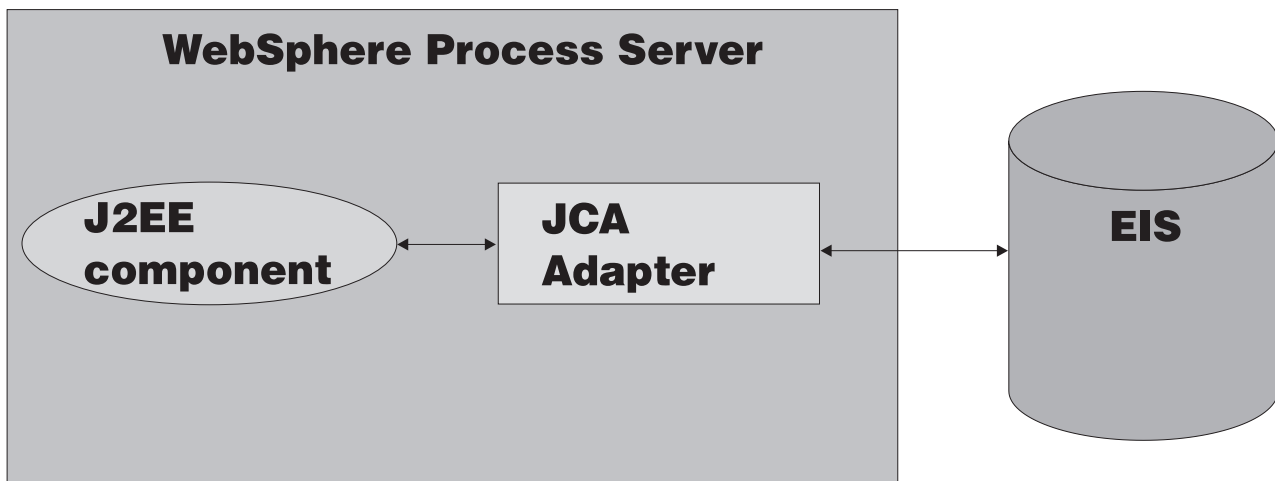


Figure 3. Simplified schematic of a WebSphere Adapter.

Differences between WebSphere Adapters and WebSphere Business Integration Adapters: There are several differences between WebSphere Adapters and WebSphere Business Integration Adapters. These distinctions are most important during development of applications. When deploying applications to a running server, the nature of the adapters used affects some of the steps which need to be followed.

Adapters provide communication mechanisms between enterprise information systems (EISs) and WebSphere applications. To illustrate the operation of the

adapters figures, 1 and 2 provide details of the communication between WebSphere Process Server and the EIS for the two types of adapter.

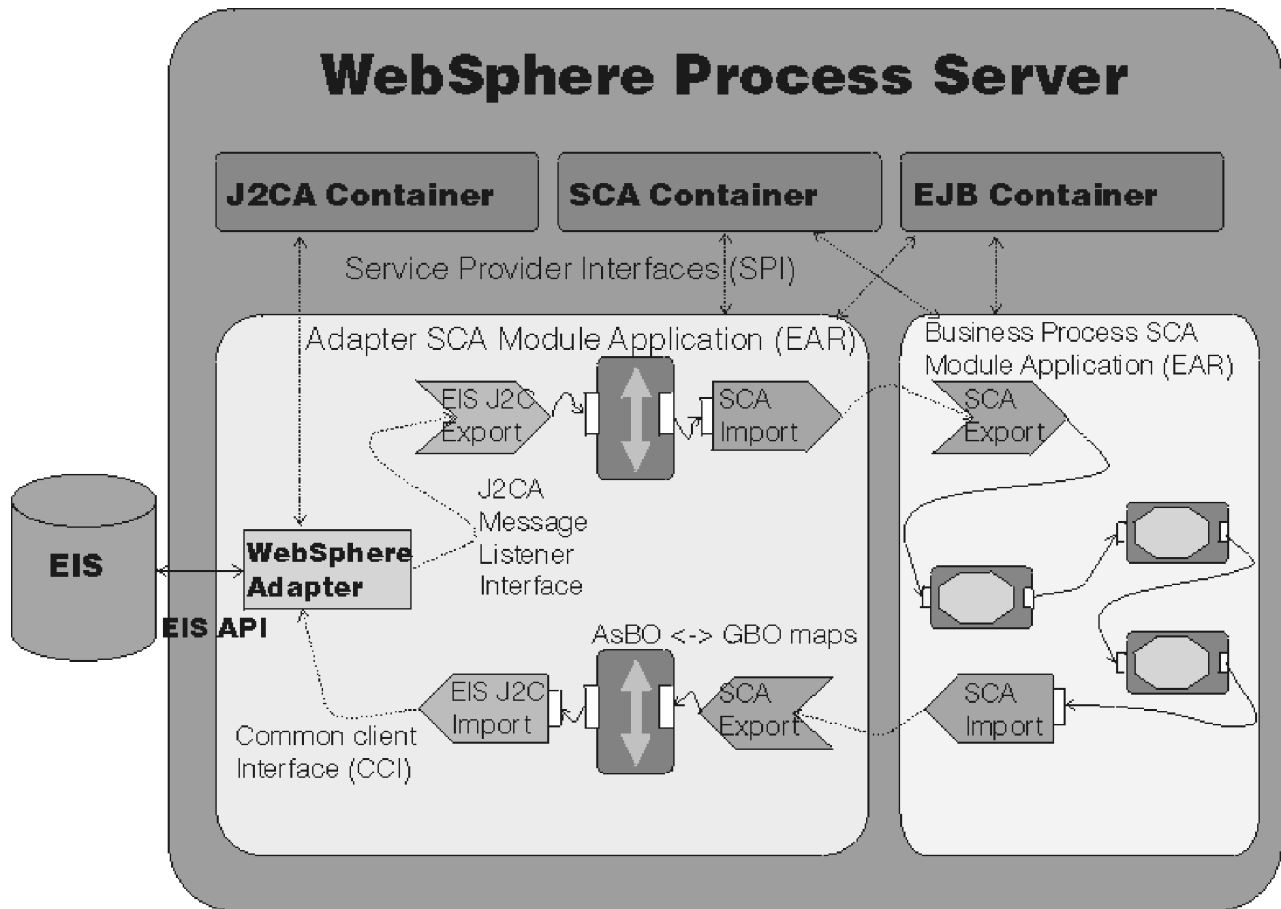


Figure 4. Detailed schematic of a WebSphere Adapter.

Figure 1 depicts a WebSphere Adapter managing the connectivity between a J2EE component supported by WebSphere Process Server and the EIS. The WebSphere Adapter resides inside WebSphere Process Server.

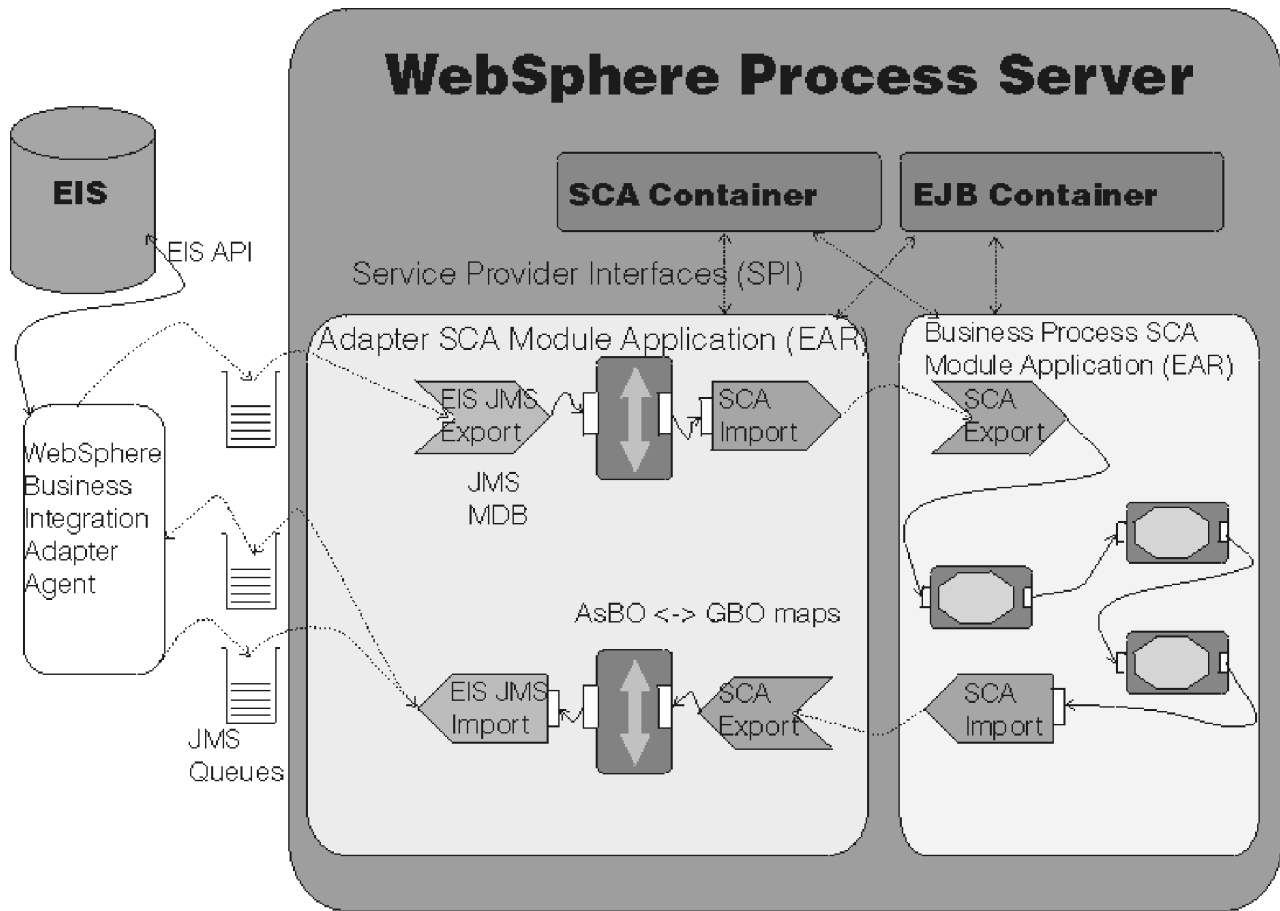


Figure 5. Detailed schematic of a WebSphere Business Integration Adapter.

Figure 2 shows a WebSphere Business Integration Adapter mediating communication between the WebSphere Integration Broker and the EIS. The integration broker communicates with the WebSphere Business Integration Adapter via a Java Messaging Service (JMS) transport layer.

Differences between the two types of adapter include:

Table 9. Differences between WebSphere Adapters and WebSphere Business Integration Adapters.

Feature	WebSphere Adapters	WebSphere Business Integration Adapters
JCA Compliance	Fully JCA compliant (version 1.5).	Not JCA-compliant.
Connectivity Manager	Rely on standard JCA contracts to manage life cycle tasks such as starting and stopping.	Rely on WebSphere Adapter Framework to manage connectivity.
Event Notification	Use an EventStore subclass to retrieve events from an EIS.	Manage event notification using a pollFor Events method.
Request Processing	Clients directly invoke one of several interaction contracts to query or modify data in the EIS.	Rely on an integration server and the WebSphere Adapter Framework to initiate and help process requests.

Table 9. Differences between WebSphere Adapters and WebSphere Business Integration Adapters. (continued)

Feature	WebSphere Adapters	WebSphere Business Integration Adapters
Data Models	Use an Enterprise Metadata Discovery (EMD) utility to parse an EIS and develop Service Data Objects (SDOs) and other useful artifacts. The EMD is part of the WebSphere Adapter implementation.	Use a separate Object Discovery Agent (ODA) to introspect an EIS and generate business object definition schemas.
Integration	Run on WebSphere Process Server.	Reside outside the application server. The server or integration broker communicates with the adapter via a Java Message Service (JMS) transport layer.

WebSphere Adapters are the recommended tool.

Advantages of using WebSphere Adapters: WebSphere Adapters provide several advantages over WebSphere Business Integration Adapters. These advantages are summarized here:

- Integration - WebSphere Adapters are integrated into WebSphere Process Server, WebSphere Business Integration Adapters are outside the application server.
- JCA compliance - only WebSphere Adapters are fully compliant with JCA version 1.5.
- Request processing - WebSphere Adapters do not rely on the WebSphere Adapter Framework nor an integration server for request initiation with Enterprise Information Systems.
- Connectivity - WebSphere Adapters do not rely on the WebSphere Adapter Framework for connectivity, but use JCA contracts to manage life cycle tasks.
- Data models - WebSphere Adapters use an enterprise service discovery wizard to parse an EIS and develop Service Data Objects (SDOs). The enterprise service discovery wizard is part of the WebSphere Adapter implementation. WebSphere Business Integration Adapters use a separate object.
- Event notification - WebSphere Adapters use a subclass of EventStore to retrieve events from an EIS, whereas WebSphere Business Integration Adapters use a pollFor Events method.

There are a limited set of WebSphere Adapters available but it is recommended to use them wherever possible.

Installing applications with embedded WebSphere Adapters

If an application is developed with a WebSphere Adapter embedded, the adapter is deployed with the application. You do not need to install the adapter separately. The steps to install an application with an embedded adapter are described.

This task should only be performed if the application is developed with an embedded WebSphere Adapter.

1. Assemble an application with resource adapter archive (RAR) modules in it. See Assembling applications.
2. Install the application following the steps in Installing a new application. In the Map modules to servers step, specify target servers or clusters for each RAR file. Be sure to map all other modules that use the resource adapters defined in

the RAR modules to the same targets. Also, specify the Web servers as targets that serve as routers for requests to this application. The plug-in configuration file (plugin-cfg.xml) for each Web server is generated based on the applications that are routed through it.

Note: When installing a RAR file onto a server, WebSphere Application Server looks for the manifest (MANIFEST.MF) for the connector module. It looks first in the connectorModule.jar file for the RAR file and loads the manifest from the _connectorModule.jar file. If the class path entry is in the manifest from the connectorModule.jar file, then the RAR uses that class path. To ensure that the installed connector module finds the classes and resources that it needs, check the Class path setting for the RAR using the console. For more information, see Resource Adapter settings and WebSphere relational resource adapter settings.

3. Save the changes. Click **Finish > Save**.
4. Create connection factories for the newly installed application
 - a. Open the administrative console.
 - b. Select the newly installed application Click **Applications > Enterprise Applications > *application name***.
 - c. Click **Connector Modules** in the Related Items section of the page.
 - d. Select the RAR file. Click on *filename.rar*
 - e. Click **Resource adapter** in the Additional Properties section of the page.
 - f. Click **J2C Connection Factories** in the Additional Properties section of the page.
 - g. Click on an **existing connection factory** to update it, or **New** to create a new one.

Note: If the WebSphere Adapter was configured using an EIS Import or EIS Export a ConnectionFactory or ActivationSpec will exist and can be updated.

If you install an adapter that includes native path elements, consider the following: If you have more than one native path element, and one of the native libraries (native library A) is dependent on another library (native library B), then you must copy native library B to a system directory. Because of limitations on most UNIX systems, an attempt to load a native library does not look in the current directory.

After you create and save the connection factories, you can modify the resource references defined in various modules of the application and specify the Java Naming and Directory Interface (JNDI) names of the connection factories wherever appropriate.

Note:

A given native library can only be loaded one time for each instance of the Java virtual machine (JVM). Because each application has its own classloader, separate applications with embedded RAR files cannot both use the same native library. The second application receives an exception when it tries to load the library.

If any application deployed on the application server uses an embedded RAR file that includes native path elements, then you must always ensure that you shut down the application server cleanly, with no outstanding transactions. If the application server does not shut down

cleanly it performs recovery upon server restart and loads any required RAR files and native libraries. On completion of recovery, do not attempt any application-related work. Shut down the server and restart it. No further recovery is attempted by the application server on this restart, and normal application processing can proceed.

WebSphere Adapter:

A WebSphere Adapter (or JCA Adapter, or J2C Adapter) is a system-level software driver that a Java application uses to connect to an enterprise information system (EIS). WebSphere Adapters conform to version 1.5 of the JCA specification.

A WebSphere Adapter plugs into an application server and provides connectivity between the EIS, the application server, and the enterprise application.

An application server vendor extends its system once to support the J2EE Connector Architecture (JCA) and is then assured of seamless connectivity to multiple EISs. Likewise, an EIS vendor provides one standard WebSphere Adapter with the capability to plug into any application server that supports the connector architecture.

WebSphere Process Server provides the WebSphere Relational Resource Adapter (RRA) implementation. This WebSphere Adapter provides data access through JDBC calls to access the database dynamically. The connection management is based on the JCA connection management architecture. It provides connection pooling, transaction, and security support. WebSphere Process Server version 6.0 supports JCA version 1.5.

Data access for container-managed persistence (CMP) beans is managed by the WebSphere Persistence Manager indirectly. The JCA specification supports persistence manager delegation of the data access to the WebSphere Adapter without specific knowledge of the back-end store. For the relational database access, the persistence manager uses the relational resource adapter to access the data from the database. You can find the supported database platforms for the JDBC API at the WebSphere Process Server prerequisite Web site.

IBM supplies resource adapters for many enterprise systems separately from the WebSphere Process Server package, including (but not limited to): the Customer Information Control System (CICS), Host On-Demand (HOD), Information Management System (IMS), and Systems, Applications, and Products (SAP) R/3.

In WebSphere Process Server, EIS Imports and EIS Exports are used to interface with WebSphere Adapters. As an alternative, applications with WebSphere Adapters can be written by developing EJB session beans or services with tools such as Rational Application Developer. The session bean uses the `javax.resource.cci` interfaces to communicate with an enterprise information system through the WebSphere Adapter.

WebSphere Adapter deployment considerations:

The deployment of WebSphere Adapters requires specific options regarding scope.

You can deploy a WebSphere Adapter in two ways, using the administrative console:

- Standalone - the adapter is installed at the node level and is not associated with a specific application.

Note: Deployment of standalone WebSphere Adapters is not supported in WebSphere Process Server v6.0.

- Embedded - the adapter is part of an application, deploying the application also deploys the adapter.

For embedded WebSphere Adapters:

- the RAR file can be application-scoped within an SCA module (with EIS imports or exports).
- the RAR file can be application-scoped within a non-SCA module. The application itself, containing the EIS imports and exports, is a separate SCA module.

You should not install standalone WebSphere Adapters.

Note: The administrative console does not preclude the installation of standalone WebSphere Adapters, but this should not be done. WebSphere Adapters should be embedded in applications.

Only embedded WebSphere Adapters are appropriate for deployment in WebSphere Process Server. Furthermore, deployment of an embedded WebSphere Adapter is only supported for RAR files that are application-scoped within an SCA module; deployment in a non-SCA module is not supported.

Installing Standalone WebSphere Adapters:

WebSphere Adapters should be embedded in applications. Standalone WebSphere Adapters are not supported in WebSphere Process Server v6.0. These instructions are for reference only. If you intend to use a standalone WebSphere Adapter you should install it, as described here. You can alternatively use an embedded adapter, which is installed automatically as part of the installation of the associated application.

You should configure the database before installing the adapter.

You must have access to, and be part of the necessary security role for, the administrative console to perform this task.

1. Open the Install RAR file dialog window.

On the administrative console:

- a. Expand **Resources**
- b. Click **Resource Adapters**
- c. Select the scope at which you want to define this resource adapter. (This scope becomes the scope of your connection factory). You can choose cell, node, cluster, or server.
- d. Click **Install RAR**

A window opens in which you can install a JCA connector and create, for it, a WebSphere Adapter. You can also use the New button, but the New button creates only a new resource adapter (the JCA connector must already be installed on the system).

Note: When installing a RAR file using this dialog, the scope you define on the Resource Adapters page has no effect on where the RAR file is installed. You can install RAR files only at the node level. The node on which the file is installed is determined by the scope on the Install RAR page. (The

scope you set on the Resource Adapters page determines the scope of the new resource adapters, which you can install at the server, node, or cell level.)

2. Install the RAR file

From the dialog, install the WebSphere Adapter in the following manner:

- a. Browse to the location of the JCA connector. If the RAR file is on the local workstation select Local Path and browse to find the file. If the RAR file is on a network server, select **Server path** and specify the fully qualified path to the file.
- b. Click **Next**
- c. Enter the resource adapter name and any other properties needed under General Properties. If you install a J2C Resource Adapter that includes native path elements, consider the following: If you have more than one native path element, and one of the native libraries (native library A) is dependent on another library (native library B), then you must copy native library B to a system directory. Because of limitations on most UNIX systems, an attempt to load a native library does not look in the current directory.
- d. Click **OK**.

WebSphere Adapter applications as members of clusters:

WebSphere Adapter module applications can be cloned as members of a cluster under certain conditions.

WebSphere Adapter module applications can be one of three types, depending on the flow of information through the adapter:

- A WebSphere Adapter application with only EIS exports - only inbound traffic.
- A WebSphere Adapter application with only EIS imports - only outbound traffic.
- A WebSphere Adapter application with both EIS imports and exports - bidirectional traffic.

Clusters are used to provide scalability and availability to your applications in a network deployment environment.

WebSphere Adapter module applications that have either inbound or bidirectional traffic, cannot be cloned as members of a cluster. An application with purely outbound traffic can be cloned as a member of a cluster.

An application that has an inbound or bidirectional WebSphere Adapter (that is, including EIS exports) can still be given availability in a network deployment by use of an external Operating System High Availability (HA) management software package, such as HACMP™, Veritas or Microsoft® Cluster Server.

WebSphere Business Integration Adapter applications as members of clusters:

WebSphere Business Integration Adapter module applications can be cloned as members of a cluster under certain conditions.

WebSphere Business Integration Adapter module applications can be one of three types, depending on the flow of information through the adapter:

- A WebSphere Business Integration Adapter application with only EIS exports - only inbound traffic.

- A WebSphere Business Integration Adapter application with only EIS imports - only outbound traffic.
- A WebSphere Business Integration Adapter application with both EIS imports and exports - bidirectional traffic.

Clusters are used to provide scalability and availability to your applications in a network deployment environment.

WebSphere Business Integration Adapter module applications that have either inbound or bidirectional traffic, cannot be cloned as members of a cluster. An application with purely outbound traffic can be cloned as a member of a cluster.

An application which has inbound or bidirectional WebSphere Business Integration Adapter (i.e., including EIS exports) can still be given availability in a network deployment by use of an external Operating System High Availability (HA) management software package, such as HACMP, Veritas or Microsoft Cluster Server.

Administering a WebSphere Adapter using the administrative console

The administrative console allows the user to administer and configure WebSphere Adapters.

In order to perform this task, you must have security permission to change settings in the administrative console.

When you have installed a WebSphere Adapter, you can administer it using the administrative console.

1. Open the resource adapter console for the adapter you wish to administer.
On the administrative console:
 - a. Expand **Resources**
 - b. Click **Resource Adapters**
 - c. Choose the WebSphere Adapter to administer.
2. Alter desired properties under the General Properties or Additional Properties headings.

When you have finished making changes click the **Apply** button.

Changes are only applied to the local configuration until you save them to the master configuration.

Note: The server may need to be restarted in order for changes to become effective.

WebSphere Adapter administrative console settings:

The administrative console settings for WebSphere Adapters and their default values are described here.

Purpose

Use the administrative console to edit the settings for your WebSphere Adapters.

To view the administrative console:

- Expand **Resources**

- Click **Resource Adapters**
- Choose the resource adapter from the list.

The various fields are described here:

Scope Specifies the level to which this resource definition is visible. For general information, see Administrative console scope settings in the Related Reference section. The Scope field is a read only string field that shows where the particular definition for a resource adapter is located. This is set either when the resource adapter is installed (which can only be at the node level) or when a new resource adapter definition is added.

Name Specifies the name of the resource adapter definition. This property is required. A string with no spaces meant to be a meaningful text identifier for the resource adapter.

Property	Value
Data type	String

Description Specifies a text description of the resource adapter. A free-form text string to describe the resource adapter and its purpose.

Property	Value
Data type	String

Archive path Specifies the path to the RAR file containing the module for this resource adapter. This property is required.

Property	Value
Data type	String

Class path Specifies a list of paths or JAR file names which together form the location for the resource adapter classes. This includes any additional libraries needed by the resource adapter. The resource adapter code base itself is automatically added to the class path, but if anything outside the RAR is needed it can be specified here.

Property	Value
Data type	String

Native path Specifies a list of paths which forms the location for the resource adapter native libraries. The resource adapter code base itself is automatically added to the class path, but if anything outside the RAR is needed it can be specified here.

Property	Value
Data type	String

ThreadPool Alias Specifies the name of a thread pool that is configured in the server that is used by the resource adapter's Work Manager. If there is no thread pool configured in the server with this name, the default configured thread pool instance, named Default, is used. This property is only necessary if this resource adapter uses Work Manager.

Property	Value
Data type	String

Administering Connection Factories

A connection factory is used in communication between an application and enterprise information system (EIS).

An application component uses a connection factory to access a connection instance, which the component then uses to connect to the underlying EIS.

Examples of connections include database connections, Java Message Service connections, and SAP R/3 connections.

Configuring J2EE Connector connection factories in the administrative console:

Connection factories are used in mediating communication between an application and an enterprise information system (EIS). The administrative console can be used to administer connection factories.

This task should be performed if you have a standalone resource adapter. Embedded WebSphere Adapters require slightly different treatment. You must also have security permission to edit the administrative console.

If your application requires access to a non-relational database, you need to configure a resource adapter and a connection factory rather than a JDBC provider and a data source.

1. Select the resource adapter to administer.
 - From the top level of the administrative console, perform the following steps:
 - a. Expand **Resources**
 - b. Select **Resource Adapters**
 - c. Select the WebSphere Adapter that you want to administer.
2. Create a new connection factory for this adapter.
 - Create the new connection factory by:
 - a. Select JCA Connection Factories under the Additional Properties
 - b. Click **New**
 - c. Specify general properties
 - d. Specify authentication preference
 - e. Select aliases for component-managed authentication, container-managed authentication, or both.
 - If no aliases are available, or you want to define a different one,
 - 1) Click **Apply**
 - 2) Click **J2C Authentication Data Entries** under Related Items
 - 3) Click **New**
 - 4) Specify general properties
 - 5) Click **OK**
 - f. Click **OK**
3. Modify the connection pool properties of the newly created connection factory to optimize the behavior of the connection pool manager.
 - Change connection pool values by:
 - a. Select the new Connection Factory

- b. Under the Additional Properties heading select **Connection pool properties**.
- c. Change any desired values by clicking the property name.
- d. Click **OK**
- e. Under Additional Properties click **Custom Properties**.
- f. Click any property name to change its value.
- g. Click **OK**.

Setting general properties for connection pools:

You can assign general settings to a connection pool using the administrative console.

To change general settings for a connection pool you must have security permission to alter values on the administrative console.

In order to assign general properties of a connection pool, you must first create a connection factory for your WebSphere Adapter.

1. Open the Connection Pool Properties panel in the administrative console.
From the top level of the administrative console:
 - a. Expand **Resources**
 - b. Click **Resource Adapter**
 - c. Click on the WebSphere Adapter which has the connection factory you wish to administer.
 - d. Click **J2C Connection Factories** under the Additional Properties heading.
 - e. Select the connection factory to administer.
 - f. Click **Connection Pool Properties** under the Additional Properties heading.
2. General connection pool properties such as timeouts, maximum and minimum connections and purge policy can be altered on this panel. Default values are supplied.
3. After changing the desired properties, click **OK**.

Connection pool settings:

You can change the values of various properties of a connection pool on the Connection Pool Properties panel of the administrative console.

Purpose

The Connection Pool Settings panel is used to assign general property values for connection pools. You can edit properties such as time-outs, purge policies and connection limits.

Configuration Tab

Connection timeout

Specifies the interval, in seconds, after which a connection request times out and a `ConnectionWaitTimeoutException` is thrown.

This value indicates the number of seconds a request for a connection waits when there are no connections available in the free pool and no new connections can be created, usually because the maximum value of connections in the particular connection pool has been reached. For example, if Connection Timeout is set to 300, and the maximum number of connections are all in use,

the pool manager waits for 300 seconds for a physical connection to become available. If a physical connection is not available within this time, the pool manager initiates a `ConnectionWaitTimeout` exception. It usually does not make sense to retry the `getConnection()` method; if a longer wait time is required you should increase the `Connection Timeout` setting value. If a `ConnectionWaitTimeout` exception is caught by the application, the administrator should review the expected connection pool usage of the application and tune the connection pool and database accordingly.

If the `Connection Timeout` is set to 0, the pool manager waits as long as necessary until a connection becomes available. This happens when the application completes a transaction and returns a connection to the pool, or when the number of connections falls below the value of `Maximum Connections`, allowing a new physical connection to be created.

If `Maximum Connections` is set to 0, which enables an infinite number of physical connections, then the `Connection Timeout` value is ignored.

Property	Value
Data type	Integer
Units	Seconds
Default	180
Range	0 to max int

Maximum connections

Specifies the maximum number of physical connections that you can create in this pool.

These are the physical connections to the back-end resource. Once this number is reached, no new physical connections are created and the requester waits until a physical connection that is currently in use returns to the pool, or a `ConnectionWaitTimeout` exception is issued.

For example, if the `Maximum Connections` value is set to 5, and there are five physical connections in use, the pool manager waits for the amount of time specified in `Connection Timeout` for a physical connection to become free.

If `Maximum Connections` is set to 0, the connection pool is allowed to grow infinitely. This also has the side effect of causing the `Connection Timeout` value to be ignored.

If multiple standalone application servers use the same data source, there is one pool for each application server. If clones are used, one data pool exists for each clone. Knowing the number of data pools is important when configuring the database maximum connections.

You can use the Tivoli Performance Viewer to find the optimal number of connections in a pool. If the number of concurrent waiters is greater than 0, but the CPU load is not close to 100%, consider increasing the connection pool size. If the `Percent Used` value is consistently low under normal workload, consider decreasing the number of connections in the pool.

Property	Value
Data type	Integer
Default	10
Range	0 to max int

Minimum connections

Specifies the minimum number of physical connections to maintain.

If the size of the connection pool is at or below the minimum connection pool size, the Unused Timeout thread does not discard physical connections. However, the pool does not create connections solely to ensure that the minimum connection pool size is maintained. Also, if you set a value for Aged Timeout, connections with an expired age are discarded, regardless of the minimum pool size setting.

For example if the Minimum Connections value is set to 3, and one physical connection is created, the Unused Timeout thread does not discard that connection. By the same token, the thread does not automatically create two additional physical connections to reach the Minimum Connections setting.

Property	Value
Data type	Integer
Default	1
Range	0 to max int

Reap time

Specifies the interval, in seconds, between runs of the pool maintenance thread.

For example, if Reap Time is set to 60, the pool maintenance thread runs every 60 seconds. The Reap Time interval affects the accuracy of the Unused Timeout and Aged Timeout settings. The smaller the interval, the greater the accuracy. If the pool maintenance thread is enabled, set the Reap Time value less than the values of Unused Timeout and Aged Timeout. When the pool maintenance thread runs, it discards any connections remaining unused for longer than the time value specified in Unused Timeout, until it reaches the number of connections specified in Minimum Connections. The pool maintenance thread also discards any connections that remain active longer than the time value specified in Aged Timeout.

The Reap Time interval also affects performance. Smaller intervals mean that the pool maintenance thread runs more often and degrades performance.

To disable the pool maintenance thread set Reap Time to 0, or set both Unused Timeout and Aged Timeout to 0. The recommended way to disable the pool maintenance thread is to set Reap Time to 0, in which case Unused Timeout and Aged Timeout are ignored. However, if Unused Timeout and Aged Timeout are set to 0, the pool maintenance thread runs, but only physical connections which timeout due to non-zero timeout values are discarded.

Property	Value
Data type	Integer
Units	Seconds
Default	180
Range	0 to max int

Unused timeout

Specifies the interval in seconds after which an unused or idle connection is discarded.

Set the Unused Timeout value higher than the Reap Timeout value for optimal performance. Unused physical connections are only discarded if the current number of connections exceeds the Minimum Connections setting. For example, if the unused timeout value is set to 120, and the pool maintenance thread is enabled (Reap Time is not 0), any physical connection that remains

unused for two minutes is discarded. Note that accuracy of this timeout, as well as performance, is affected by the Reap Time value. See Reap Time for more information.

Property	Value
Data type	Integer
Units	Seconds
Default	1800
Range	0 to max int

Aged timeout

Specifies the interval in seconds before a physical connection is discarded.

Setting Aged Timeout to 0 supports active physical connections remaining in the pool indefinitely. Set the Aged Timeout value higher than the Reap Timeout value for optimal performance. For example, if the Aged Timeout value is set to 1200, and the Reap Time value is not 0, any physical connection that remains in existence for 1200 seconds (20 minutes) is discarded from the pool. Note that accuracy of this timeout, as well as performance, are affected by the Reap Time value. See Reap Time for more information.

Property	Value
Data type	Integer
Units	Seconds
Default	0
Range	0 to max int

Purge policy

Specifies how to purge connections when a stale connection or fatal connection error is detected.

Valid values are:

- **EntirePool**: All connections in the pool are marked stale. Any connection not in use is immediately closed. A connection in use is closed and issues a stale connection exception during the next operation on that connection. Subsequent `getConnection()` requests from the application result in new connections to the database opening. When using this purge policy, there is a possibility that some connections in the pool are closed unnecessarily when they are not stale. However, this is a rare occurrence. In most cases, a purge policy of `EntirePool` is the best choice.
- **FailingConnectionOnly**: Only the connection that caused the stale connection exception is closed. Although this setting eliminates the possibility that valid connections are closed unnecessarily, it makes recovery from an application perspective more complicated. Only the currently failing connection is closed, so there is a good possibility that the next `getConnection()` request from the application will return a connection from the pool that is also stale, resulting in more stale connection exceptions. The connection pretest function attempts to insulate an application from pooled connections that are not valid. When a back-end resource, such as a database, goes down, pooled connections that are not valid might exist in the free pool. This is especially true when the purge policy is `failingConnectionOnly`; in this case, the failing connection is removed from the pool. Depending on the failure, the remaining connections in the pool might not be valid.

Property	Value
Data type	String
Default	EntirePool

Setting advanced properties for connection pools:

You can assign advanced settings to a connection pool using the administrative console.

To change advanced settings for a connection pool you must have security permission to alter values on the administrative console.

In order to assign advanced properties of a connection pool, you must first create a connection factory for your WebSphere Adapter.

1. Open the Advanced Connection Pool Properties panel in the administrative console.
 - From the top level of the administrative console:
 - a. Expand **Resources**
 - b. Click **Resource Adapter**
 - c. Click on the WebSphere Adapter which has the connection factory you wish to administer.
 - d. Click J2C Connection Factories under the Additional Properties heading.
 - e. Select the connection factory to administer.
 - f. Click Connection Pool Properties under the Additional Properties heading.
 - g. Click Advanced Connection Pool Properties under the Additional Properties heading.
2. Advanced connection pool settings such as partitioning, protection from connection overloading and dealing with connections that are not-responding. Default values are supplied, but tuning these properties is likely to provide improved performance and is recommended.
3. After changing any desired properties click **OK**.

Connection pool advanced settings:

You can change advanced settings for the connection pool on the Advanced Connection Pool Settings panel of the administrative console.

Purpose

The Advanced Connection Pool Settings panel is used to assign values for connection pools. On this panel you can alter properties such as partitioning, avoiding connection overloading and dealing with connections that are not responding.

Configuration Tab

Number of shared partitions

Specifies the number of partitions that are created in each of the shared pools.

Property	Value
Data type	Integer
Default value	0

Property	Value
Range	0 to max int

Number of free pool partitions

Specifies the number of partitions that are created in each of the free pools.

Property	Value
Data type	Integer
Default value	0
Range	0 to max int

Free pool distribution table size

The free pool distribution table size is used for better distribution of the Subject and CRI hash values within a hash table to minimize collisions for faster retrieval of a matching free connection.

If there are many incoming requests with varying credentials, this value can help with the distribution of finding a free pool for a connection for that user. Larger values are more common for installations that have many different credentials accessing the resource. Smaller values (1) should be used if the same credentials apply to all incoming requests for the resource.

Property	Value
Data type	Integer
Default value	0
Range	0 to max int

Surge Threshold

Specifies the number of connections created before surge protection is activated.

Surge protection is designed to prevent overloading of a data source when too many connections are created at the same time. Surge protection is controlled by two properties, surge threshold and surge creation interval.

The surge threshold property specifies the number of connections created before surge protection is activated. After you reach the specified number of connections, you enter surge mode.

The surge creation interval property specifies the amount of time, in seconds, between the creation of connections when in surge mode.

For example, assume the follow settings:

- maxConnections = 50
- surgeThreshold = 10
- surgeCreationInterval = 30 seconds

If the connection pool receives 15 connection requests, 10 connections are created at about the same time. The 11th connection is created 30 seconds after the first 10 connections. The 12th connection is created 30 seconds after the 11th connection. Connections continue to be created every 30 seconds until there are no more new connections needed or you reach the maxConnections value.

Surge connection support starts if the surge threshold is > -1 and the surge creation interval is > 0. The surge threshold property has a default value of -1, which indicates that it is turned off.

wsadmin example

```
$AdminControl getAttribute $objectname surgeCreationInterval
$AdminControl setAttribute $objectname surgeCreationInterval 30
$AdminControl getAttribute $objectname surgeThreshold
$AdminControl setAttribute $objectname surgeThreshold 15
```

Property	Value
Data type	Integer
Default value	-1
Range	-1 to max int

Surge creation interval

Specifies the amount of time between connection creates when you are in surge protection mode.

If the number of connections specified in the surge threshold property have been made, each request for a new connection must wait to be created on the surge creation interval. This property has a default value of 20, which indicates that at least 20 seconds should pass between connections being created. Valid values for this property are any positive integer.

Property	Value
Data type	Integer
Default value	20
Range	0 to max int

Stuck timer time

A stuck connection is an active connection that is not responding or returning to the connection pool. If the pool appears to be stuck (you have reached the stuck threshold), a resource exception is given to all new connection requests until the pool is unstuck. The stuck timer time property is the interval for the timer. This is how often the connection pool checks for stuck connections. The default value is 5 seconds.

If an attempt to change the stuck time, stuck timer time, or stuck threshold properties using the **wsadmin** scripting tool fails, an `IllegalState` exception occurs. The pool cannot have any active requests or active connections during this request. For the stuck connection support to start, all three stuck property values must be greater than 0 and maximum connections must be greater than 0.

Also, the stuck timer time, if it is set, must be less than the stuck time value. In fact, it is suggested that the stuck timer time should be one-quarter to one-sixth the value of stuck time so that the connection pool checks for stuck connections 4 to 6 times before a connection is declared stuck. This reduces the likelihood of false positives.

wsadmin example

```
$AdminControl getAttribute $objectname stuckTime
$AdminControl setAttribute $objectname stuckTime 30
$AdminControl getAttribute $objectname stuckTimerTime
$AdminControl setAttribute $objectname stuckTimerTime 15
$AdminControl getAttribute $objectname stuckThreshold
$AdminControl setAttribute $objectname stuckThreshold 10
```

Property	Value
Data type	Integer
Default value	5
Range	0 to max int

Stuck time

A stuck connection is an active connection that is not responding or returning to the connection pool. If the pool appears to be stuck (you have reached the stuck threshold), a resource exception is given to all new connection requests until the pool is unstuck. The stuck time property is the interval, in seconds, allowed for a single active connection to be in use to the back-end resource before it is considered to be stuck.

Property	Value
Data type	Integer
Default value	0
Range	0 to max int

Stuck Threshold

A stuck connection is an active connection that is not responding or returning to the connection pool. If the pool appears to be stuck (you have reached the stuck threshold), a resource exception is given to all new connection requests until the pool is unstuck. An application can explicitly catch this exception and continue processing. The pool will continue to periodically check for stuck connections when the number of stuck connections is past the threshold. If the number of stuck connections drops below the stuck threshold, the pool will detect this during its periodic checks and enable the pool to begin servicing requests again. The stuck threshold is the number of connections that need to be considered stuck for the pool to be in stuck mode.

Property	Value
Data type	Integer
Default value	0
Range	0 to max int

Configuring connection factories for WebSphere Adapters within applications:

WebSphere Adapters can be standalone or embedded within applications. The process of configuring the connection factories depends whether the adapter is within the application. When a WebSphere Adapter is embedded in the application use the following instructions to configure your connection factory.

You must have the necessary permissions to edit the settings on the administrative console in order to perform this task.

1. Indicate the EAR file which contains the RAR file that you wish to upload and install.

From the top level of the administrative console follow these steps:

- a. Expand **Applications**.
- b. Select **Install New Application**.
- c. Browse to the EAR file on the local or remote system.

Select the radio button associated with the remote or local file system, and then specify a path to the EAR file.

- d. Click Next.
2. Install the application indicating that you wish to map to a J2C connection factory.
Select resource ref mapping to a J2C connection factory and click Next.
3. After the application installs, create and configure a connection factory for the WebSphere Adapter embedded in the newly installed application.
From the top level of the administrative console follow these steps:
 - a. Expand **Applications**.
 - b. Select **Enterprise Applications**.
 - c. Select the newly installed application.
 - d. Click **Connector Modules** under Related Items.
 - e. Select the RAR file.
 - f. Under Additional Properties, select **Resource Adapters**.
 - g. Click **J2C Connection Factories** under Additional Properties.
 - h. Click **New**.
 - i. Specify any necessary general properties.
 - j. **Optional:** Specify an Authentication alias for XA recovery, or use a component-managed authentication alias. This field is only displayed for resources that support XA transactions.
 - k. **Optional:** Select a Component-managed authentication alias.
If a desired alias is not available, or you want to define a different one,
 - 1) Click **Apply**.
 - 2) Click **J2C Authentication Data Entries** under Related Items.
 - 3) Click **New**.
 - 4) Specify general properties.
 - 5) Click **OK** to return to the J2C Connection Factories Settings panel.
Select or define an alias, if any components of your application obtain connections from this connection factory with the empty-argument getConnection() method and with Application or Per connection factory authentication specified in the resource reference.
 - l. Complete the creation of your new connection factory. Click **OK**.
4. **Optional:** Change any connection pool properties of your new connection factory.
From the J2C Connection Factory Collection panel:
 - a. Select the connection factory that you just created
 - b. Open the Connection Pool settings panel Click **Connection pool properties** under the Additional Properties heading.
 - c. Change the values of any properties by clicking on the property name.
 - d. Confirm these changes to the connection pool settings. Click **OK**.
5. **Optional:** Change any custom properties
From the Connection Factory Settings panel of the new connection factory:
 - a. Click **Custom Properties** under the Additional Properties heading.
 - b. Click any property name to change its value.

Note: UserName and Password, if present, are overridden by the component-managed authentication alias you specified in a previous step.

- c. Click **Save**.

J2C connection factory collection:

The J2C connection factory collection panel provides a selectable list of connection factories.

Purpose

Use this panel to select a connection factory, which represents one set of connection configuration values.

Application components such as enterprise beans have resource reference descriptors that refer to the connection factory, not the WebSphere Adapter. The connection factory is really a configuration properties list holder. In addition to the arbitrary set of configuration properties defined by the vendor of the WebSphere Adapter, there are several standard configuration properties that apply to the connection factory. These standard properties are used by the Java 2 Connectors connection pool manager in the application server run time and are not known by the vendor-supplied WebSphere Adapter code.

Connection Factory collection panel

Name

Specifies a list of the connection factory display names.

Property	Value
Data type	String

JNDI Name

Specifies the Java Naming and Directory Interface (JNDI) name of this connection factory.

Property	Value
Data type	String

Description

Specifies a text description of this connection factory.

Property	Value
Data type	String

Category

Specifies a string that you can use to classify or group this connection factory.

Property	Value
Data type	String

J2C connection factory settings:

You specify settings for various properties of a connection factory on the J2C Connection Factory Settings panel of the administrative console.

Purpose

The J2C Connection Factory Settings panel is used to assign general property values for the selected connection factory. You can edit properties such as time-outs, purge policies and connection limits.

Configuration tab

Connection Factory Interface

Specifies the fully qualified name of the Connection Factory Interfaces supported by the resource adapter.

This is a required property. For new objects, the list of available classes is provided by the resource adapter in a drop-down list. After you create the connection factory, the field is a read only text field.

Property	Value
Data type	Drop-down list or text

Authentication Alias for XA recovery

This optional field is used to specify the authentication alias that should be used during XA recovery processing.

If the WebSphere Adapter does not support XA transactions, then this field will not be displayed. The default value will come from the selected alias for application authentication (if specified).

Use Component-managed Authentication Alias

Selecting this radio button specifies that the alias set for component-managed authentication is used at XA recovery time.

Property	Value
Data type	Radio button

Specify:

Selecting this radio button enables you to choose an authentication alias from a drop-down list of configured aliases.

Property	Value
Data type	Radio button

Name

Specifies a list of connection factory display names.

This is a required property.

Property	Value
Data type	String

JNDI Name

Specifies the JNDI name of this connection factory.

For example, the name could be *eis/myECIConnection*.

After you set this value, save it and restart the server. You can see this string when you run the dumpNameSpace tool. This is a required property. If you do not specify a JNDI name, it is filled in by default using the Name field.

Property	Value
Data type	String
Default	<i>eis/display name</i>

Description

Specifies a text description of this connection factory.

Property	Value
Data type	String

Category

Specifies a string that you can use to classify or group this connection factory.

Property	Value
Data type	String

Component-managed Authentication Alias

Specifies authentication data for component-managed signon to the resource.

Choose from aliases defined under Security>JAAS Configuration> J2C Authentication Data.

To define a new alias not already appearing in the pick list:

- Click **Apply** to expose Related Items.
- Click **J2C Authentication Data Entries**.
- Define an alias.
- Click the connection factory name at the top of the J2C Authentication Data Entries page to return to the connection factory page.
- Select the alias.

Property	Value
Data type	Pick-list

Container-managed Authentication Alias

Specifies authentication data (a string path converted to userid and password) for container-managed signon to the resource.

Note: The container-managed authentication alias is superseded by the specification of a login configuration on the resource-reference mapping at deployment time, for components with res-auth=Container.

Choose from aliases defined under Security>JAAS Configuration> J2C Authentication Data.

To define a new alias not yet included in the list:

- Click **Apply** to expose Related Items.
- Click **J2C Authentication Data Entries**.
- Define an alias.

- Click the connection factory name at the top of the J2C Authentication Data Entries page to return to the connection factory page.
- Select the alias.

Property	Value
Data type	Pick-list

Authentication Preference

Specifies the authentication mechanisms defined for this connection factory.

Note: The authentication preference is superseded by the combination of the <res-auth> application component deployment descriptor setting and the specification of a login configuration on the resource-reference mapping at deployment time.

This setting specifies which of the authentication mechanisms defined for the corresponding resource adapter applies to this connection factory. Common values, depending on the capabilities of the resource adapter, are: KERBEROS, BASIC_PASSWORD, and None. If None is chosen, the application component is expected to manage authentication (<res-auth>Application</res-auth>). In this case, the user ID and password are taken from one of the following:

- The component-managed authentication alias
- UserName, Password Custom Properties
- Strings passed on the getConnection method

For example, if two authentication mechanism entries are defined for a resource adapter in the *ra.xml* document:

- <authentication-mechanism-type>BasicPassword</authentication-mechanism-type>
- <authentication-mechanism-type>Kerbv5</authentication-mechanism-type>

the authentication preference specifies the mechanism to use for container-managed authentication. An exception is issued during server startup if a mechanism that is not supported by the WebSphere Adapter is selected.

Property	Value
Data type	Pick-list
Default value	BASIC_PASSWORD

Mapping-Configuration Alias

Allows users to select from the Security > JAAS Configuration > Application Logins Configuration list.

Note: The Mapping-Configuration Alias is superseded by the specification of a login configuration on the resource-reference mapping at deployment time, for components with res-auth=Container.

The DefaultPrincipalMapping JAAS configuration maps the authentication alias to the userid and password. You may define and use other mapping configurations.

Property	Value
Data type	Pick-list

J2C connection factory advanced settings:

You can change advanced settings for the J2C connection factory on the J2C Connection Factory Advanced Settings panel of the administrative console.

Purpose

The J2C Connection Factory Advanced Settings panel is used to assign values for advanced properties of the connection factory. On this panel you can alter properties pertaining to the management of cached handles and logging of missing transactions.

Configuration tab

Manage cached handles

If checked, cached handles (handles held in inst vars in a bean) are tracked by the container.

Property	Value
Data type	Check box

Log missing transaction contexts

If checked, the container logs that there is a missing transaction context when a connection is obtained.

Property	Value
Data type	Check box

Configuring WebSphere Business Integration Adapters

The process of configuring and using the WebSphere Business Integration Adapters is three-fold:

1. Install the application EAR file.
Installation of the application EAR file is accompanied by the creation of all necessary artifacts for the WebSphere Business Integration Adapter to work.
2. Set up the administration of the WebSphere Business Integration Adapter.
In order to administer a WebSphere Business Integration Adapter you must:
 - a. Create a connection queue factory
 - b. Create a WebSphere Business Integration Adapter resource
 - c. Enable the WebSphere Business Integration Adapter service.

Note: When you enable or disable a WebSphere Business Integration Adapter service, you must restart the server in order for the changes to take effect.

3. Manage the WebSphere Business Integration Adapter
Use the administrative console to manage the WebSphere Business Integration Adapters.

Create the artifacts necessary for the WebSphere Business Integration Adapter to work:

To create the artifacts necessary for the WebSphere Business Integration Adapter to work, install the application EAR file.

In order to create the artifacts that you need to use a WebSphere Business Integration Adapter in WebSphere Process Server, you should follow the instructions for installing an application.

Setting up administration of a WebSphere Business Integration Adapter:

You must perform several administrative functions before you can manage a WebSphere Business Integration Adapter. The required steps are described here.

You must install the application EAR file to create the artifacts required for the WebSphere Business Integration Adapter before you perform this task.

In order to have administrative control over a WebSphere Business Integration Adapter you must first perform the following administrative functions.

1. Create a Queue Connection Factory.

From the top level of the administrative console follow these steps:

- a. Expand **Resources**.
- b. Expand **JMS Providers**.
- c. Select **Default Messaging**.
- d. Select **JMS queue connection factory**.

Under the JMS subheading select **JMS queue connection factory**.

- e. Create a new JMS queue connection factory.

Click **New**.

- f. Accept all the default values with the following exceptions:

- Name: QueueCF
- JNDI Name: jms/QueueCF
- Bus. Name: *Your Bus name*

- g. Complete the creation of your new JMS queue connection factory.

Click **OK**.

A message window appears at the top of the JMS queue connection factory panel.

- h. Apply the changes that you have made at the local configuration level to the master configuration.

Click **Save** in the message window.

2. Create a WebSphere Business Integration Adapter resource

From the top level of the administrative console follow these steps:

- a. Expand **Resources**.
- b. Open the WebSphere Business Integration Adapters panel.
Select **WebSphere Business Integration Adapters**.

- c. Create a new WebSphere Business Integration Adapter.

Click **New**.

- d. Accept all the default values with the following exceptions:

- Name: EISConnector
- Queue Connection Factory JNDI Name: jms/QueueCF
- Administration Input Queue JNDI Name: *connectorName/AdminInQueue*
- Administration Output Queue JNDI Name: *connectorName/AdminOutQueue*

- e. Complete the creation of the WebSphere Business Integration Adapter.

Click **OK**.

A message window appears at the top of the WebSphere Business Integration Adapters panel.

- f. Apply the changes that you have made at the local configuration level to the master configuration.

Click **save** in the message window.

3. Enable the WebSphere Business Integration Adapter Service.

From the top level of the administrative console follow these steps:

- a. Expand **Servers**.
- b. Select **Application Servers**.
- c. From the list of servers select a server where the WebSphere Business Integration Adapter Service is to be enabled.

Click on the name of the server that hosts the resources of interest.

- d. Select **WebSphere Business Integration Adapter Service**.

Under the Business Integration subheading on the Configuration tab select **WebSphere Business Integration Adapter Service**.

- e. Ensure that the **Enable Service at startup** check box is selected.

- f. Click **OK**.

A message window appears at the top of the WebSphere Business Integration Adapters panel.

- g. Repeat steps 3c to 3f for each server on which the WebSphere Business Integration Adapter Service is to be enabled.

- h. Apply the changes that you have made at the local configuration level to the master configuration.

Click **save** in the message window.

Note: When you enable or disable a WebSphere Business Integration Adapter service, you must restart the server in order for the changes to take effect.

Managing the WebSphere Business Integration Adapter:

When a WebSphere Business Integration Adapter is running, it can be managed using the Manage the WebSphere Business Integration Adapter resources panel on the administrative console.

The WebSphere Business Integration Adapter must be running in order to be managed.

You can manage a WebSphere Business Integration Adapter from the administrative console. The Manage the WebSphere Business Integration Adapter resources panel allows you to choose one or more resources to manage and perform various administrative actions upon these resources.

1. Select the resource or resources to manage.

From the top level of the administrative console follow these steps:

- a. Expand **Servers**.
- b. Select **Application Servers**.
- c. From the list of servers select the server where the resources you intend to manage reside.

Click on the name of the server that hosts the resources of interest.

- d. Select **WebSphere Business Integration Adapter Service**.
Under the Business Integration subheading on the Configuration tab select **WebSphere Business Integration Adapter Service**.
 - e. Select **Manage the WebSphere Business Integration Adapter resources**.
 - f. From the list of resources choose those that you want to manage.
Select the check boxes associated with the resources you intend to manage.
2. Manage the selected resources.
Click one of the command buttons to act upon the selected resources.

Command	Description
Deactivate	Changes the status of the selected resources from active to paused or inactive.
Activate	Changes the status of the selected resources from inactive to active.
Suspend	Changes the status of the selected resources from active to paused.
Resume	Changes the status of the selected resources from paused to active.
Shutdown	Changes the status of the selected resources from active to unavailable.

Generating service component definitions and the MQClientLink configuration file

Before using WebSphere Business Integration Adapters, it is necessary to generate service component definitions (SCA artifacts) and the MQClientLink configuration file (.wbia file). This is achieved with the WebSphere Business Integration Adapter Artifact Importer in the WebSphere Integration Developer environment.

The WebSphere Integration Developer uses the WebSphere Business Integration Adapter Artifact Importer to discover and import the WebSphere Business Integration Adapter Connector Configuration File and WebSphere Business Integration Adapter Business Objects directory and generate the desired service component definitions supporting the specified interaction-styles for the WebSphere Business Integration Adapter.

Note: This task is performed in WebSphere Integration Developer, and is described here only for reference. See the WebSphere Integration Developer information center for more details.

1. Obtain the necessary configuration files and Business Objects.
2. Use the WebSphere Business Integration Adapter Artifact Importer to generate the necessary service component definitions (SCA artifacts).

The service component definitions (SCA artifacts) and the WebSphere Process Server MQClientLink configuration file (.wbia file) are generated.

When an application is deployed to WebSphere Process Server, the service component definitions and the MQClientLink configuration (.wbia) file are handled automatically by the deployment tools. It is recommended that the configuration file be left in its default state, but it is possible to manually edit the file if required.

Using JMS resources of a generic provider

This topic is the entry-point into a set of topics about enabling WebSphere applications to use JMS resources provided by a generic messaging provider. The term “generic messaging provider” is used to mean messaging providers other than the WebSphere default messaging provider or WebSphere MQ. Note that WebSphere MQ is a separate product, and can only be used as a messaging provider if installed.

You can install a messaging provider other than the default messaging provider. WebSphere MQ is one choice of messaging provider, but is not installed as part of WebSphere Process Server. WebSphere applications can use the JMS 1.1 interfaces or JMS 1.0.2 interfaces to access JMS resources provided by the generic messaging provider, in addition to JMS resources provided by the default messaging provider or by WebSphere MQ (if installed).

You can use the WebSphere Process Server administrative console to administer the JMS connection factories and destinations provided by generic messaging providers.

Defining a generic messaging provider:

Use this task to define a new messaging provider to WebSphere Process Server, for use instead of the default messaging provider or instead of WebSphere MQ, if you have this installed.

Before starting this task, you should have installed and configured the messaging provider and its resources by using the tools and information provided with the messaging provider.

To define a new generic messaging provider to WebSphere Process Server, use the administrative console to complete the following steps:

1. In the navigation pane, click **JMS Providers > Generic**. This displays the existing generic messaging providers in the content pane.
2. To define a new generic messaging provider, click **New** in the content pane. Otherwise, to change the definition of an existing messaging provider, click the name of the provider. This displays the properties used to define the messaging provider in the content pane.
3. Specify the following required properties. You can specify other properties, as described in a later step.
 - Name. The name by which this messaging provider is known for administrative purposes within IBM WebSphere Application Server.
 - External initial context factory. The Java classname of the initial context factory for the JMS provider.
 - External provider URL. The JMS provider URL for external JNDI lookups.
4. **Optional:** Click **Apply**. This enables you to specify additional properties.
5. **Optional:** Specify other properties for the messaging provider. Under Additional Properties, you can use the **Custom Properties** link to specify custom properties for your initial context factory, in the form of standard javax.naming properties.
6. Click **OK**.
7. Save the changes to the master configuration.
8. To have the changed configuration take effect, stop then restart the application server.

You are now in position to configure JMS resources for the generic messaging provider.

Displaying administrative lists of generic messaging resources:

Use the WebSphere Process Server administrative console to display administrative lists of JMS resources provided by a messaging provider other than the default messaging provider or by WebSphere MQ, if you have this installed.

You can use the WebSphere Process Server administrative console to display lists of the following types of JMS resources provided by a generic messaging provider. You can use the panels displayed to select JMS resources to administer, or to create or delete JMS resources (where appropriate).

To display administrative lists of JMS resources for a generic messaging provider, complete the following general steps:

1. Start the WebSphere administrative console.
2. In the navigation pane, click **Resources > JMS Providers > Generic**.
3. If appropriate, in the content pane, change the scope of the generic messaging provider.
4. In the content pane, under Additional Resources, click the link for the type of JMS resource. This displays a list of any existing resources of the selected type. For more information about the settings panels displayed for resources, see the related reference topics.

JMS provider collection:

Use this panel to list JMS providers, or to select a JMS provider to view or change its configuration properties.

To view this administrative console page, click **Resources > JMS providers > Generic**

To view or change the properties of a JMS provider or its resources, select its name in the list displayed.

To define a new generic JMS provider, click **New**.

To act on one or more of the JMS providers listed, click the check boxes next to the names of the objects that you want to act on, then use the buttons provided.

Name The name by which this JMS provider is known for administrative purposes.

Description A description of this JMS provider for administrative purposes.

Generic JMS connection factory collection:

The JMS connection factories configured in the associated generic messaging provider for both point-to-point and publish/subscribe messaging. Use this panel to create or delete JMS connection factories, or to select a connection factory to browse or change its configuration properties.

Collection panel

This panel shows a list of the generic JMS connection factories with a summary of their configuration properties.

To view this administrative console page, use the administrative console to complete the following steps:

1. In the navigation pane, expand **Resources > JMS Providers > Generic**
2. In the content pane, click the name of the generic messaging provider that you want to support the JMS connection factory
3. Under Additional Properties, click **JMS connection factories**

To define a new JMS connection factory, click **New**.

To view or change the properties of a JMS connection factory, select its name in the list displayed.

To act on one or more of the JMS connection factories listed, click the check boxes next to the names of the objects that you want to act on, then use the buttons provided.

Generic JMS connection factory settings:

Use this panel to browse or change the configuration properties of the selected JMS connection factory for use with the associated generic JMS provider. These configuration properties control how connections are created to the JMS destinations on the provider.

Purpose

A JMS connection factory is used to create connections to JMS destinations. The JMS connection factory is created by the associated JMS provider.

To view this page, use the administrative console to complete the following steps:

1. In the navigation pane, expand **Resources > JMS Providers > Generic**.
2. In the content pane, click the name of the messaging provider that you want to support the JMS connection factory.
3. If appropriate, in the content pane, change the scope of the generic messaging provider.
4. Under Additional Properties, click **JMS connection factories**.
5. Click the name of the JMS connection factory that you want to work with.

A JMS connection factory for a generic JMS provider (other than the default messaging provider or, if you have it installed: WebSphere MQ, as a JMS provider) has the following properties:

Properties

Name

The name by which this JMS connection factory is known for administrative purposes within IBM WebSphere Process Server. The name must be unique within the associated messaging provider.

Type

Whether this connection factory is for creating JMS queue destinations or JMS topic destinations.

Select one of the following options:

- **QUEUE** A JMS queue connection factory for point-to-point messaging.
- **TOPIC** A JMS topic connection factory for publish/subscribe messaging.

JNDI name

The JNDI name that is used to bind the connection factory into the WebSphere Process Server name space.

As a convention, use the fully qualified JNDI name; for example, in the form `jms/Name`, where `Name` is the logical name of the resource.

This name is used to link the platform binding information. The binding associates the resources defined by the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

Description

A description of this connection factory for administrative purposes within WebSphere Process Server.

Category

A category used to classify or group this connection factory, for your WebSphere Process Server administrative records.

External JNDI name

The JNDI name that is used to bind the connection factory into the name space of the generic messaging provider.

As a convention, use the fully qualified JNDI name; for example, in the form `jms/Name`, where `Name` is the logical name of the resource.

This name is used to link the platform binding information. The binding associates the resources defined by the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

Component-managed Authentication Alias

This alias specifies a user ID and password to be used to authenticate connection to a JMS provider for application-managed authentication.

This property provides a list of the J2C authentication data entry aliases that have been defined to WebSphere Process Server. You can select a data entry alias to be used to authenticate the creation of a new connection to the JMS provider.

If you have enabled global security for WebSphere Process Server, select the alias that specifies the user ID and password used to authenticate the creation of a new connection to the JMS provider. The use of this alias depends on the resource authentication (`res-auth`) setting declared in the connection factory resource reference of an application component's deployment descriptors.

Container-managed Authentication Alias

This alias specifies a user ID and password to be used to authenticate connection to a JMS provider for container-managed authentication.

This property provides a list of the J2C authentication data entry aliases that have been defined to WebSphere Process Server. You can select a data entry alias to be used to authenticate the creation of a new connection to the JMS provider.

If you have enabled global security for WebSphere Process Server, select the alias that specifies the user ID and password used to authenticate the creation of a new connection to the JMS provider. The use of this alias depends on the resource authentication (res-auth) setting declared in the connection factory resource reference of an application component's deployment descriptors.

Connection pool

Specifies an optional set of connection pool settings.

Connection pool properties are common to all J2C connectors.

The application server pools connections and sessions with the messaging provider to improve performance. You need to configure the connection and session pool properties appropriately for your applications, otherwise you may not get the connection and session behavior that you want.

Change the size of the connection pool if concurrent server-side access to the JMS resource exceeds the default value. The size of the connection pool is set on a per queue or topic basis.

Session pool

An optional set of session pool settings.

This link provides a panel of optional connection pool properties, common to all J2C connectors.

The application server pools connections and sessions with the messaging provider to improve performance. You need to configure the connection and session pool properties appropriately for your applications, otherwise you may not get the connection and session behavior that you want.

Custom properties

An optional set of name and value pairs for custom properties passed to the messaging provider.

Container-managed Authentication Alias

This alias specifies a user ID and password to be used to authenticate connection to a JMS provider for container-managed authentication.

This property provides a list of the J2C authentication data entry aliases that have been defined to WebSphere Process Server. You can select a data entry alias to be used to authenticate the creation of a new connection to the JMS provider.

If you have enabled global security for WebSphere Process Server, select the alias that specifies the user ID and password used to authenticate the creation of a new connection to the JMS provider. The use of this alias depends on the resource authentication (res-auth) setting declared in the connection factory resource reference of an application component's deployment descriptors.

Container-managed Authentication Alias

This alias specifies a user ID and password to be used to authenticate connection to a JMS provider for container-managed authentication.

This property provides a list of the J2C authentication data entry aliases that have been defined to WebSphere Process Server. You can select a data entry alias to be used to authenticate the creation of a new connection to the JMS provider.

If you have enabled global security for WebSphere Process Server, select the alias that specifies the user ID and password used to authenticate the creation of a new connection to the JMS provider. The use of this alias depends on the resource authentication (res-auth) setting declared in the connection factory resource reference of an application component's deployment descriptors.

Configuration tab

Scope

Specifies the level to which this resource definition is visible to applications.

Resources such as messaging providers, namespace bindings, or shared libraries can be defined at multiple scopes, with resources defined at more specific scopes overriding duplicates which are defined at more general scopes.

The scope displayed is for information only, and cannot be changed on this panel. If you want to browse or change this resource (or other resources) at a different scope, change the scope on the messaging provider settings panel, then click Apply, before clicking the link for the type of resource.

Mapping-Configuration Alias

The module used to map authentication aliases.

his field provides a list of the modules that have been configured on the **Global Security > JAAS Configuration > Application Logins Configuration** property.

Generic JMS destination collection:

The JMS destinations configured in the associated messaging provider for point-to-point and publish/subscribe messaging. Use this panel to create or delete JMS destinations, or to select a JMS destination to browse or change its configuration properties.

Collection panel

To view this administrative console page, use the administrative console to complete the following steps:

1. In the navigation pane, expand **Resources > JMS Providers > Generic**
2. In the content pane, click the name of the generic messaging provider that you want to support the JMS destination
3. Under Additional Properties, click **JMS destinations**

To define a new JMS destination, click **New**.

To view or change the properties of a JMS destination, select its name in the list displayed.

To act on one or more of the JMS destinations listed, click the check boxes next to the names of the objects that you want to act on, then use the buttons provided.

Generic JMS destination settings:

Use this panel to browse or change the configuration properties of the selected JMS destination for use with the associated JMS provider.

Purpose

A JMS destination is used to configure the properties of a JMS destination for the associated generic messaging provider (not the default messaging provider or WebSphere MQ). Connections to the JMS destination are created by the associated JMS connection factory.

To view this page, use the administrative console to complete the following steps:

1. In the navigation pane, expand **Resources > JMS Providers > Generic**.
2. In the content pane, click the name of the messaging provider that you want to support the JMS destination.
3. If appropriate, in the content pane, change the scope of the generic messaging provider.
4. Under Additional Properties, click **JMS destinations**.
5. Click the name of the JMS destination that you want to work with.

A JMS destination for use with a generic messaging provider has the following properties.

Properties

Name

The name by which this queue is known for administrative purposes within IBM WebSphere Process Server.

Type

Whether this JMS destination is a queue (for point-to-point) or topic (for publish/subscribe).

Select one of the following options:

- **QUEUE** A JMS queue destination for point-to-point messaging.
- **TOPIC** A JMS topic destination for publish/subscribe messaging.

JNDI name

The JNDI name that is used to bind the connection factory into the WebSphere Process Server name space.

As a convention, use the fully qualified JNDI name; for example, in the form **jms/Name**, where **Name** is the logical name of the resource.

This name is used to link the platform binding information. The binding associates the resources defined by the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

Description

A description of the queue, for administrative purposes.

Category

A category used to classify or group this queue, for your IBM WebSphere Process Server administrative records.

External JNDI name

The JNDI name that is used to bind the queue into the WebSphere Process Server name space.

As a convention, use the fully qualified JNDI name; for example, in the form `jms/Name`, where **Name** is the logical name of the resource.

This name is used to link the platform binding information. The binding associates the resources defined by the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

Configuration tab

Scope

Specifies the level to which this resource definition is visible to applications.

Resources such as messaging providers, namespace bindings, or shared libraries can be defined at multiple scopes, with resources defined at more specific scopes overriding duplicates which are defined at more general scopes.

The scope displayed is for information only, and cannot be changed on this panel. If you want to browse or change this resource (or other resources) at a different scope, change the scope on the messaging provider settings panel, then click **Apply**, before clicking the link for the type of resource.

Configuring JMS resources for a generic messaging provider:

Use the following tasks to configure the JMS connection factories and destinations for a generic messaging provider (not the default messaging provider or WebSphere MQ).

You only need to complete these tasks if your WebSphere Process Server environment uses a messaging provider other than the default messaging provider or WebSphere MQ to support enterprise applications that use JMS. To enable use of such a generic messaging provider, you must have installed and configured the messaging provider, as described in *Defining a new JMS provider to WebSphere Application Server*.

Note: WebSphere MQ is only available as a messaging provider if you have it installed. It is not part of WebSphere Process Server.

To configure JMS resources for a generic messaging provider, complete the subsequent tasks.

Configuring a JMS connection factory, generic JMS provider:

Use this task to browse or change the properties of a JMS connection factory for use with a generic JMS provider, other than the default messaging provider or, if you have it installed, WebSphere MQ.

To configure a JMS connection factory for use with a generic JMS provider, use the administrative console to complete the following steps:

1. Display the generic messaging provider. In the navigation pane, expand **Resources > JMS Providers > Generic**.
2. **Optional:** Change the **Scope** setting to the level at which the connection factory is visible to applications.

3. In the content pane, under Additional Properties, click **JMS connection factories**. This displays a table listing any existing JMS connection factories, with a summary of their properties. .
4. To browse or change an existing JMS connection factory, click its name in the list. Otherwise, to create a new connection factory, complete the following steps:
 - a. Click **New** in the content pane.
 - b. Specify the following required properties. You can specify other properties, as described in a later step.
 - **Name** The name by which this JMS connection factory is known for administrative purposes within IBM WebSphere Process Server.
 - **Type** Select whether the connection factory is for JMS queues (QUEUE) or JMS topics (TOPIC).
 - **JNDI Name** The JNDI name that is used to bind the JMS connection factory into the WebSphere Application Server name space.
 - **External JNDI Name** The JNDI name that is used to bind the JMS connection factory into the name space of the messaging provider.
 - c. Click **Apply**. This defines the JMS connection factory to WebSphere Process Server, and enables you to browse or change additional properties.
5. **Optional:** Change properties for the JMS connection factory, according to your needs.
6. Click **OK**.
7. Save any changes to the master configuration.
8. To effect the changed configuration, stop then restart the application server.

Configuring a JMS destination, generic JMS provider:

Use this task to browse or change the properties of a JMS destination for use with a generic JMS provider, other than the default messaging provider or, if you have it installed, WebSphere MQ.

To configure a JMS destination for use with a generic JMS provider, use the administrative console to complete the following steps:

1. Display the generic messaging provider. In the navigation pane, expand **Resources > JMS Providers > Generic**.
2. **Optional:** Change the **Scope** setting to the level at which the connection factory is visible to applications.
3. In the content pane, under Additional Properties, click **JMS destinations**. This displays a table listing any existing JMS destinations, with a summary of their properties. .
4. To browse or change an existing JMS destination, click its name in the list. Otherwise, to create a new destination, complete the following steps:
 - a. Click **New** in the content pane.
 - b. Specify the following required properties. You can specify other properties, as described in a later step.
 - **Name** The name by which this JMS destination is known for administrative purposes within IBM WebSphere Process Server.
 - **Type** Select whether the destination is for JMS queues (QUEUE) or JMS topics (TOPIC).
 - **JNDI Name** The JNDI name that is used to bind the JMS destination into the WebSphere Process Server name space.

- **External JNDI Name** The JNDI name that is used to bind the JMS destination into the name space of the messaging provider.
- c. Click **Apply**. This defines the JMS destination to WebSphere Process Server, and enables you to browse or change additional properties.
 5. **Optional:** Change properties for the JMS destination, according to your needs.
 6. Click **OK**.
 7. Save any changes to the master configuration.
 8. To effect the changed configuration, stop then restart the application server.

Installing EIS applications

An EIS application module, a service component architecture (SCA) module that follows EIS application module pattern can be deployed to either a J2SE platform or a J2EE platform.

The steps required to deploy an EIS module depend on the platform.

See the subsequent tasks for detailed information.

Deploying an EIS application module to the J2SE platform:

The EIS Module can be deployed to J2SE platform however only EIS Import will be supported.

You need to create an EIS application module with a JMS Import binding in the WebSphere Integration Development environment before commencing this task.

An EIS application module would be furnished with a JMS Import binding when you want to access EIS systems asynchronously through the use of message queues.

Deploying to the J2SE platform is the only instance where the binding implementation can be executed in the non-managed mode. The JMS Binding requires asynchronous and JNDI support, neither of which is provided by the base service component architecture or the J2SE. The J2EE Connector Architecture does not support non-managed inbound communication thus eliminating EIS Export.

When the EIS application module with the EIS Import is deployed to J2SE, in addition to the module dependencies, the WebSphere Adapter used by the import has to be specified as the dependency, in the manifest or any other form supported by SCA.

Deploying an EIS application module to the J2EE platform:

The deployment of EIS module to the J2EE platform results in an application, packaged as an EAR file deployed to the server. All the J2EE artifacts and resources are created, the application is configured and ready to be run.

You need to create an EIS module with a JMS Import binding in the WebSphere Integration Development environment before commencing this task.

The deployment to the J2EE platform creates the following J2EE artifacts and resources:

Table 10. Mapping from bindings to J2EE artifacts

Binding in the SCA module	Generated J2EE artifacts	Created J2EE resources
EIS Import	Resource References generated on the module Session EJB.	ConnectionFactory
EIS Export	Message Driven Bean, generated or deployed, depending on the listener interface supported by the Resource Adapter.	ActivationSpec
JMS Import	Message Driven Bean (MDB) provided by the runtime is deployed, resource references generated on the module Session EJB. Note that the MDB is only created if the import has a receive destination.	<ul style="list-style-type: none"> • ConnectionFactory • ActivationSpec • Destinations
JMS Export	Message Driven Bean provided by the runtime is deployed, resource references generated on the module Session EJB	<ul style="list-style-type: none"> • ActivationSpec • ConnectionFactory • Destinations

When the import or export defines a resource like a ConnectionFactory, the resource reference is generated into the deployment descriptor of the module Stateless Session EJB. Also, the appropriate binding is generated into the EJB binding file. The name, to which resource reference is bound, is either the value of the target attribute, if one is present, or default JNDI lookup name given to the resource, based on the module name and import name.

Upon deployment, the implementation locates the module session bean and uses it to lookup the resources.

During deployment of the application to the server, the EIS installation task will check for the existence of the element resource to which it is bound. If it does not exist, and the SCDL file specifies at least one property, the resource will be created and configured by the EIS installation task. If the resource does not exist, no action is taken, it is assumed that resource will be created before execution of the application.

When the JMS Import is deployed with a receive destination, a Message Driver Bean (MDB) is deployed. It listens for replies to requests that have been sent out. The MDB is associated (listens on) the Destination sent with the request in the JMSreplyTo header field of the JMS message. When the reply message arrives, the MDB uses its correlation ID to retrieve the callback information stored in the callback Destination and then invokes the callback object.

The installation task creates the ConnectionFactory and three destinations from the information in the import file. In addition, it creates the ActivationSpec to enable the runtime MDB to listen for replies on the receive Destination. The properties of the ActivationSpec are derived from the Destination/ConnectionFactory properties. If the JMS provider is a SIBus Resource Adapter, the SIBus Destinations corresponding to the JMS Destination are created.

When the JMS Export is deployed, a Message Driven Bean (MDB) (not the same MDB as the one deployed for JMS Import) is deployed. It listens for the incoming requests on the receive Destination and then dispatches the requests to be processed by the SCA. The installation task creates the set of resources similar to the one for JMS Import, an ActivationSpec, ConnectionFactory used for sending a reply and two Destinations. All the properties of these resources are specified in the export file. If the JMS provider is an SIBus Resource Adapter, the SIBus Destinations corresponding to JMS Destination are created.

Remote deployment of a WebSphere Adapter application

In a network deployment environment it may be convenient to remotely install WebSphere Adapter applications on nodes which reside closer to the EIS (e.g., on the same machine as the EIS) than the node which contains the consuming application (e.g., an application which contains a BPEL process). The best practice for such remote deployment is described herein.

To perform this task you must have a WebSphere Adapter embedded in an application. You must also be working in a network deployment environment with an operating Deployment Manager, and have sufficient privileges to install the application on a remote machine.

This task is appropriate when you have a consuming application, such as a BPEL process application, which requires communication with an EIS mediated by a WebSphere Adapter application in a network deployment environment.

1. Install the consuming application on any node of the cell.
2. Remotely install the WebSphere Adapter application on a node situated in close proximity to the EIS with which it interacts.

The consuming process can access the information in the EIS via the WebSphere Adapter application, without the need for installation of the adapter on every node in the cell.

A BPEL process application requiring communication with an EIS, mediated by a WebSphere Adapter application is a situation where remote deployment of the WebSphere Adapter application is ideal. The WebSphere Adapter application can reside on a node in the cell which is conveniently close to the EIS with which it interacts. The BPEL process application need not reside on the same node. A WebSphere Process Server Network Deployment environment following best practice guidelines would have an architecture similar to that shown in figure 1.

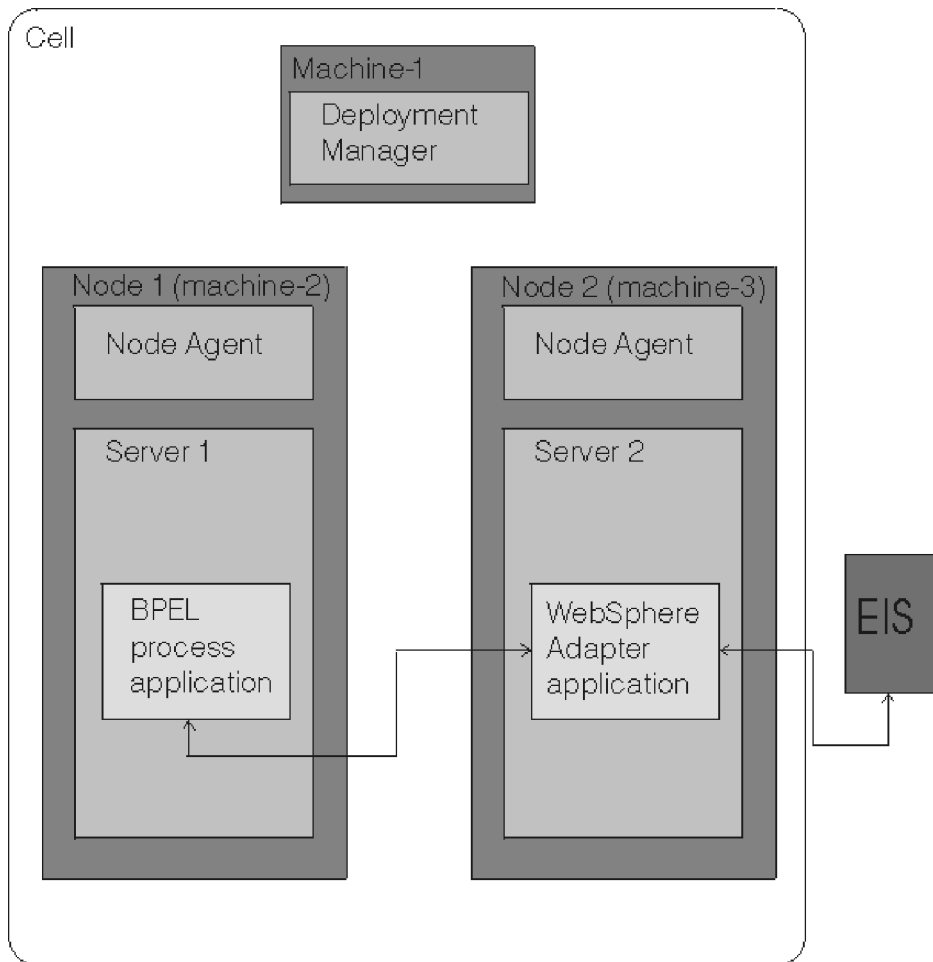


Figure 6. The cell architecture of a remote deployment of a WebSphere Adapter application in a network deployment environment.

JMS Headers

A JMS Header is a service data object (SDO) that contains all the properties of the JMS Message properties.

A JMS Header is a generated service data object containing all the properties of the JMS Message properties from the inbound message or the properties that will be applied to the outbound message.

The JMS Headers are provided using the system programming interface, consequently you set or get the JMS Headers by accessing the service component architecture Message. You cannot set or get the JMS header from a client program. You can however use an ESB Mediation to retrieve the JMS Header values or set them on an outgoing request.

JMS Header and JMS Message properties are only used when the base service component architecture SCDL binding switch is turned on. When the switch is turned on, context information is propagated. By default, this switch is on. To prevent context information propagation change the value to false.

To turn on and off context propagation, which will allow header information to flow to the message or to the target component: specify true or false for the **ContextPropagationEnabled** attribute of the eisBinding, both for Import and Exports. For example:

```
<esbBinding xsi:type="eis:JMSImportBinding" ContextPropagationEnabled="true">
```

Selector component administration overview

As businesses change, the business processes that drive them must change, too. Some of those changes may require that certain processes return different results than as originally designed without changing the design of the process. The selector component provides the framework for that flexibility.

Selector components provide a single interface to a service that may change results based on certain criteria. The selector component comprises an interface and a selector table. The selector table determines, based on a criteria, which component (named the target component) processes the request. The server returns the processing result provided by a target component to the client.

When building a business process, the solution architect identifies the need for a selector component and defines the interface and selector table the selector component uses to complete processing. The tasks involved in developing a selector component are described in the WebSphere Integration Developer information center.

Administering a selector component consists of tasks related to the selector component or tasks related to the selector table.

Displaying selector components

Displaying selector components is the first step in administering selector components. From the display you can export any or all of the selector components or display the selector tables which comprise the selector components.

You must be at the administrative console for the WebSphere Process Server to perform this task.

Perform this task to determine what selector components exist in your server.

1. Click on **Servers** to display the different server types.
2. Click on **Application servers** to expand the Application server list.
3. Click *servername* to select the server from the list server list which has the selectors to display.
4. Click **Selectors** under **Business Integration**.

The console displays a list of all the selector components defined with each components' descriptions.

Displaying selector tables

Displaying selector tables is the first step in administering the tables. The resulting display is a list of target components from which you can alter the processing criteria, change the target component that runs for a specific criterion, add a new target component or delete a target component from the table, thereby removing a criterion.

You must be at the administrative console for the WebSphere Process Server to perform this task.

To perform other selector table-related tasks, display a selector table when you are determining the entries that comprise the table. This task begins after you display selector components.

1. Click on the selector component from the Selector Components display. The browser displays the selector tables in the selected component.
2. Click on one of the selector tables in the display. The browser displays the target components that comprise the selector table.

Changing target components

Changing target components allows you to alter selector component processing by either changing the selection criteria for a specific target component, changing the target component for a selection criteria, or changing both the selection criteria and the target component.

To perform this task, a selector table must exist.

Change a target component to alter the selection criteria or use a different target component for an entry in the selector table.

Important: Before changing target components for long-running applications, stop the application before proceeding. Do not change target components while a long-running application is processing.

1. Display the selector table as described in Displaying selector tables.
2. Click the **Target ID** next to the target component to change. The system displays the Target component details panel.
3. Change the entry.

Portion of entry to change
Target destination

Steps to change

1. Click arrow next to the target component list. The system displays the eligible target components.
2. Select the target component from the list.

Selection criteria

1. Type over either the **Start Date**, **End Date** or both. The date you enter depends on the locale of your system and will be displayed according to the locale format. For the US English locale the format displayed is:
 - Month
 - Day of month
 - Year in YYYY format.
 - Time in HH:MM:SS format
 - Time zone

Portion of entry to change
Target destination and selection criteria

Steps to change

1. Click arrow next to the target component list. The system displays the eligible target components.
2. Select the target component from the list.
3. Type over either the **Start Date**, **End Date** or both. The date you enter depends on the locale of your system and will be displayed according to the locale format. For the US English locale the format displayed is:
 - Month
 - Day of month
 - Year in YYYY format.
 - Time in HH:MM:SS format
 - Time zone
4. **Optional:** Click the **Default** check box to make this the default target component.
If the selection criteria does not fall within the range of any other target components, the selector component uses this target component.
5. Click **Apply** or **OK**.
To continue working in this display, click **Apply**. To return to the target component display, click **OK**.
6. Click **Save** on the target component display to save the changes to the selector table.

The selector table file now contains the updated selection criteria and target components. The selector component uses the updated selector table to process the next request received.

Adding target components

Add a target component when you need additional processing for a different selection criterion than currently exists in the selector table.

To perform this task, a selector table must exist.

Add a target component when you need additional flexibility in your business process. The new components can be added while the selector component is active.

1. Display the selector components as described in *Displaying selector components*.
2. Display the selector table as described in *Displaying selector tables*.
3. Click **New** on the selector table display. The browser displays a pre-filled target component detail panel.
4. Edit the target destination information to fit the application requirements as described in *Changing target components*.
5. Click **OK** to save the target component and return to the target component display.

The selector table now contains the new target components. The selector component uses the updated selector table to process the next request received.

Deleting target components

Deleting target components alters selector component processing by removing the entry in the selector table for a specific selection criterion.

To perform this task, a selector table must exist.

Delete a target component when the processing is no longer required for the business process. After deleting a target component, if a request comes in and it does not match any other specific selection criteria, the default criteria processes the request.

1. Display the target components as described in *Displaying selector tables*. The panel displays the selector table display.
2. Click the check box next to the target components to delete, then click **Delete**. The system updates the panel by displaying the remaining target components.
3. Click **Save**. The system saves the updated selector table with the entries representing the remaining target components.

The selector table file now contains only the remaining target components. The selector component uses the updated selector table to process the next request received.

Exporting selector components

Exporting selector components creates a file that you can then import into your development environment, thereby keeping the development artifacts synchronized with the actual production system artifacts.

Before starting this task, you should already have displayed your selector components as described in *Displaying selector components*.

Export selector components when you have made changes to the selector tables and you need to synchronize your development environment with your production environment. This task begins with the selector component display.

1. Choose the selectors to export.
Click the check boxes next to the selectors and then click **Export**. The browser displays a list of HTML links to the selector components you chose. (This is the **Selectors to Export** panel.) Each selector has a file extension of zip.
2. Download the files.
Click on each filename and the system prompts you to save the file. When prompted, click **OK** to place the file in your file system.

Note: If you choose to, you can rename the file as you download it.
3. Return to the selectors display.
Click **Back** to return to the list of selectors.

The system saves file where you specified. You can then copy it to your test system.

You will need to import this file into your WebSphere Integration Developer environment. See the information center for WebSphere Integration Developer.

Considerations for modules containing business rules and selectors

This topic contains information to consider when you install or delete modules that contain business rules and selectors

Business rules and selectors add flexibility to your modules. The added flexibility affects how you install or delete a module because the server saves business rules and selectors in a central repository.

Considerations for changing business rules or selectors

You can change business rules and selectors in your production environment without reassembling and reinstalling the affected modules. These changes are made directly to the repository and are not copied into any of the files that contain the business rules or the selectors. After making a change to business rules or selectors, export the business rules or selectors and reimport them into your development environment. If you are unfamiliar with exporting business rules and selectors, see the topics that describe those tasks in this information center.

Considerations for replacing a module containing business rules or selectors

When you replace a module that contains business rules or selectors, the server overwrites the copies of the business rules and selectors in the repository. When you replace a module, any changes that you made dynamically are lost. To prevent that loss, export the business rules and selectors used by the module, reimport them into your development environment, and rebuild the module before replacing the module on your production system.

If you have made changes to the business rules or selectors implemented by one module, other modules running in the server need the current copies of the business rules or selectors. If this is the case, you will have to configure different repositories so that the updated module has no effect on the other modules when you install that module in the server. The topic “Configuring the environment” describes configuring the databases.

Considerations for deleting a module containing business rules or selectors

When you delete a module that contains business rules or selectors from the server, the server does not remove the business rules and selectors from the repository. It keeps these artifacts because it cannot determine if another application or module requires the rules.

If you determine that there is no requirement for a business rule or selector, remove it from the repository. “Removing business rule and selector data from the repository” describes how to clear out unneeded business rules or selectors.

Considerations for database configuration

The dynamic artifact repository for business rule and selector artifacts uses the target name space/name/type to form the primary key. DB2 for z/OS version 7 limits the size of the primary key to 255 bytes.

If you configured your system to use DB2 for z/OS version, you must limit the names as follows:

- target name space = 170 bytes
- maximum name = 75 bytes
- maximum type = 10 bytes

Note: DB2 for z/OS version 8 does not share this limitation.

“Removing business rule and selector data from the repository” on page 84
When you uninstall an application that uses business rules or selectors, the server does not remove these artifacts from the repository. This task removes unneeded business rule and selector artifacts from the repository.

Removing business rule and selector data from the repository

When you uninstall an application that uses business rules or selectors, the server does not remove these artifacts from the repository. This task removes unneeded business rule and selector artifacts from the repository.

Make sure that you uninstall all copies of applications that use the business rules or selectors to be removed from all servers.

When you install an application containing business rule or selector artifacts, the server stores these artifacts in database tables so that you can dynamically update them without changing the application. This also allows other servers to share these artifacts. When you uninstall an application, the server does not automatically remove these artifacts from the database tables because the application may still be installed and running on another server. Deleting the artifacts from the database causes the other running copies of the application to fail when they try to use business rules or selectors.

To delete the unused artifacts from the database, you must do so manually after you uninstall all applications that use them. Remove artifacts using the tools supplied by the database platform of your repository.

1. Locate the database.

Locating the database depends on the database platform.

Database platform

Cloudscape

Location

WASHOME\cloudscape\ /
databases\RepositoryDB

Other databases

Depends on the location configured during installation and configuration of the server. For example, if you configured the server automatically and selected the default database name, the name of the database is WPSDB.

2. Locate the following database tables from which you will delete rows:

BYTESTORE

The main table that contains the business rule and selector artifacts

BYTESTOREOVERFLOW

The overflow table for the main table

APPTIMESTAMP

The installed applications that contain business rule and selector artifacts

3. Delete the artifacts for an application.

Using the tools for your database platform, follow these steps to delete all business rule and selector artifacts for a given application:

- a. Find all of the rows in the BYTESTORE table where the **APPNAME** column is the same as the name of the application.
- b. Record the values of the primary key columns for all the rows found. The primary key columns for the BYTESTORE table are **ARTIFACTTNS**, **ARTIFACTNAME**, and **ARTIFACTTYPE**.
- c. Delete the rows found in step 3a on page 85 from the BYTESTORE table.
- d. For each set of primary key values recorded in step 3b on page 85, find rows in the BYTESTOREOVERFLOW table that have the same values in the corresponding columns.

Note: For a given set of primary key values, there may be zero, one, or more than one row in the BYTESTOREOVERFLOW table.

- e. Delete rows found in step 3d on page 85 from the BYTESTOREOVERFLOW table.
- f. Delete the row in the APPTIMESTAMP table where the **APPNAME** column equals the name of the application.

You have removed the unneeded business rules and selector artifacts from the database tables.

Overview of targets

Targets provide additional flexibility by providing the capability of modifying processing by changing the target configured for a reference.

A component can call a component in another module thereby reusing existing logic to minimize the time and cost in building an application. WebSphere Process Server provides additional flexibility through targets. Targets allow an installed application to benefit from advances in processing by allowing the application to change the endpoint of a cross-module invocation, using the administrative console, without rewriting or redeploying the application.

To take advantage of the flexibility provided, you must understand how the system names the targets. The link from the calling module must connect to the correct target.

Target names

Target names are derived from how the calling component invokes the target. The names have the following format:

Invocation type

Name format

Synchronous

A name that follows the Java Naming and Directory Interface (JNDI) format, for example:

folder/export/fullpath_to_target/target_component_name

Asynchronous

A name with the format

*folder/calling_component_name/
full_path_to_target_component/target_component_name*

Multiple destinations

This name is the same as an asynchronous invocation but the target actually sends a message to multiple destination components.

Related tasks

“Changing targets”

Changing the target of a reference provides applications with the flexibility of taking advantage of advances in components as they happen without recompiling and reinstalling the application.

Changing targets

Changing the target of a reference provides applications with the flexibility of taking advantage of advances in components as they happen without recompiling and reinstalling the application.

Before changing the target for a reference you must:

- Make sure the new target uses the same data object type
- Know whether the module is synchronously or asynchronously invoking the target
- Know whether the reference targets a single or multiple services

Change the target of an import from a module when another service with the same interface as the original target provides new or improved functionality that your module can use.

1. Stop the module that contains the reference that you are changing.
 - a. Using the administrative console, display the Service Component Architecture (SCA) modules.
Navigate to this panel using **Applications > SCA Modules**
 - b. Select your module and press **Stop**. The display updates to show the application as stopped.

2. Change the target destination of the reference.

How you make the change depends on how the module invokes the target.

Type of invocation	How to change
Single target service	<ol style="list-style-type: none">1. Using the administrative console, display the SCA Modules. Navigate to the panel using Applications > SCA Modules.2. From the displayed list, select the module that contains the import that references the target to change.3. Expand the list of imports by clicking the plus sign (+) next to Imports.4. Select the import to change from the list.5. In the Target area, select the Module from the list.6. After the Export list refreshes, select the export for the new target.7. Save the change by clicking OK.

Type of invocation	How to change
Multiple target services	<ol style="list-style-type: none"> 1. Display the buses on the system on which the module resides. Navigate to the panel using Service Integration > Buses. 2. Select the SCA.System.cellname.Bus 3. Display the destination targets for the bus by clicking Destinations. 4. Select the destination that represents the import that connects the calling module to the targets. This identifier will contain the word import. 5. Display the list of properties by clicking Context properties. 6. Select the property to change by clicking on the targets property in the list. 7. Change the Context value field to the new destination targets. 8. Return to the Context properties panel by clicking OK. 9. Save the change by clicking OK.

3. Save your changes. Click **Save** when prompted.

Start the module and make sure the module receives the expected results.

Deleting J2C activation specifications

The system builds J2C applications specifications when installing an application that contains services. There are occasions when you must delete these specifications before reinstalling the application.

If you are deleting the specification because of a failed application installation, make sure the module in the Java Naming and Directory Interface (JNDI) name matches the name of the module that failed to install. The second part of the JNDI name is the name of the module that implemented the destination. For example in `sca/SimpleBOCrsmA/ActivationSpec`, **SimpleBOCrsmA** is the module name.

Delete J2C activation specifications when you inadvertently saved a configuration after installing an application that contains services and do not require the specifications.

1. Locate the activation specification to delete.

The specifications are contained in the resource adapter panel. Navigate to this panel by clicking **Resources > Resource adapters**.

 - a. Locate the **Platform Messaging Component SPI Resource Adapter**.

To locate this adapter, you must be at the **node** scope for a stand alone server or at the **server** scope in a Network Deployment environment.
2. Display the J2C activation specifications associated with the Platform Messaging Component SPI Resource Adapter.

Click on the resource adapter name and the next panel displays the associated specifications.
3. Delete all of the specifications with a **JNDI Name** that matches the module name that you are deleting.
 - a. Click the check box next to the appropriate specifications.

- b. Click **Delete**.

The system removes selected specifications from the display.

Save the changes.

Deleting SIBus destinations

SIBus destinations are the connections that make services available to applications. There will be times that you will have to remove destinations.

If you are deleting the destination because of a failed application installation, make sure the module in the destination name matches the name of the module that failed to install. The second part of the destination is the name of the module that implemented the destination. For example in `sca/SimpleBOCrsmA/component/test/sca/cros/simple/cust/Customer`, **SimpleBOCrsmA** is the module name.

Delete SIBus destinations when you inadvertently saved a configuration after installing an application that contains services and you no longer need the destinations.

Note: This task deletes the destination from the SCA system bus only. You must remove the entries from the application bus also before reinstalling an application that contains services (see Deleting J2C activation specifications in the Administering section of this information center).

1. Log into the administrative console.
2. Display the destinations on the SCA system bus.
Navigate to the panel by clicking **Service integration > buses**
3. Select the SCA system bus destinations.
In the display, click on **SCA.SYSTEM.cellname.Bus**, where *cellname* is the name of the cell that contains the module with the destinations you are deleting.
4. Delete the destinations that contain a module name that matches the module that you are removing.
 - a. Click on the check box next to the pertinent destinations.
 - b. Click **Delete**.

The panel displays only the remaining destinations.

Delete the J2C activation specifications related to the module that created these destinations.

Administering extended messaging resources: Overview

Extended messaging enables container-managed messaging. It extends the base Java Message Service (JMS) support, the Enterprise Java Bean (EJB) component model, and support for EJB 2.0 message-driven beans to allow use of the existing container-managed persistence and transactional behavior.

Extended messaging uses the bean-managed messaging implementation to provide the JMS interfaces, which ensures that both bean-managed and extended messaging use consistent JMS support. JMS usage is simplified since its support is managed by the extended messaging service.

The administrative console interface enables you to configure the resources needed by the extended messaging service and by the applications that use the service.

For a complete description of extended messaging, see the following articles in the WebSphere Business Integration Server Foundation information center:

- Extended messaging: Overview
- Using extended messaging in applications

Note: The Extended Messaging Service feature is being deprecated in WebSphere Process Server 6.0.1. Although you can continue to use extended messaging with new and existing applications in this release, you will need to replace these applications with ones that use standard JMS APIs, or replace them with equivalent messaging technologies.

Related tasks

“Enabling the extended messaging service”

The Extended Messaging Service provides runtime service to support container-managed messaging (extended messaging). The service can be automatically started when the application server is started, or it can be started manually.

“Configuring listener port extensions to handle late responses”

“Managing extended messaging providers” on page 146

“Adding a new input port” on page 146

“Adding a new output port” on page 149

Enabling the extended messaging service

The Extended Messaging Service provides runtime service to support container-managed messaging (extended messaging). The service can be automatically started when the application server is started, or it can be started manually.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. Ensure that the administrative console is running.
2. Click **Servers > Application servers > *server_name* > Extended Messaging Service** to display the Extended Messaging Service page.
3. If you want to enable the Extended Messaging Service to start automatically with server startup, select the **Enable service at server startup** check box. If you want to start the service manually, ensure the check box is cleared.
4. Click **OK**.
5. When prompted, click **Save** on the console task bar to save your changes to the master repository.
6. Stop and restart the application server in order for the changes to take effect.

You can also use the Extended Messaging Service page to configure a listener port extension to handle late responses, as described in “Configuring listener port extensions to handle late responses.”

Configuring listener port extensions to handle late responses

An application’s listener port can be configured with an extension to handle late responses in an extended messaging environment.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. Ensure you have a listener port defined and configured.

2. From the administrative console, click **Servers > Application servers > *server_name* > Extended Messaging Service > Listener Port Extensions**. The Listener Port Extensions page opens.
3. Click **New** to create a new listener port extension. The New Listener Port Extension page opens.
4. Select the **Enabled** check box to enable late response handling.
5. In the **Request Interval** field, either accept the default value or specify a new value. The request interval specifies how often the listener port checks for late responses.
6. In the **Request Timeout** field, either accept the default value or specify a new value. The request timeout value specifies how long the listener port waits for late responses. Responses received after that time are discarded.
7. Use the **Listener Ports** drop-down menu to specify the listener port you want to use for the extension.
8. Click **OK**.
9. When prompted, click **Save** on the console task bar to save your changes to the master repository.
10. Stop and restart the application server in order for the changes to take effect.

Managing extended messaging providers

An extended messaging provider manages the resources defined for use with container-managed messaging (extended messaging). Use the Extended Messaging Provider page in the administrative console to view the general properties for each resource and to select a resource for editing.

Extended messaging resources can be defined at a cell, node, or server scope:

- Cell scope—The most general scope. Extended messaging resources defined at the cell scope are visible from all nodes and servers, unless they have been overridden.
- Node scope—Extended messaging resources defined at the node scope override any duplicates defined at the cell scope. They are visible to all servers on the same node, unless they have been overridden at a server scope on that node.
- Server scope—Extended messaging resources defined at the server scope override any duplicate definitions defined at the cell or parent node scope. They are visible only to a specific server.

For detailed information about scopes, see the WebSphere Application Server for z/OS Information Center.

Security role required: You must be logged in as administrator, operator, configurator, or monitor to complete this task.

1. From the administrative console, click **Resources > Extended Messaging Provider**. The Extended Messaging Provider page opens.
2. Click the radio button next to the appropriate scope.
3. Click **Apply**. The **Scope**, **Name**, and **Description** fields on the bottom of the page are updated to reflect the values for the selected resource provider.

You can now create, modify or delete input ports, output ports, or other custom properties for the selected extended messaging provider.

Adding a new input port:

A receiver bean constructed from a session bean requires an input port to define the properties for the receiving Java Message Service (JMS) destination. The input port can also provide details for selecting and handling messages and for the reply destination (when required).

Note: Receiver beans that are constructed from message-driven beans do not require an input port; the details they contain are associated with the deployed message-driven bean and the Message Listener Service.

During this task, you configure the initial properties of the input port. You can later change the properties of the port as needed.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. From the administrative console, click **Resources > Extended Messaging Provider**. The Extended Messaging Provider page opens.
2. Select the scope for the resource provider to which you want to add the new port.
3. Click **Apply**.
4. Click **Input Ports** from the Additional Properties table. The Input Port page opens.
5. Click **New**. The Input Port configuration page opens.
6. Specify the appropriate properties for the new input port. See “Input port settings” on page 148 for detailed information about property settings.
7. Click **OK**.
8. When prompted, click **Save** on the console task bar to save your changes to the master repository.
9. Stop and restart the application server in order for the changes to take effect.

Modifying the configuration of an input port:

You can modify the configuration of an existing input port as needed. Input ports define the properties for the receiving Java Message Service (JMS) destination. They also provide details for selecting and handling messages and for the reply destination (when required).

Security role required: You must be logged in as administrator or configurator to complete this task.

1. From the administrative console, click **Resources > Extended Messaging Provider**. The Extended Messaging Provider page opens.
2. Select the scope for the resource provider to which you want to add the new port.
3. Click **Apply**.
4. Click **Input Ports** from the Additional Properties table. The Input Port page opens.
5. Select the input port that you want to modify. The Input Port configuration page opens. It displays the current configuration properties for that port.
6. Modify the properties for the input port. See “Input port settings” on page 148 for detailed information about property settings.
7. Click **OK**.
8. When prompted, click **Save** on the console task bar to save your changes to the master repository.

9. Stop and restart the application server in order for the changes to take effect.

Input port settings: An input port has the following configuration properties:

Scope The scope at which the extended messaging provider is defined. The value represents the location of the configuration file.

Name The name of the input port, used for administrative purposes. The value for this field must be a string.

JNDI Name

The Java Naming and Directory Interface (JNDI) name for the resource. The value for this field must be a string.

Description

A description of the input port, used for administrative purposes. The value for this field must be a string.

This field is optional.

Category

A category string to use when classifying or grouping the resource. The value for this field must be a string between 1 and 30 ASCII characters in length.

This field is optional.

JMS Connection Factory JNDI Name

The JNDI name for the Java Message Service (JMS) connection factory used by the input port (for example, jms/connFactory1). The value for this field must be a string.

JMS Destination JNDI Name

The JNDI name for the JMS destination used by the input port (for example, jms/destn1). The value for this field must be a string.

JMS Acknowledgement Mode

The JMS mode that is used to acknowledge messages. This property is used only for message-driven beans that use bean-managed transaction demarcation (in other words, the transaction type is set to Bean).

The valid values for this field are as follows:

- Auto Acknowledge—The session automatically acknowledges a message in either of the following cases:
 - When the session has successfully returned from a call to receive a message
 - When the session has called a message listener to process the message and received a successful response from that listener
- Dups OK Acknowledge—The session acknowledges only the delivery of messages. This can result in the delivery of duplicate messages if JMS fails.

The default mode is Auto Acknowledge.

Destination Type

The JMS resource type. The valid values for this field are as follows:

- Queue—The receiver bean receives messages from a queue destination
- Topic—The receiver bean receives messages from a topic destination

The default value is Queue.

Subscription Durability

Specifies whether a JMS topic subscription is durable. The valid values for this field are as follows:

- **Durable**—A subscriber registers a durable subscription with a unique identity that is retained by JMS. Subsequent subscriber objects with the same identity resume the subscription in the state in which it was left by the previous subscriber. If there is no active subscriber for a durable subscription, JMS retains the subscription's messages until they are received or they expire.
- **Nondurable**—Nondurable subscriptions last for the lifetime of their subscriber. A client sees the messages published on a topic only while its subscriber is active. If the subscriber is inactive, the client misses the messages published on its topic.

The default value is Durable.

This field is required only if the JMS destination type is a topic.

Reply JMS Connection Factory JNDI Name

The JNDI name of the JMS connection factory that is used for replies. The value for this field must be a string.

Reply JMS Destination JNDI Name

The JNDI name of the JMS destination that is used for replies. The value for this field must be a string.

Adding a new output port:

An output port specifies the properties needed by sender beans to define the destination for the message being sent. It also specifies optional properties if a response is expected. The output port is associated with the sender bean at deployment time.

During this task, you are configuring the initial properties of the output port. You can modify these properties later, as needed.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. From the administrative console, click **Resources > Extended Messaging Provider**. The Extended Messaging Provider page opens.
2. Select the scope for the resource provider to which you want to add the new port.
3. Click **Apply**.
4. Click **Output Ports** from the Additional Properties table. The Output Port page opens.
5. Click **New**. The Output Port configuration page opens.
6. Specify the appropriate properties for the new output port. See "Output port settings" on page 150 for detailed information about property settings.
7. Click **OK**.
8. When prompted, click **Save** on the console task bar to save your changes to the master repository.
9. Stop and restart the application server in order for the changes to take effect.

Modifying the configuration of an output port:

You can modify the configuration of an existing output port as needed. Output ports define the destination for the message being sent and provide the destination if a response is expected.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. From the administrative console, click **Resources > Extended Messaging Provider**. The Extended Messaging Provider page opens.
2. Select the scope for the resource provider to which you want to add the new port.
3. Click **Apply**.
4. Click **Output Ports** from the Additional Properties table. The Output Port page opens.
5. Select the output port that you want to modify. The Output Port configuration page opens. It displays the current configuration properties for that port.
6. Modify the properties for the output port. See “Output port settings” for detailed information about property settings.
7. Click **OK**.
8. When prompted, click **Save** on the console task bar to save your changes to the master repository.
9. Stop and restart the application server in order for the changes to take effect.

Output port settings: An input port has the following configuration properties:

Scope This field specifies the extended messaging provider scope. The value represents the location of the configuration file.

This field cannot be edited.

Name The name of the output port, used for administrative purposes. The value for this field must be a string.

JNDI Name

The Java Naming and Directory Interface (JNDI) name for the output port. The value for this field must be a string.

Description

A description of the output port, used for administrative purposes. The value for this field must be a string.

This field is optional.

Category

A category string to use when classifying or grouping the resource. The value for this field must be a string with a maximum of 30 ASCII characters.

This field is optional.

JMS Connection Factory JNDI Name

The JNDI name for the Java Message Service (JMS) connection factory used by the output port (for example, jms/connFactory1). The value for this field must be a string.

JMS Destination JNDI Name

The JNDI name for the JMS destination used by the output port. The value for this field must be a string (for example, jms/destn1).

JMS Delivery Mode

The JMS mode that is used to deliver messages. The value must be one of the following:

- Persistent—Messages put onto the destination are persistent.
- Nonpersistent—Messages put onto the destination are not persistent.

The default value is Persistent.

JMS Priority

The message priority for the queue destination. The value must be an integer from 0 to 9. The default value is 4.

JMS Time To Live

The time, in milliseconds, a message remains in the queue. After the specified time elapses, the message expires.

The value must be an integer from 0 to n

- 0—Messages never time out.
- n —Messages time out after n milliseconds

JMS Disable Message I.D.

Specifies whether the system generates a JMS message ID. The value must be one of the following:

- Selected—The system does not generate JMS message IDs
- Cleared—The system generates JMS message IDs automatically

By default, JMS message IDs are generated.

JMS Disable Message Time Stamp

Specifies whether the system generates a JMS message timestamp. The value must be one of the following:

- Selected—Message timestamps are not added to sent messages
- Cleared—Message timestamps are automatically added to sent messages

By default, message timestamps are added.

Response JMS Connection Factory JNDI Name

The JNDI name of the JMS connection factory that is used for responses handled by the output port (for example, `jms/connFactory1`). The value for this field must be a string.

Response JMS Destination JNDI Name

The JNDI name of the JMS destination that is used for responses handled by the output port (for example, `jms/destn1`). The value for this field must be a string.

Administering WebSphere ESB

Administering WebSphere ESB involves managing a service bus environment, deploying and managing mediation modules, managing resources used for service integration and mediation modules, and managing the clients and adapters that interact with the mediation modules. Administrators can also administer the full range of features of the underlying WebSphere Application Server.

With WebSphere ESB, administrators create an environment of ESB servers and service integration buses that support the deployment of mediation modules as service applications. When you install WebSphere ESB, you get two service integration buses to use for service applications deployed into the ESB. You can start with one server in a bus and optionally add capacity and enhanced

availability by build up to multiple servers or server clusters. You can add other buses if you need, to provide separate ESBs, to deploy applications that connect to an ESB, or to enable integration with WebSphere MQ.

Administrators can deploy mediation modules (as SCA modules) into the server and bus environment. They can view the modules that you have deployed, and monitor that requests are being processed correctly. Administrators can start or stop mediation modules, and can administer modules in other ways; for example, to change the configuration of a module, to stop or update the module, and otherwise manage its activity. Administrators can also make changes to the SCA imports of a mediation module; for example, to redirect one module to another module. This allows a module to invoke different service providers or process service requests and responses in different ways without having to rebuild and redeploy the module.

Administrators can use a variety of tools to administer WebSphere ESB, including the WebSphere administrative console, the WebSphere administrative (wsadmin) scripting program, command-line tools, and administrative programs.

The main descriptions of administrative tasks are based on use of the administrative console. Each task in the administrative console is supported by one or more panels. You can use a task filter to display the set of panels that are most appropriate to the tasks that you want to complete, and thereby focus your activities on only those panels.

All This displays all administrative console panels. This is most appropriate for an administrator interested in managing all parts of WebSphere ESB and the underlying WebSphere Application Server.

For more information about administration of enterprise service bus features, see the related tasks listed below.

Application Integration

This displays panels suitable for the following task areas:

- Adjusting the configuration of service integration buses, servers, server clusters, messaging engines and network topologies needed to support the deployment of mediation modules and service applications
- Creating resources (for example, JMS connection factories and Common Event Infrastructure profiles) needed by deployed service applications and mediation modules
- Operational control of mediation modules and service applications

This is most appropriate for an administrator interested in deploying and managing mediation modules as service applications, as described in “Managing service applications” on page 188.

Server and Bus

This displays panels suitable for the following task areas:

- Defining service integration buses, servers, server clusters, messaging engines and network topologies needed to support the deployment of mediation modules and service applications
- Enabling and disabling infrastructure services
- Installing applications and mediation modules
- Creating resources (for example, JMS connection factories and Common Event Infrastructure profiles) needed by deployed service applications and mediation modules

- Operational control of the bus and server environment

This is most appropriate for an administrator interested in managing the server and bus environment needed to support the deployment of service applications and mediation modules. This includes defining the network and bus topology, defining appropriate resources and monitoring the runtime system and troubleshooting any runtime errors. For more information about managing the bus and server environment, see “Managing the bus environment.”

Managing the bus environment

Service Integrators can deploy SCA modules without considering the server and bus environment that powers the enterprise service bus. However, an Administrator might want to manage the server and bus environment; for example, to start or stop a server, change the quality of service provided to SCA modules, add extra server capacity, or to adopt a more distributed bus environment.

If you choose a Complete (default) installation for WebSphere ESB, a default bus environment is created for you. This bus environment comprises a single server assigned to two service integration buses that are used for deploying SCA modules. You can also use the server to deploy J2EE components and resources, like servlets, enterprise Java beans, and JMS destinations.

The default bus environment might be adequate for your SCA modules. However, you might want to change the topology of the bus environment; for example, to add extra server capacity, or to adopt a more distributed bus environment for different departments or to separate test and production facilities.

Managing the bus environment includes some aspects of setting up WebSphere ESB, sometimes as part of a larger system, typically as a production environment or realistic test environment. This includes some installation and customization activities, bus topology planning, and creating product configurations. The focus is on the administration of service integration buses, servers, their resources, the set up and management of logical administrative domains of cells and nodes, and how to balance workload through clustering and high availability configurations.

The bus environment in WebSphere ESB uses the service integration technologies provided by WebSphere Application Server. This set of topics provides information about tasks involving the service integration technologies needed to manage the bus environment.

- Getting started with the server and bus environment
- Managing the SCA.SYSTEM bus topology
- Configuring a server or cluster for the SCA runtime
- Managing servers
- Balancing workloads with server clusters
- Doing more with bus topologies

Getting started with the bus environment

Service Integrators can deploy SCA modules without considering the server and bus environment that powers the enterprise service bus. However, an Administrator might want to manage the server and bus environment, and therefore need to understand what that management involves.

This set of topics provides some background to managing the bus environment, to help you get started for the first time. If you are familiar with managing bus environments, you might choose instead to go directly to the other sets of task descriptions that are sub-topics of “Managing the bus environment” on page 153.

Overview of the bus environment:

The bus environment comprises an one or more service integration buses, ESB servers, and their resources, organized into logical administrative domains of cells and nodes.

If you create a complete (default) installation for WebSphere ESB, you get a stand-alone server on which you can deploy Service Component Architecture (SCA) modules without having to do any configuration of the server.

However, administrators may still want to act on the bus environment, so would benefit from some detail about the environment.

- The SCA runtime (exploited by mediation modules) uses queues on an SCA.SYSTEM service integration bus as a robust infrastructure to support asynchronous interactions between components and modules. The queues are hosted by the server as a member of the SCA.SYSTEM bus.
- The ESB server provides the integration technologies, infrastructure services, configuration, and runtime administration needed to run mediation modules and service applications in WebSphere ESB. As a bus member, the server has a messaging engine that provides the core messaging functionality of the SCA.SYSTEM bus.

Both the server and SCA.SYSTEM are configured with default properties that may be suitable for you to deploy and run your SCA modules.

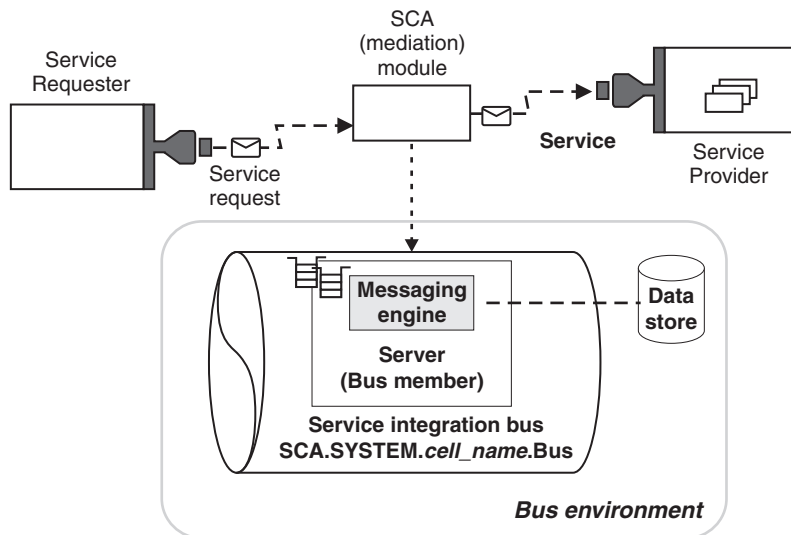


Figure 7. A bus environment with one server assigned to the SCA.SYSTEM service integration bus. As a bus member, the server is assigned one messaging engine, which has a data store for storing state data and messages. This figure also shows a mediation module deployed into the bus environment and assigned to the bus member.

For more advanced usage, you may want to change the configuration of the bus environment for a stand-alone server, or create a bus environment for a deployment manager cell. For example:

- You can configure a variety of quality of service from secure, assured delivery (where messages are guaranteed not to get lost and are transported securely) to best-effort (where messages may get lost in case of a system failure).
- You may want to set up a deployment manager cell to provide several servers to host mediation modules. This provides advantages of scalability, the ability to handle more client connections, and greater message throughput. You can also create server clusters, which enables you to manage a group of servers together and enables those servers to participate in workload management.
- Your complete bus environment may be made up of several stand-alone and deployment manager profiles, to provide separate administrative domains for different departments or to separate test and production facilities. Each profile has its own SCA.SYSTEMservice integration bus.

Besides the SCA.SYSTEM bus used for SCA modules, you can also create other service integration buses that you can use to support the service integration logic provided by the modules. For example, the SCA.APPLICATION.cell_name.Bus is provided and used to define JMS queue destinations and other JMS resources for modules deployed with JMS bindings.

You can create other buses for use as in WebSphere Application Server; for example, for applications acting as service requesters and providers within WebSphere ESB, or to link to WebSphere MQ. You can also use a WebSphere ESB deployment manager to manage separate application servers for use with applications and modules deployed onto WebSphere Application Server.

The application server:

WebSphere ESB incorporates WebSphere Application Server, which provides a high performance, secure and manageable middleware container that supports standard J2EE application components, such as enterprise beans, Web services, servlets, JCA connectors, and asynchronous beans.

WebSphere ESB provides functions for the deployment and administration of service applications, including a browser-based administrative console, command scripting, topology management (of service buses, servers, and clusters), a set of standard management programming interfaces called JMX, and an administrative model encompassing all hosted application components and infrastructure.

The server infrastructure provides network I/O, queuing, scheduling, threading, and dispatching functions common to most middleware. It also supports various transports and protocols to relieve developers of coding protocol semantics and message encodings.

WebSphere ESB can augment and enhance an application by providing qualities of service as runtime capabilities that are normally difficult to implement. For example, requirements on the hosting container such as atomic transactions or a specific security role for the caller are expressed declaratively in a service's metadata. Operational capabilities such as performance monitoring and measurement, workload routing, and workload management can be enabled administratively.

WebSphere ESB provides application runtime services that implement standard programming interfaces of the J2EE programming model; for example, application tracing and logging interfaces are integrated with those of the server runtime to provide a consistent view of problem diagnosis. Security interfaces extend

WebSphere ESB security to address application-specific business needs. Name space services enable the discovery and binding of services, and can be persisted throughout the administrative domain.

A service implementation deployed in WebSphere ESB gains all of these advantages without the need to explicitly code them, nor to be aware of any of the server infrastructure.

Service integration buses:

A service integration bus provides a scope within which you can configure resources for mediation modules and interaction endpoints deployed in WebSphere ESB.

Remember: An enterprise service bus is not only a service integration bus. An enterprise service bus is more an architecture used for the integration logic and connectivity for SCA-described interaction endpoints. A service integration bus is part of the technologies provided by the WebSphere Application Server on which WebSphere ESB is built. Your enterprise service bus might include multiple service integration buses, WebSphere MQ, and other technologies.

When abbreviating references about an enterprise service bus, we use the term *ESB*. When abbreviating references about a service integration bus, we use the term *bus*.

For WebSphere ESB, a bus enables message routing between endpoints with specific quality of interaction service and can temporarily persist messages if required. You can configure a variety of quality of service from secure, assured delivery (where messages are guaranteed not to get lost and are transported securely) to best-effort (where messages might get lost in case of a system failure). Endpoint implementers choose their desired quality of service by declaring the annotations on SCA module exports and imports. If a quality of service is unspecified, WebSphere ESB applies its defaults.

When you install WebSphere ESB, an SCA.SYSTEM bus is created for you to deploy SCA modules. If the configuration of this bus is not adequate for your SCA modules, or you want service integration buses for other uses, you can choose between a variety of bus environments including a multiple-server bus, several single-server buses that use different servers, and buses linked to WebSphere MQ.

Many scenarios require a simple bus topology; perhaps, for example, a single server. By adding multiple servers to a single bus, you can increase the number of connection points. By adding server clusters as members of a bus, you can increase scalability and achieve high availability. Servers, however, do not have to be bus members to connect to a bus. In more complex bus topologies, multiple buses are configured, and might be interconnected to form complex networks. An enterprise might deploy multiple buses for organizational reasons; for example, an enterprise with several autonomous departments might want to have separately-administered buses in each location.

Server clusters:

A *server cluster* is a set of servers that are managed together and participate in workload management. The servers that are members of a cluster can be on different host machines.

Servers that belong to a cluster are *members* of that cluster set and must all have identical application components deployed on them. Other than the applications configured to run on them, cluster members do not have to share any other configuration data. One cluster member might be running on a huge multi-processor enterprise server system, while another member of that same cluster might be running on a smaller system. The server configuration settings for each of these two cluster members are very different, except in the area of application components assigned to them. In that area of configuration, they are identical. This allows client work to be distributed across all the members of a cluster instead of all workload being handled by a single application server.

In WebSphere ESB, the support for server clusters is provided by the underlying WebSphere Application Server Network Deployment. The only difference is that you cannot create a cluster from an existing application server as a template.

For more information about server clusters, see the related topics.

Choosing a bus environment

If you choose a Complete (default) installation for WebSphere ESB, you get a single-server bus environment. If this is not adequate for your service applications, you can create a bus environment that includes a choice of bus topologies including a multiple-server bus in a deployment manager cell, several single-server buses that use different servers, and other buses for serving applications or linking to WebSphere MQ.

If you choose a Complete (default) installation for WebSphere ESB, you get a stand-alone node in its own administrative domain, known as a *cell*. The node hosts one server that is assigned to the SCA.SYSTEM bus for the cell, for you to deploy SCA modules.

If your SCA modules only need one server, you can use the SCA.SYSTEM bus of the Complete (default) installation. If your service applications need more than one server, you need to choose between a variety of bus topologies.

You should choose the bus environment needed for an enterprise service bus before deploying SCA modules, because it affects installation-related actions, including what WebSphere ESB profiles you need to create, and what databases you want messaging engines to use.

- You can install WebSphere ESB with the default stand-alone server profile, start with the SCA.SYSTEM bus provided then, if needed, later create a deployment manager profile and profiles for managed nodes, for a more advanced bus environment.
- To use more than one server in your bus environment, you need to use a profile for a managed node in a deployment manager cell.
- You might choose your bus environment before installing WebSphere ESB, then install the profiles that you need to best support your chosen bus environment.

Besides using the SCA.SYSTEM bus provided for SCA modules, you can create other application servers and service integration buses to support other applications and modules, or to connect to WebSphere MQ networks. This set of topics is mainly focused on using the SCA.SYSTEM bus to support SCA modules.

Information about using other service integration buses, as in WebSphere Application Server Network Deployment, is provided by links into the WebSphere Application Server topics.

To choose a bus environment, consider the following points and the descriptions of bus topologies given in the sub-topics referred to below.

- Consider the number of client connections and the throughput that you want for modules deployed to a bus.

The aim is to identify the point at which the performance of the modules, as perceived by the clients, starts to deteriorate:

- The number of concurrent client connections to the bus beyond which the performance starts to deteriorate when new client connections are made.
- The number of requests and replies flowing through a messaging engine beyond which the performance starts to deteriorate when new attempts are made to send requests through the messaging engine.

It is not possible to give a specific formula that applies to all environments, because it depends on the characteristics of the host on which the server runs, the nature of the modules deployed, and other factors.

If you use a single-server bus and observe that the number of client connections is causing a deterioration in performance, or that the throughput starts to deteriorate, you can increase the capacity of the bus environment in several ways:

- In a stand-alone profile, you might create several single-server buses using the same server. This enables the client connections to be distributed across several buses, but the throughput of requests still depends on the one server.
- For greater capacity of client connections and request throughput, you might use multiple servers distributed across several buses. (To use multiple servers distributed over one or more buses, you need to have a server profile for a managed node in a deployment manager cell.)

- Consider the size of requests flowing through a messaging engine.

Every messaging engine manages two memory buffers that contain requests and request-related data. If there is not enough space when the messaging engine attempts to add data to a buffer, the messaging engine might discard data already in the buffer to make space.

You might observe that a running messaging engine discards data from its buffer more often than is acceptable. In this case, you might add another server to the bus, to provide another messaging engine. Alternatively, you might choose to create several single-server buses that use different servers as their bus members. The messaging engine in each server uses a separate set of memory buffers, and a separate data store. (To use multiple servers distributed over one or more buses, you need to have a server profile for a node in a deployment manager cell.)

- Consider if you want to use different qualities of service for your service applications.

Each bus has a unique configuration of qualities of service and other properties. You might choose to create several buses and configure them with different qualities of service, then deploy each of your modules to the bus that has a suitable configuration.

- Consider other reasons for using multiple servers within a bus.

A service integration bus consisting of just one server is adequate for some applications. However, there are advantages in using more than one server in a bus (with each server providing a messaging engine):

- Spreading messaging workload across multiple servers.
- Placing request processing close to the requester applications, so as to reduce network traffic. For example, if the sending and receiving applications are

running in the same server process it would be inefficient to route all the requests that flow between them through a messaging engine running in a remote server.

- Improving availability in the face of system or link failure. This includes removing a single point of failure, and allowing store and forward between two servers should this be required.
 - Providing improved scalability
 - Accommodating firewalls or other network restrictions that limit the ability of network hosts to all connect to a single messaging engine.
- Consider other reasons for using a multiple SCA.SYSTEM bus environment. Because each service integration bus has a separate configuration, you might choose to have several buses each with a different configuration suitable for separate modules; for example, you might have some buses for a production environment with security, and other buses for testing without security. You might also choose to create several buses to separate the administration of modules; for example, separate administrative cells and their SCA.SYSTEM buses might be used for different departments within organizations, or perhaps to separate test and production facilities. Besides the SCA.SYSTEM buses, other buses might be created for other application use, and can be connected to allow messaging across the buses. Buses in different organizations can also be connected. When buses are interconnected, applications can send messages to applications on other buses, and use resources provided on other buses. Published messages can also span multiple buses, if the links between the buses are configured to allow it.
 - Consider reasons for using non-SCA service integration buses. Besides the SCA.SYSTEM bus used for SCA modules, you can also create other service integration buses that you can use to support the service integration logic provided by the modules. For example, the SCA.APPLICATION.cell_name.Bus is provided and used to define JMS queue destinations and other JMS resources for modules deployed with JMS bindings. You can create other buses for use as in WebSphere Application Server, for applications acting as service requesters and providers within WebSphere ESB, or to link a bus to WebSphere MQ. You can also use a WebSphere ESB deployment manager to manage separate application servers, for use with applications and modules deployed onto WebSphere Application Server.
 - Consider if you want application servers that do not support SCA modules. A WebSphere ESB deployment manager cell can include application server nodes that run WebSphere Application Server servers. You can use these application servers for applications and modules supported by WebSphere Application Server. You do not need to add the application servers into a service integration bus, unless you want to exploit the service integration technologies of WebSphere Application Server.

The SCA* buses provided:

When you create a WebSphere ESB stand-alone profile or a deployment manager profile, you get an enterprise service bus based on two service integration buses. These buses, with names starting SCA, are for you to deploy SCA modules.

SCA.SYSTEM.cell_name.Bus

This is the bus used to host queue destinations for SCA modules, such as mediation modules that provide service integration logic in the enterprise service bus. The SCA runtime (exploited by mediation modules) uses

queue destinations on the SCA.SYSTEM bus as a robust infrastructure to support asynchronous interactions between components and modules.

SCA.APPLICATION.cell_name.Bus

This bus is used to create resources for modules deployed with JMS bindings. This bus is an example of how you might create a service integration bus for use other than to deploy SCA modules.

The SCA.SYSTEM bus provides a scope within which resources, such as queue destinations, are configured for mediation modules and interaction endpoints. The bus enables message routing between endpoints with specific quality of interaction service and can temporarily persist messages if required. You can configure a variety of quality of service from secure, assured delivery (where messages are guaranteed not to get lost and are transported securely) to best-effort (where messages might get lost in case of a system failure). Endpoint implementers choose their desired quality of service by declaring the annotations on SCA module exports and imports. If a quality of service is not specified, WebSphere ESB applies its defaults.

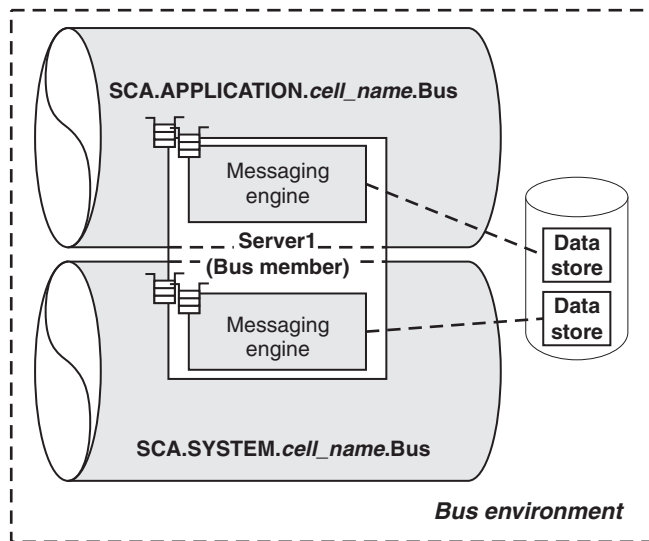


Figure 8. The SCA.*buses for a stand-alone server profile. The service integration buses created when you install WebSphere ESB.

In a stand-alone profile, there is one bus member that provides the one messaging engine in a bus, a topology that is adequate for some applications.

A single messaging engine might not be adequate if the number of client connections becomes excessive, if the rate of message throughput cannot be sustained by the one messaging engine, or if the size of messages has a detrimental affect on the message buffers used by the messaging engine.

To add more than one messaging engine to a service integration bus, you need to use a profile for a managed node in a deployment manager cell.

Creating a single-server enterprise service bus:

The simplest topology is an enterprise service bus consisting of a single server, as created if you install WebSphere ESB in a stand-alone server profile.

In a single-server enterprise service bus, the SCA.SYSTEM bus has the one server as its only bus member. When you install a mediation module into WebSphere ESB, the queue destinations used by the module are defined on that bus member. These queue destinations are used by the SCA runtime exploited by the mediation module as a robust infrastructure to support asynchronous interactions between components and modules.

The server hosts a messaging engine that provides the service integration technologies used by the queue destinations and the processing of requests.

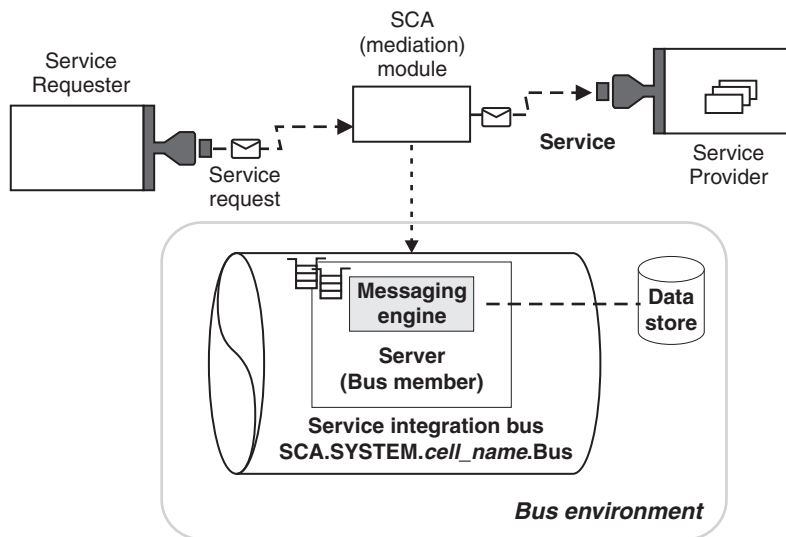


Figure 9. A single-server bus. A bus environment showing one service integration bus, with one ESB server as a bus member. This figure also shows an SCA module deployed into the bus environment, where the destinations needed are assigned to the bus member.

In a stand-alone profile, there can be only one messaging engine in a bus, a topology that is adequate for some scenarios.

A single messaging engine might not be adequate if the number of client connections becomes excessive, if the rate of message throughput cannot be sustained by the one messaging engine, or if the size of messages needs more space than the one messaging engine can provide in its data store.

To add more than one messaging engine to a service integration bus, you need to use a profile for a managed node in a deployment manager cell.

The easiest way to create a stand-alone server is to perform a *Complete* installation. With a Complete installation, you get a stand-alone server profile named default with a server named server1.

1. Perform a Complete installation of WebSphere ESB. This installs the core product files and creates the first stand-alone server profile.
2. Start server1 using the First steps console or the startServer server1 command.

You can now run the WebSphere ESB samples and deploy service applications into your enterprise service bus.

Creating a multiple-server enterprise service bus without clustering:

A enterprise service bus that consists of multiple servers provides advantages of scalability, the ability to handle more client connections and greater message throughput. You can also deploy SCA modules to different servers; for example, to provide different resources and qualities of service, or to provide some separation for different departments within organizations, or perhaps to separate test and production facilities.

To create more than one server in your bus environment, you need to have a managed node in a deployment manager cell.

You configure each server for the SCA runtime required by mediation modules. This advanced configuration defines whether queue destinations exploited by the SCA runtime are hosted locally on the server or on a remote server. If you specify that the server is to host queue destinations, the server is made a member of the SCA.SYSTEM bus and gets a messaging engine to which the queue destinations are assigned. If you specify that the server does not host queue destinations, it does not need to be a member of the SCA.SYSTEM bus, so does not need a messaging engine.

Consider the scenario shown in the figure Figure 10.

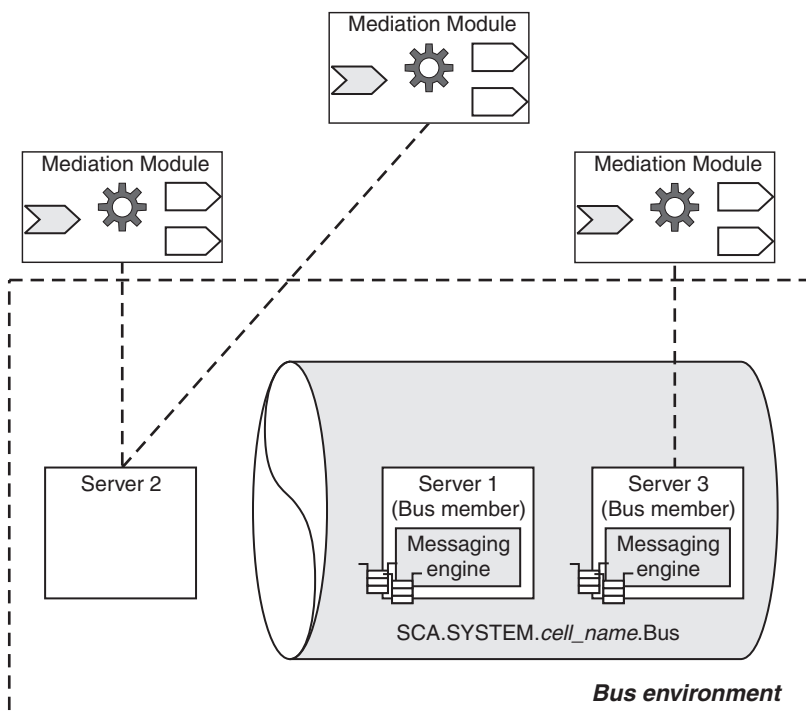


Figure 10. A multiple-server bus without clustering. Server 1 is configured to host queue destinations for mediation modules deployed to any of the servers, but not to host mediation modules or other service applications. Server 2 is configured to host mediation modules or other service applications, but to use queue destinations hosted on another server (server 2 or server 3 in this figure.) Server 3 is configured to both host mediation modules or other service applications and to host queue destinations for mediation modules deployed to any of the servers.

All of the messaging engines in the SCA.SYSTEM bus are implicitly connected, and requests can be processed by any messaging engine in the bus. Knowledge of the resources assigned to each messaging engine in a bus is shared between all the messaging engines in the bus.

There is no requirement for all the messaging engines in the bus to be running at the same time; if one of the messaging engines is stopped, the rest of the messaging engines continue to operate. However resources owned by a messaging engine, specifically queue points for mediation modules, are unavailable if the engine is stopped. Also, a messaging engine can only run in the server it was created for. The server is therefore a single point of failure; if the server cannot run, the messaging engine is unavailable. By configuring a server cluster as a bus member, a messaging engine has the ability to run in one server in the cluster, and if that server fails, the messaging engine can run in an alternative server within the cluster.

There are several different ways to create a multiple-server enterprise service bus:

- Install a cell of managed server nodes on one machine.
- Install a cell of managed server nodes on several machines.

You can now run the WebSphere ESB samples and deploy service applications into your enterprise service bus.

Multiple-server enterprise service bus with clustering:

A deployment manager cell can be used for an enterprise service bus that consists of multiple servers, some or all of which are members of server clusters.

A server that hosts queue destinations for SCA modules has one messaging engine in the SCA.SYSTEM bus. For many purposes this is sufficient, but such a messaging engine can only run in the server it was created for. The server is therefore a single point of failure; if the server cannot run, the messaging engine is unavailable. By configuring a server cluster as a bus member instead, the messaging engine has the ability to run in one server in the cluster, and if that server fails, the messaging engine can run in an alternative server. This is illustrated in Figure 11 on page 164.

Another advantage of configuring a cluster bus member is the ability to share the workload associated with an SCA module across multiple servers. For an SCA module deployed to a cluster bus member, the queue destinations used are partitioned across the set of messaging engines run by the cluster servers. The messaging engines in the cluster each handle a share of the messages passing through the SCA module.

To summarize, with a cluster bus member you can achieve either failover, workload sharing, or both, depending on policies that you can configure.

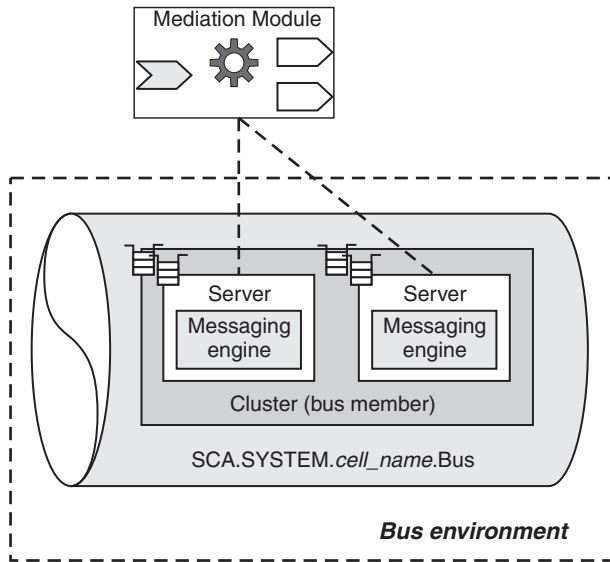


Figure 11. A multiple-server bus with clustered servers for failover

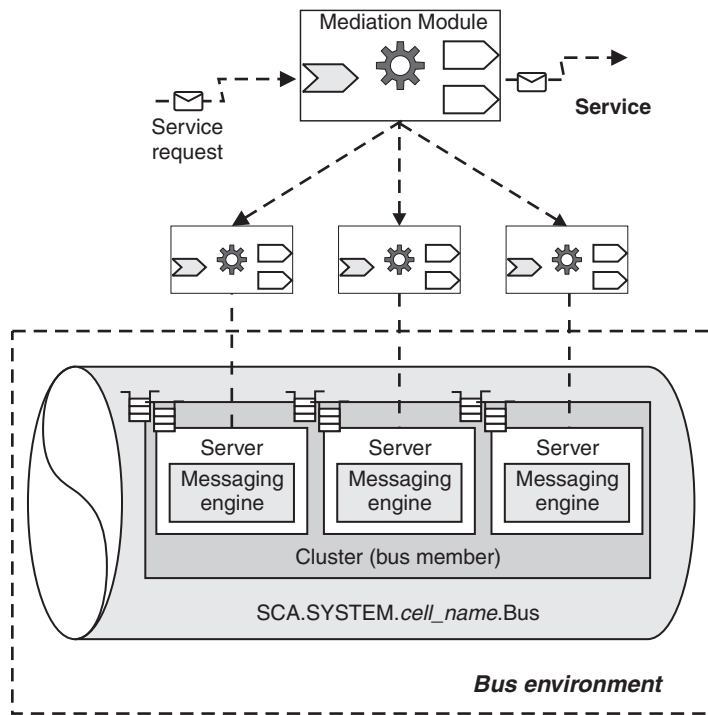


Figure 12. A multiple-server bus with clustered servers for workload sharing

There are several different ways to create a multiple-server enterprise service bus:

- Install a cell of managed server nodes on one machine.
- Install a cell of managed server nodes on several machines.

You can now run the WebSphere ESB samples and deploy service applications into your enterprise service bus.

Creating a multiple enterprise service bus topology:

You might want to deploy and manage SCA modules in a distributed bus environment; for example, with separate enterprise service buses for different departments or to separate test and production facilities.

Each stand-alone profile or deployment manager profile that you create has its own administrative cell that can be seen as the administrative domain for a separate enterprise service bus.

Your complete bus environment might be made up of several stand-alone and deployment manager cells, each representing a separate enterprise service bus, with its own SCA.SYSTEM bus used for SCA modules.

Besides the SCA.SYSTEM bus used for SCA modules, you can also create other service integration buses that you can use to support the service integration logic provided by the modules. For example, the SCA.APPLICATION.*cell_name*.Bus is provided and used to define JMS queue destinations and other JMS resources for modules deployed with JMS bindings.

You can create other buses for use as in WebSphere Application Server; for example, for applications acting as service requesters and providers within WebSphere ESB, or to link a bus to WebSphere MQ.

You can also use a WebSphere ESB deployment manager to manage separate application servers, for use with applications and modules deployed onto WebSphere Application Server.

Whilst you can use these other buses separately, you can also connect them to allow messages to pass between the buses. You can also connect together buses in different organizations. When buses are interconnected, applications can send messages to applications on other buses, and use resources provided on other buses. Published messages can span multiple buses where the links between the buses are configured to allow it.

A service integration bus must be contained within a single cell; that is, a bus cannot span multiple cells. However, a cell can contain more than one bus. In this case, each bus in the cell is “foreign” to each other bus in the cell. You can connect buses together within a cell, or between different cells.

The process for linking one bus to another bus is the same, whether the buses are in the same cell or in different cells.

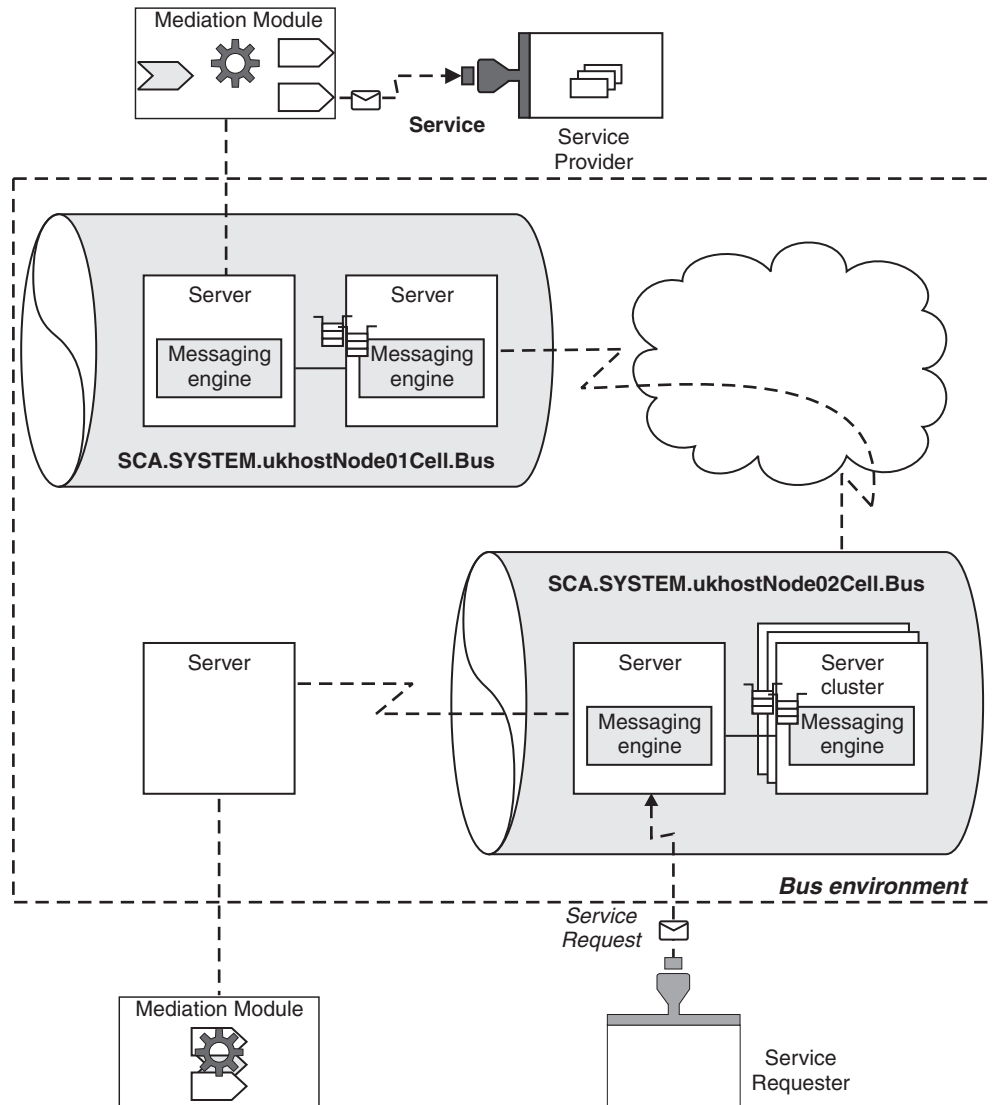


Figure 13. A multiple-enterprise service bus topology

There are several different ways to create a multiple enterprise service bus topology; for example:

1. Install a single-server enterprise service bus on one machine. . This effectively creates one enterprise service bus, with its own SCA.SYSTEM bus.
2. Install a cell of managed server nodes on several machines.

If you want one or more of the nodes on the same machine, you can either use the same installation of WebSphere ESB or use separate installed copies of WebSphere ESB on the same machine.

You can now run the WebSphere ESB samples and deploy service applications into your enterprise service bus.

An enterprise service bus with links to WebSphere MQ networks:

Service integration buses can contain links to WebSphere MQ networks. This allows applications connected to a WebSphere MQ queue manager to send

messages to an application that is attached to a service integration bus, and for such an application to send messages to WebSphere MQ.

When a service integration bus is connected to a WebSphere MQ network, the network is extended by adding support for:

- Service applications deployed into the service integration bus
- JMS applications attached to the service integration bus
- Web services requesters or providers

The WebSphere MQ network is represented by a *foreign bus* configured on a messaging engine. A construct called an *WebSphere MQ link* connects the messaging engine to an MQ queue manager using sender and receiver channels, thereby providing a bridge between the bus and the so-called *gateway queue manager* of a WebSphere MQ network.

The WebSphere MQ link provides connectivity not just with the messaging engine that hosts the link, but also with the other messaging engines in the bus. All the messaging engines in the bus appear to the WebSphere MQ network as if they were a single queue manager (they inherit the queue manager name from the WebSphere MQ link)

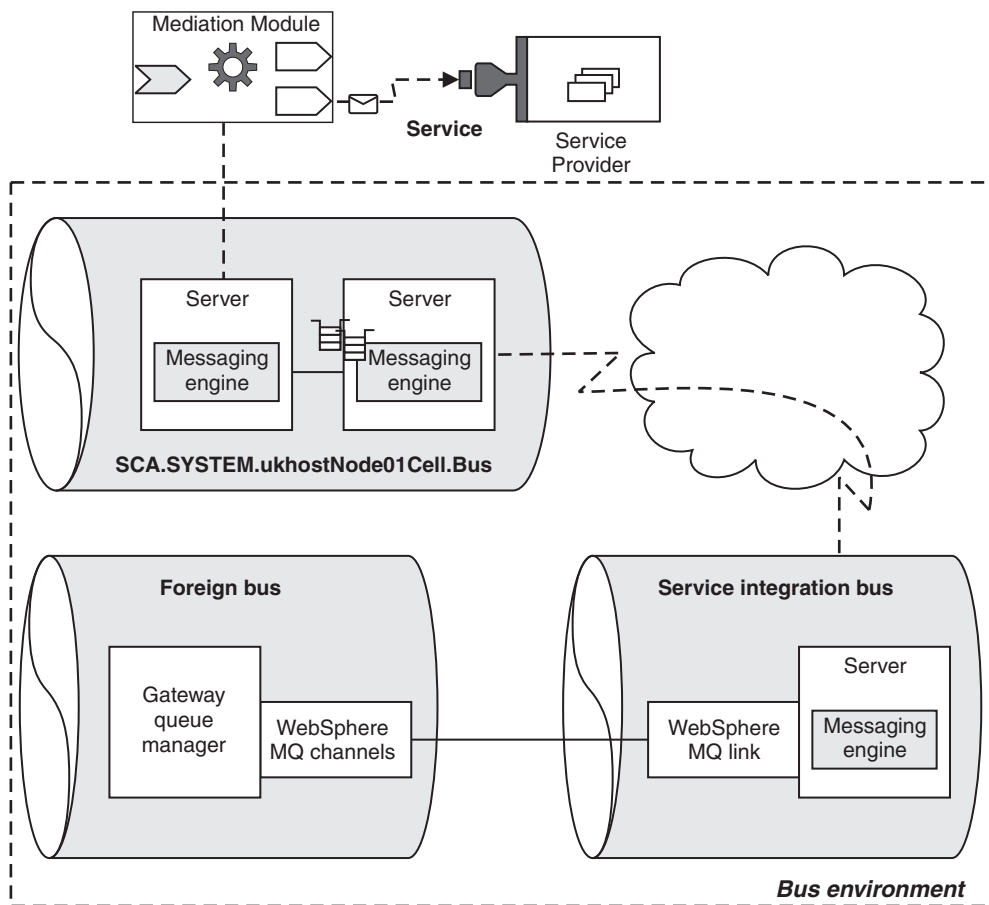


Figure 14. Service integration buses with links to a WebSphere MQ network

WebSphere MQ links can be used in a number of different configurations. A messaging engine can contain multiple WebSphere MQ links to different gateway queue managers.

Links to WebSphere MQ networks are implemented by the service integration technologies of WebSphere Application Server. .

Server and bus features of the administrative interfaces

WebSphere ESB allows you to view and change aspects of a server and bus environment from the administrative console, and from the command line.

Servers → Application servers

This set of pages provides access to ESB servers, which are application servers created from a special template for an enterprise service bus.

The ESB server provides the integration technologies, infrastructure services, configuration, and runtime administration needed to run mediation modules and service applications in WebSphere ESB. As a bus member, the server has a messaging engine that provides the core messaging functionality of a service integration bus. A messaging engine manages bus resources and provides a connection point for service requesters.

The detail settings page for a server provides access to the service integration properties under the **Server messaging** category label, including messaging engines, messaging engine inbound transports, and WebSphere MQ link inbound transports.

Service integration → Buses

This set of pages provides access to the service integration technologies of WebSphere Application Server, which are used in the deployment of mediation modules (and other SCA modules).

Each service integration bus provides a scope within which you can configure resources for mediation modules and interaction endpoints deployed in WebSphere ESB. The bus enables message routing between endpoints with specific quality of interaction service and can temporarily persist messages if required.

When you install WebSphere ESB, two service integration buses are created for you. These service integration buses, with names starting SCA, are for you to deploy SCA modules. When you install a mediation module into WebSphere ESB, the module is deployed onto a member of the service integration bus called SCA.SYSTEM.*cell_name*.Bus, and a set of bus destinations are created on a member of that bus.

The detail settings page for a bus provides access to service integration resources under a number of category labels, including bus destinations through [**Destination resources**] **Destinations**.

Managing the SCA.SYSTEM bus topology

The service integration bus, SCA.SYSTEM.*cell_name*.Bus, is used to deploy SCA modules. You can display the local topology of the bus, change the properties of the bus, add servers and clusters as bus members, and perform other administrative tasks to manage the overall topology.

If you create a Complete (default) installation for WebSphere ESB, you get a stand-alone node in its own administrative cell. The node hosts one server that is assigned to the SCA.SYSTEM service integration bus for the cell.

This bus topology is adequate to deploy SCA modules for some application integration scenarios, but you may want to change the bus topology to take advantage of using more than one member in a bus, or of using more advanced environments involving multiple service integration buses and perhaps links to WebSphere MQ.

If you create a deployment manager profile, you can add managed nodes to the deployment manager cell, and create one or more servers or server clusters on those nodes, as members of the SCA.SYSTEM bus, to deploy SCA modules.

When created, the SCA.SYSTEM bus and servers are assigned values for only some of the important properties, with other properties taking default values.

Use the information in the following sub-topics to display the topology and properties of the SCA.SYSTEM bus and servers, to configure them to your needs, and to manage their runtime state. These sub-topics are aimed at routine tasks to support the deployment of mediation modules into the SCA.SYSTEM bus. These tasks, and other tasks, for the general management of the server and bus environment are provided in the list of related tasks and under “Doing more with bus topologies” on page 188.

When managing the SCA.SYSTEM bus and servers, you consider a variety of issues, including the following:

- How are SCA modules to be distributed across servers?
- If you want servers to be managed together or participate in workload management, what server clusters do you want?
- What are the naming conventions for buses, servers, and other resources?
- What message persistence and other qualities of services do you want to provide?
- What databases do you want to use for messaging engines to persist messages?

Remember: Your bus environment might be a combination of several bus topologies. For example, if you create several WebSphere ESB stand-alone profiles or deployment manager profiles, each profile has a separate cell (with its own SCA.SYSTEM bus). You might have some buses with a single server, and other buses with several servers and server clusters.

Creating the initial SCA.SYSTEM bus topology:

When you create a stand-alone server profile or deployment manager profile, you get a separate administrative cell. The service integration bus SCA.SYSTEM.*cell_name*.Bus is created automatically for you to deploy SCA modules into the cell.

If you create a Complete (default) installation for WebSphere ESB, you get a stand-alone server in its own administrative cell. The server that is assigned to the SCA.SYSTEM service integration bus for the cell.

If you create a deployment manager profile, you need to create managed nodes in the cell then create servers or server clusters as members of the SCA.SYSTEM bus.

Before continuing with this task, consider whether you are likely to need more than one server or server cluster in the bus environment. In a stand-alone server profile, there can be only the one server. If you are likely to want several servers,

or server clusters, in the same bus environment, you should consider creating a deployment manager cell with managed nodes. You can also create several stand-alone servers on one or machines.

Each stand-alone node or deployment manager cell forms a separate administrative cell that has its own service integration bus called `SCA.SYSTEM.cell_name`.Bus that is used to deploy SCA modules within the cell.

Use this task to create the `SCA.SYSTEM` bus for an administrative cell, with one ESB server as a bus member. The `SCA.SYSTEM` bus and server are created with values for only some of the important properties, with other properties taking default values.

You can change the configuration of the buses; for example, to change bus properties or, for a deployment manager cell, add servers or server clusters as bus members.

To create the `SCA.SYSTEM` bus with one server as a bus member, complete one of the following steps:

- Create a stand-alone server.
If you choose a Complete (default) installation for WebSphere ESB, you get a stand-alone server node in its own administrative cell. The node hosts one server that is assigned to the `SCA.SYSTEM` bus for you to deploy SCA modules. The server is created and added automatically to the `SCA.SYSTEM` bus for your cell. This topology is limited to the one server.
- Create a deployment manager cell with a server on a managed node. This has advantages of centralized administration and potential workload balancing across server clusters. You can start with a cell containing one managed node for a custom profile, then add more servers to that node, or create more profiles, if needed later.

When you log in to the administrative console for the cell, you can see the `SCA.SYSTEM` bus in the list on the **Service integration** → **Buses** page. You can also see the new server in the list on the **Servers** → **Application Servers** page.

After starting the new server, you can use the `SCA.SYSTEM` bus to deploy SCA modules.

You can change the configuration of the buses; for example, to add servers or server clusters to a deployment manager cell.

Displaying the topology of a service integration bus:

The bus topology view of the administrative console displays the bus members and messaging engines of a selected bus. For each messaging engine, the runtime status is displayed.

Use this task to display a tree view of the bus members and messaging engines in a service integration bus. You can use this *local topology* view to add servers and server clusters as members of the bus, and to add messaging engines to cluster bus members.

To display the local topology view of a service integration bus, use the administrative console to complete the following steps:

1. In the navigation pane, click **Service integration** → **Buses**. A list of buses is displayed in the content pane.
2. In the content pane, click the name of the bus. For example, `SCA.SYSTEM.localhostCell01.Bus`
3. Click the tab **Local Topology**

The topology of the bus is displayed as an expandable tree.

You can expand nodes of the tree to display the bus members and their messaging engines.

You can add servers and server clusters as members of the bus, and add messaging engines to cluster bus members.

Changing bus properties:

You can create a service integration bus with its properties left to take their default values. You can later change the values of properties to adapt the use of a bus to meet changes in your needs.

If you think that you might need to change the use of a service integration bus, you can use this task to help you choose which properties to change. This task provides an overview of the considerations for changing bus properties, and links to a set of task descriptions provided for the service integration technologies of the underlying WebSphere Application Server.

- **Configuring messaging security for the SCA.SYSTEM bus.**

When messaging security is switched on, all access to the bus itself and to all the destinations on the bus must be authorized. This means that all users who want to connect to the bus must have permission to use the bus resources, either directly or as part of a user group.

When the SCA.SYSTEM bus is created, messaging security for a bus is switched on by default. Only authorized messaging engines are allowed to create a connection to a secure bus.

The SCA.SYSTEM bus is configured with the authentication alias called `SCA_Auth_Alias`, with an initial set of default authorization permissions that allows SCA to connect to the secured bus, and grants full access to all local destinations on the bus. You can change the default authorization permissions to restrict access to a bus to a specific set of users, and can choose to create and use a different authentication alias.

By default, the same `SCA_Auth_Alias` is used as the inter-engine authentication alias, to authorize communication between messaging engines on the bus. You can choose to create and use a different inter-engine authentication alias.

To further configure security, you can use secure transport connections (SSL or HTTPS) to ensure confidentiality and integrity of messages in transit between application clients and messaging engines and between messaging engines. You can specify the transport chain `InboundSecureMessaging` (for JFAP over SSL over TCP/IP) that is provided with WebSphere ESB. Alternatively, you can also create and specify another transport chain.

- **Configure whether messages on a deleted message point should be retained or discarded.**

If you uninstall the application for an SCA module, the queue points that the SCA runtime uses for that module are deleted. By default, the `Discard messages` property is cleared, so any messages on any of those queue points are retained

at a system exception destination so that you can process them later. You can select the Discard messages property so that any messages on deleted queue points are discarded.

- Configure whether any updates to the configuration information are dynamically passed to messaging engines.

By default, the Configuration reload enabled property of the SCA.SYSTEM bus is selected. This indicates that any updates to the configuration information are dynamically passed to the server, and therefore made available to messaging engines whether or not they are started. (When a messaging engine is started, it uses the information in the server that it is running in.)

- Configure the maximum total number of messages that a messaging engine can place on its message points.

By default, the High message threshold property of the SCA.SYSTEM bus is set to 50000 messages. This sets the initial value of that property on each messaging engine in the bus. If the current message depth on a queue point is equal to the high message threshold, the messaging engine does not accept new messages until the queued messages have been consumed. You can change the High message threshold property (on the bus or on a messaging engine) as one way to overcome problems with the throughput of messages.

Configuring servers or clusters for the SCA runtime

Before you can deploy a mediation module onto a server or server cluster, you must have configured that server or cluster for the SCA runtime required. This advanced configuration defines whether queue destinations exploited by the SCA runtime are hosted locally or on a remote server or cluster.

Before starting this task, you must have already created one or more servers or server clusters required:

- The server or cluster on which you want to deploy mediation modules.
- The server or cluster on which you want to create queue destinations for the SCA runtime to exploit.

You can use the same server or cluster in both cases.

For more information about creating servers or server clusters, see the following task descriptions:

- Creating a server
- Creating a server cluster

When you install a mediation module into WebSphere ESB, a number of queue destinations are created for components of the mediation module that use asynchronous interactions. The queue destinations are used by the SCA runtime to hold messages being processed for the mediation module.

To specify where the queue destinations are created, you complete one of the following configuration alternatives for the server or cluster.

- Set the advanced configuration of the server or cluster to host destinations. The configuration defines that this server or cluster can host queue destinations for mediation modules deployed locally or for mediation modules deployed on other servers or clusters. This configuration also adds the server or cluster as a member of the SCA.SYSTEM bus, and configures the messaging engine for the server or cluster.

To perform this configuration task, see “Configuring a server or cluster to host queue destinations for mediation modules” on page 173.

- Set the advanced configuration of the server or cluster to use remote destinations. The configuration defines that this server or cluster uses a *remote destination location* for destinations.

This server or cluster does not need to be a member of the SCA.SYSTEM bus.

If you deploy a mediation module to this server or cluster, the queue destinations are configured on a different server or cluster that is a member of the SCA.SYSTEM bus.

To perform this configuration task, see “Configuring a server or cluster to use remote destinations for mediation modules” on page 175.

The server or cluster is configured to accept the deployment of mediation modules.

You can now deploy mediations onto the server or cluster or take other actions to manage the server or cluster.

Configuring a server or cluster to host queue destinations for mediation modules:

For mediation modules, a cluster or server can be configured to host the queue destinations used by the SCA runtime. The server or cluster can host queue destinations for mediation modules that are deployed to any server or cluster in the administrative cell.

Before starting this task, you must have already created the server or cluster.

When you install a mediation module into WebSphere ESB, a number of queue destinations are created for components of the mediation module that use asynchronous interactions. The queue destinations are used by the SCA runtime to hold messages being processed for the mediation module.

You can configure your servers and clusters so that the queue destinations are created on the server or server cluster that the mediation module is deployed to, or on a different server or cluster.

Note: You must complete this task before the server or cluster can be used to deploy mediation modules and host their queue destinations. Alternatively, you can configure the server or cluster to use a *remote destination location* for destinations, as described in “Configuring a server or cluster to use remote destinations for mediation modules” on page 175.

A server or cluster that hosts the queue destinations of mediation modules can be optimized for queue serving, and can provide those queues to a number of other servers and clusters on which mediation modules are deployed.

This configuration task defines that the server or cluster can host queue destinations for mediation modules deployed locally or for mediation modules deployed on other servers or clusters. This configuration also adds the server or cluster as a member of the SCA.SYSTEM bus, and configures the messaging engine for the server or cluster.

- If you configure a server, WebSphere can create a messaging engine with default properties. By default, the messaging engine is configured to use the default JDBC data source and Cloudscape JDBC Provider for its data store. This enables the messaging engine to run without needing any additional configuration.

If you do not want to use the default data source configuration, you can choose to use a different data source or you can configure the data store to use a different JDBC provider.

- If you configure a cluster, you must explicitly create both the data store and the JDBC data source that the messaging engine uses to interact with the data store. You must also set the connection pool Purge policy to EntirePool.

To configure a server or cluster to host the queue destinations for mediation modules, use the administrative console to complete the following steps:

1. Display the Advanced Configuration page for the server or cluster.
 - To configure a server: **Servers** → **Application Servers** → *server_name* → **Advanced Configuration**
 - To configure a cluster: **Servers** → **Clusters** → *cluster_name* → **Advanced Configuration**
2. Verify the value in the Emitter Factory Profile JNDI Name field.
 - If you want to change the default configuration for emitting events to the CEI server, select the appropriate emitter factory profile JNDI name from the field. In a clustered environment, the value you select is used for all servers in the cluster.
 - If you do not want to specify an emitter factory profile (thus preserving an existing CEI configuration), select **None** from the menu.
3. Click **Default Destination Location**. Ensure that the Do not configure to host SCA applications option is cleared.
4. Configure the messaging engine options for the server or cluster:

Resource being configured	Steps to perform
Configuring a server	<ul style="list-style-type: none"> • If you want the messaging engine to use the default JDBC data source and Cloudscape JDBC Provider for its data store, select the Use default data source values check box, then goto step 5 on page 175. • If you want the messaging engine to use non-default options, use the same steps listed for a cluster, from step 1 on page 175.

Resource being configured	Steps to perform
Configuring a cluster	<ol style="list-style-type: none"> 1. Use the JDBC Provider menu to select the appropriate JDBC provider template. 2. In the Application bus schema name field, type the name of the database schema used to contain the tables for the SCA.APPLICATION bus data source. Each messaging engine stores its resources, such as tables, in a single schema. Each database schema is used by one messaging engine only. Although every messaging engine uses the same table names, its relationship with the schema gives each messaging engine exclusive use of its own tables. The schema used for the default data source is IBMWSSIB. 3. In the System bus schema name field, type the name of the database schema used to contain the tables for the SCA.SYSTEM bus data source. 4. In the Data source user name field, type the user name you are using to access the database. 5. In the Data source password field, type the password associated with the user name. 6. In the Application Bus Database Name field, type the name of the database used by the messaging engine created on the SCA.APPLICATION bus. 7. In the System Bus Database Name field, type the name of the database used by the messaging engine created on the SCA.SYSTEM bus. 8. If you want the messaging engine to create the database tables for the data source, select the Create tables check box. (The tables are created when the server hosting the messaging engine is started.) Otherwise, the database administrator must create the database tables.

5. Click **OK**.

6. Save your changes to the master configuration.

You can deploy mediation modules to another server or cluster that has been configured to use a *remote destination location*. If that other server or cluster has its Remote Destination Location property set to this server or cluster, the queue destinations for the mediation module are configured on this server or cluster.

You can tune the JDBC data source of a messaging engine for performance.

Configuring a server or cluster to use remote destinations for mediation modules:

For mediation modules, a cluster or server can be configured so that the queue destinations are created on a different server or cluster.

Before starting this task, you must have already created one or more servers or server clusters required:

- The server or cluster on which you want to deploy mediation modules. This is the server or cluster to be configured in this task.
- The server or cluster on which you want to create queue destinations for the SCA runtime to exploit.

Note: This server or cluster must have been configured to host queue destinations for mediation modules, as described in “Configuring a server or cluster to host queue destinations for mediation modules” on page 173.

You can use the same server or cluster in both cases.

When you install a mediation module into WebSphere ESB, a number of queue destinations are created for components of the mediation module that use asynchronous interactions. The queue destinations are used by the SCA runtime to hold messages being processed for the mediation module.

You can configure your servers and clusters so that the queue destinations are created on the server or server cluster that the mediation module is deployed to, or on a different server or cluster.

Note: You must complete this task before the server or cluster can be used to deploy mediation modules that want their queue destinations hosted elsewhere. Alternatively, you can configure the server or cluster to host the queue destinations, as described in “Configuring a server or cluster to host queue destinations for mediation modules” on page 173.

A server or cluster that does not host the queue destinations for mediation modules can be optimized to run other components, and does not have to be a member of the SCA.SYSTEM bus.

This configuration task defines that the server or cluster uses queue destinations created on a different server or cluster. This configuration does not add the server or cluster as a member of the SCA.SYSTEM bus

To configure a server or cluster to use queue destinations created on a different server or cluster, use the administrative console to complete the following steps:

1. Display the Advanced Configuration page for the server or cluster.
 - To configure a server: **Servers** → **Application Servers** → *server_name* → **Advanced Configuration**
 - To configure a cluster: **Servers** → **Clusters** → *cluster_name* → **Advanced Configuration**
2. Verify the value in the Emitter Factory Profile JNDI Name field.
 - If you want to change the default configuration for emitting events to the CEI server, select the appropriate emitter factory profile JNDI name from the field. In a clustered environment, the value you select is used for all servers in the cluster.
 - If you do not want to specify an emitter factory profile (thus preserving an existing CEI configuration), select **None** from the menu.
3. Clear the Do not configure to host Process Server applications option.

4. Configure the Remote Destination Location option for the server or cluster:
 - a. Click **Remote Destination Location**.
 - b. Use the associated menu to select the name of the remote server or cluster that hosts queue destinations for mediation modules.
5. Click **OK**.
6. Save your changes to the master configuration.

If you deploy a mediation module to this server or cluster, the queue destinations for the mediation module are configured on the server or cluster specified by the Remote Destination Location property.

You can change the configuration of the server or cluster, add it as a member of a service integration bus, or take other actions to manage the server or cluster.

Managing ESB servers

Server configuration defines settings that control how a server provides services for running applications and their components. Administrators can create and configure servers in an existing server and bus environment.

- To create new servers, you need to have a managed node in a deployment manager cell.

If you choose a Complete (default) installation for WebSphere ESB, a default bus environment is created for you. This bus environment comprises a single server assigned to two service integration buses that are used for deploying SCA modules. You can also use the server to deploy J2EE components and resources, like servlets, enterprise Java beans, and JMS destinations.

The configuration of the single server might be adequate for your SCA modules. However, you might want to change the configuration; for example, to add extra server capacity, change the transport chains that provide networking services, or provide your own custom services.

- Create servers.

You can create servers using either the Create New Application Server wizard in the administrative console or the `createApplicationServer wsadmin` command.

- Manage available servers.

After creating a server, you can monitor its runtime state, change its configuration, start and stop the server, and perform a range of other administrative tasks to manage the server.

- Configure transport chains.

You need to configure transport chains to provide networking services to such functions as the service integration technologies, WebSphere Secure Caching Proxy, and the high availability manager core group bridge service.

- Develop custom services.

To define a hook point to be run when a server or node agent starts and shuts down, you develop a custom service class and then configure a custom service instance. When the application server or node agent starts, the custom service starts and initializes.

- Define processes for a server.

To enhance the operation of a server, you can define command-line information for starting or initializing an application server process. Such settings define run-time properties such as the program to run, arguments to run the program, and the working directory.

- Manage server use of the Java virtual machine.
As part of configuring a server, you might define settings that enhance your system's use of the Java virtual machine (JVM).

Creating a server:

You can create servers using either the Create New Application Server wizard in the administrative console or the `createApplicationServer wsadmin` command.

Before you can create a server, you must have created a custom profile then federated the node into a deployment manager cell; for example, as described in [Creating a new WebSphere ESB custom profile](#).

Before you can deploy an SCA module, you must create a server that is to run the SCA module.

You can deploy several SCA modules to the same server, so you only need to complete this task if you want to create a new server; for example, to handle more client connections or provide greater message throughput than the one server, or to separate the administration of SCA modules onto different servers.

To create a server, use the administrative console to complete the following steps:

1. Display the list of servers.
In the navigation pane, expand **Servers** → **Application servers**.
2. Click **New**.
3. On the Create New Application Server page, follow the instructions to define your server.
 - a. Select the node on which you want the server to run.
The node must be a WebSphere ESB managed node.
 - b. Type a unique name for the server.
The name must be unique within the node.
 - c. Click **Next**
 - d. Select the **defaultESBServer** template (or an existing server) to create the server.
Instead of the **defaultESBServer** template, you can use an existing ESB server as a template. The new server inherits all properties of the template server.
 - e. Click **Next**
 - f. For the **Generate Unique Http Ports** option, choose whether the new server will have unique ports for each HTTP transport.
By default, the **Generate Unique Http Ports** option is selected. If you select this option, you might need to update the alias list for the virtual host that you plan to use with this server to contain these new port values. If you deselect this option, ensure that the default port values do not conflict with other servers on the same physical machine.
For more information about using unique HTTP ports, see [Configuring HTTP transports](#).
 - g. Click **Next**
 - h. If you create the new server using an existing server as a template, do not select to map applications from the existing server to the new server. By default, this option is disabled.

- i. Click **Next**
 - j. On the Confirm new server page, check the summary. To complete the server creation, click the **Finish**.
If there are settings you wish to change, click **Previous** to review or change the server settings.
4. **Optional:** To use multiple language encoding support in the administrative console, configure an application server with UTF-8 encoding enabled.
 5. Save your changes to the master configuration.

The server is created and added automatically to the SCA* buses for your cell.

The new server appears in the list of servers on the Application Servers page.

You can now manage the server; for example, to start the server or deploy SCA modules.

Note: The server was created with default values for some of its properties. A server has many properties that can be set, and creating a server on the Create New Application Server page specifies values for only some of the important properties. To view all of the properties of your server, and to customize your server further, click on the name of your server on the Application Servers page. This displays the server detail settings page, on which you can view and change properties as needed.

Configuring servers or clusters for the SCA runtime:

Before you can deploy a mediation module onto a server or server cluster, you must have configured that server or cluster for the SCA runtime required. This advanced configuration defines whether queue destinations exploited by the SCA runtime are hosted locally or on a remote server or cluster.

Before starting this task, you must have already created one or more servers or server clusters required:

- The server or cluster on which you want to deploy mediation modules.
- The server or cluster on which you want to create queue destinations for the SCA runtime to exploit.

You can use the same server or cluster in both cases.

For more information about creating servers or server clusters, see the following task descriptions:

- Creating a server
- Creating a server cluster

When you install a mediation module into WebSphere ESB, a number of queue destinations are created for components of the mediation module that use asynchronous interactions. The queue destinations are used by the SCA runtime to hold messages being processed for the mediation module.

To specify where the queue destinations are created, you complete one of the following configuration alternatives for the server or cluster.

- Set the advanced configuration of the server or cluster to host destinations. The configuration defines that this server or cluster can host queue destinations for mediation modules deployed locally or for mediation modules deployed on

other servers or clusters. This configuration also adds the server or cluster as a member of the SCA.SYSTEM bus, and configures the messaging engine for the server or cluster.

To perform this configuration task, see “Configuring a server or cluster to host queue destinations for mediation modules” on page 173.

- Set the advanced configuration of the server or cluster to use remote destinations. The configuration defines that this server or cluster uses a *remote destination location* for destinations.

This server or cluster does not need to be a member of the SCA.SYSTEM bus.

If you deploy a mediation module to this server or cluster, the queue destinations are configured on a different server or cluster that is a member of the SCA.SYSTEM bus.

To perform this configuration task, see “Configuring a server or cluster to use remote destinations for mediation modules” on page 175.

The server or cluster is configured to accept the deployment of mediation modules.

You can now deploy mediations onto the server or cluster or take other actions to manage the server or cluster.

Managing available servers:

After creating a server, you can monitor its runtime state, change its configuration, start and stop the server, and perform a range of other administrative tasks to manage the server.

These tasks are for servers that you have already created.

The running of applications and services depends on the configuration and running of ESB servers. After creating a server, you can perform a range of other administrative tasks to manage the server.

You can manage servers using either the administrative console or a range of wsadmin commands.

This topic provides a summary of alternative ways to manage available servers with the administrative console, and provides links to where the associated tasks are described.

- Display the list of servers.

In the navigation pane, click **Servers** → **Application Servers**.

This displays the Application Servers collection page, which lists the servers, their cell, and the nodes holding the servers. For each server, this page also shows the status that indicates whether a server is started, stopped, or unavailable. You can use this page to perform a number of actions on servers; for example, to create new servers, start or stop servers, or open a page of detail settings for a server.

- Display details about a server.

Display the list of servers on the Application Servers page, then click the name of the server.

This displays the settings page for that server. You can use this page to view detailed information about a server and its resources, or to change the configuration of the server and its resources.

- Start a server.

To be able to run applications and services, a server must be started. If the server is in a deployment manager cell, the node agent must already be running.

- To start a node agent for a managed node, run the command `wesb_install\profiles\profile_name\bin\startNode` where *profile_name* is the name of the custom profile for the managed node.
- To start a server, display the list of servers on the Application Servers collection page, select the checkbox for the server, then click **Start**.

- Monitor the running of servers.

You must monitor the status of run-time components to ensure that after starting they remain operational as needed.

You can browse messages displayed in the server logs. You can also use the **Troubleshooting** → **Logs and Trace** page of the administrative console to monitor the status of run-time components.

- Stop a server.

Stopping a server prevents it being used to run applications and services.

Display the list of servers on the Application Servers collection page, select the checkbox for the server, then click **Stop**.

- “Deleting a server”

Deleting a server deletes the server identity and configuration, including the deployment of applications and service on that server. The server is then no longer available to deploy applications and services.

Display the list of servers on the Application Servers page, select the check box for the server, then click **Delete**.

Deleting a server:

Deleting a server deletes the server identity and configuration, including the deployment of applications and service on that server. The server is then no longer available to deploy applications and services.

If you no longer need a specific server to run applications and services, you can delete the server.

To delete a server, use the administrative console to complete the following steps:

1. In the navigation pane, click **Servers** → **Application Servers** This displays the Application Servers page, which lists the servers and shows the status that indicates whether a server is started, stopped, or unavailable.
2. Ensure that the server is stopped.
To stop the server, select the check box next to the server name, then click **Stop**.
3. Delete the server
To delete the server, select the check box next to the server name, then click **Delete**.
4. Save your changes to the master configuration.

The server is removed from the list on the Application Servers page, and is no longer available to deploy applications and services.

Balancing workloads with server clusters

A server cluster is a set of servers that are managed together and participate in workload management. Administrators can create and configure server clusters in an existing server and bus environment.

Before creating a clustered environment, you should take the following actions:

1. Plan and prepare your environment to use server clusters. For example, consider the following items:
 - Ensure that you have adequate resources to implement clustering successfully.
 - Determine if clustered servers would be beneficial for your applications. If an application has a large number of requests, installing on a cluster could help improve processing throughput. Other reasons to consider installing on a cluster includes availability of the application during scheduled maintenance and during equipment failures.
 - Determine if any of the applications contain services.
 - Familiarize yourself with Network Deployment and clustering support provided by WebSphere Application Server Network Deployment. For more information about the clustering support, see the related topics.
 - Review this task before performing any of the steps.
2. Create a network deployment cell, by creating a Deployment Manager profile, one or more custom profiles, and federating the custom profiles into the deployment manager cell.

If you want to balance workload, such as service requests, over a set of servers, you can create a server cluster, then add servers as members of that cluster. You can also create a *backup cluster*, to provide failover support for the server cluster that it is assigned to.

To assist you in understanding how to configure and use clusters for workload management, consider this scenario. Client requests are distributed among the cluster members on a single machine. (A client refers to any service requester, servlet, Java application, or other program or component that connects the end user with a server.) In more complex workload management scenarios, you can distribute cluster members within the same sysplex.

For more information about managing server clusters, see the following sub-topics, which provide information related to using server clusters with WebSphere ESB and provide links to appropriate WebSphere Application Server topics:

- Create a server cluster.
For more information about creating a server cluster, see “Creating a server cluster.”
- Create several server members of the cluster.
For more information about creating a server member, see “Creating a cluster member” on page 184.
- Manage available clusters.
After creating a server cluster, you can monitor its runtime state, change its configuration, start and stop the cluster, and perform a range of other administrative tasks to manage the cluster.

Creating a server cluster:

Use this task to create a server cluster, which is a set of servers that are managed together and participate in workload management.

Before you can create a server cluster, you must have created a custom profile then federated the node into a deployment manager cell.

If you want to balance workload, such as service requests, over a set of servers, you can create a server cluster, then add servers as members of that cluster. You can also create a *backup cluster*, to provide failover support for the server cluster that it is assigned to.

Note: On the z/OS platform, if you plan to create a cluster of servers that spans multiple systems in a sysplex and has stateful session beans with an activation policy of Transaction deployed in them, the passivation directory should reside on an HFS (hierarchical file system) that is shared across the multiple systems in the sysplex on which the clustered servers are running. .

To create a server cluster, use the administrative console to complete the following steps:

1. In the navigation pane, click **Servers** → **Clusters**.
2. Click **New**.
3. On the Enter basic cluster information. page, specify the following details:
 - a. For Cluster name, type a name for the cluster.
 - b. **Optional:** To disable node-scoped routing optimization, clear Prefer local. By default, the Prefer local option is selected. This indicates that, if possible, Enterprise JavaBean (EJB) requests are routed to the client node. If you enable this feature, performance is improved because client requests are sent to local enterprise beans.
 - c. **Optional:** To create a replication domain for this cluster, select Create a replication domain for this cluster.
Use replication domains to transfer data, objects, or events for session manager, dynamic cache, or stateful session beans among the application servers in a cluster. Create a separate replication domain to use with each component that acts as a consumer of the replication. For example, you can configure one replication domain to use with a session manager and another domain to use with dynamic cache. The replication domain name that is created is identical to the cluster name.
 - d. Under Existing application server, leave the Do not include an existing server in this cluster option selected (the default).
You must create an empty cluster, as specified by this option.
 - e. Click **Next**.
4. On the Create cluster members page, specify details of any cluster members that you want to create with the cluster.
For each new cluster member, perform the following actions:
 - a. For Member name, type the name of a new server to add as a member of the cluster.
 - b. For Select node, select the node on which the server runs.
 - c. Make sure that Generate Unique HTTP Port is selected.
 - d. Under Select template:, select the defaultESBServer template.
 - 1) Select Default application server template
 - 2) From the drop down list, select defaultESBServer.
 - e. Click **Apply** to finish the cluster member. You can add more cluster members. All cluster members you add are based on the same server template, so there are less steps to complete.
5. Click **Next**.
6. On the View the summary. page, to create the cluster click **Finish**.

If any of the details are incorrect, you can click **Previous**, to return to earlier pages and change the details.

7. Define a virtual host with a unique port number.
 - a. In the navigation pane, click **Environment** → **Virtual Hosts**.
 - b. In the content pane, click **default_host** → **[Additional Properties] Host Aliases**.
 - c. Click **New**.
 - d. On the settings page for a virtual host, specify the host name and port number.

Note: Setting up a virtual host is optional on z/OS.

8. Save your changes to the master configuration. As part of saving the change to the configuration, you can select Synchronize changes with Nodes before clicking **Save** on the Save page.
9. Before you can start the cluster, the configuration needs to be synchronized to the nodes. If you selected Synchronize changes with Nodes when saving your configuration in the previous step, you can ignore this step.

Otherwise, take one of the following actions:

- If you are using automatic synchronization, wait until synchronization runs.
- Run manual synchronization to get the configuration files moved to the nodes.
 - a. Click **System administration** → **Nodes**
 - b. On the Nodes page, select the node name.
 - c. Click Synchronize or Full resynchronize.

The Nodes page displays status indicating whether the node is synchronized.

You can display the properties and local topology of the cluster. Click **Servers** → **Clusters** → **[Content pane] cluster_name**.

You can change the configuration of the cluster. Note that if you have not clicked **Save** and saved your administrative configuration, you only see the Configuration and Local Topology tabs. To see the Runtime tab, you must save your administrative configuration. Also, ensure that changes are synchronized to the nodes as described above.

You can create more cluster members or start the cluster. For more information about cluster configuration options, see *Balancing workloads with clusters*.

You can use scripting to automate the task of creating clusters. For information about using scripting for clusters, see *Creating clusters using scripting*.

To manage a server cluster, you can use the Server cluster collection. You can use this page to display the properties and topology of a cluster, and to make changes as needed.

Creating a cluster member:

Use this task to create a server as a member of a server cluster.

Before you can create a cluster member, you must have created the cluster, as described in “Creating a server cluster” on page 182.

If you want a server to take part in the distribution of workload in a server cluster, create a cluster member for the server.

To create a cluster member, use the administrative console to complete the following steps:

1. In the navigation pane, click **Servers** → **Clusters**. This displays the Server Cluster page, which lists the clusters in the deployment manager cell.
2. In the content pane, click the name of the cluster.
3. Click **Cluster members**. This displays the Cluster Members page, which lists members of the cluster, states the nodes on which members reside, and states whether members are started, stopped, or encountering problems.
4. Click **New**, then follow the steps on the Create new cluster members page.
 - a. For Member name, type the name of a new server to add as a member of the cluster.
 - b. For Select node, select the node on which the server runs.
 - c. Make sure that Generate Unique HTTP Port is selected.
 - d. Under Select template:, select the defaultESBServer template.
 - 1) Select Default application server template
 - 2) From the drop down list, select defaultESBServer.
 - e. Click **Apply** to finish the cluster member. You can add more cluster members. All cluster members you add are based on the same server template, so there are less steps to complete.
 - f. Click **Next**.
 - g. Review the summary of information on new cluster members and click **Finish**.
5. Save your changes to the master configuration. As part of saving the change to the configuration, you can select Synchronize changes with Nodes before clicking **Save** on the Save page.
6. To display the properties of a cluster member, click the member's name under Member name on the Cluster members page. This displays the settings page for the cluster member instance.

You created application servers that became members of an existing server cluster.

You can display the properties and local topology of the cluster. Click **Servers** → **Clusters** → [Content pane] *cluster_name*.

You can change the configuration of the cluster. Note that if you have not clicked **Save** and saved your administrative configuration, you only see the Configuration and Local Topology tabs. To see the Runtime tab, you must save your administrative configuration. Also, ensure that changes are synchronized to the nodes as described above.

You can create more cluster members or start the cluster. For more information about cluster configuration options, see Balancing workloads with clusters.

You can use scripting to automate the task of creating cluster members.

To manage cluster members, you can use the Creating clusters using scripting. You can use this page to list the cluster members, and to select members to display more details or to act on.

You can use scripting to automate the task of creating cluster members. For information about using scripting for cluster members, see [Creating cluster members using scripting](#).

Configuring servers or clusters for the SCA runtime:

Before you can deploy a mediation module onto a server or server cluster, you must have configured that server or cluster for the SCA runtime required. This advanced configuration defines whether queue destinations exploited by the SCA runtime are hosted locally or on a remote server or cluster.

Before starting this task, you must have already created one or more servers or server clusters required:

- The server or cluster on which you want to deploy mediation modules.
- The server or cluster on which you want to create queue destinations for the SCA runtime to exploit.

You can use the same server or cluster in both cases.

For more information about creating servers or server clusters, see the following task descriptions:

- [Creating a server](#)
- [Creating a server cluster](#)

When you install a mediation module into WebSphere ESB, a number of queue destinations are created for components of the mediation module that use asynchronous interactions. The queue destinations are used by the SCA runtime to hold messages being processed for the mediation module.

To specify where the queue destinations are created, you complete one of the following configuration alternatives for the server or cluster.

- Set the advanced configuration of the server or cluster to host destinations. The configuration defines that this server or cluster can host queue destinations for mediation modules deployed locally or for mediation modules deployed on other servers or clusters. This configuration also adds the server or cluster as a member of the SCA.SYSTEM bus, and configures the messaging engine for the server or cluster.

To perform this configuration task, see [“Configuring a server or cluster to host queue destinations for mediation modules”](#) on page 173.

- Set the advanced configuration of the server or cluster to use remote destinations. The configuration defines that this server or cluster uses a *remote destination location* for destinations.

This server or cluster does not need to be a member of the SCA.SYSTEM bus.

If you deploy a mediation module to this server or cluster, the queue destinations are configured on a different server or cluster that is a member of the SCA.SYSTEM bus.

To perform this configuration task, see [“Configuring a server or cluster to use remote destinations for mediation modules”](#) on page 175.

The server or cluster is configured to accept the deployment of mediation modules.

You can now deploy mediations onto the server or cluster or take other actions to manage the server or cluster.

Managing available server clusters:

After creating a server cluster, you can monitor its runtime state, change its configuration, start and stop the cluster or cluster members, and perform a range of other administrative tasks to manage the cluster.

The running of applications and services depends on the configuration and running of the cluster and the servers that are members of the cluster. After creating a server cluster, you can perform a range of administrative tasks to manage the cluster.

You can manage clusters using either the administrative console or a range of wsadmin commands.

This topic provides a summary of alternative ways to manage available clusters with the administrative console, and provides links to where the associated tasks are described.

- Display the list of clusters.

In the navigation pane, click **Servers** → **Clusters**.

This displays the Server Cluster collection page, which lists the clusters. For each cluster, this page also shows the status that indicates whether a cluster is started, stopped, or unavailable. You can use this page to perform a number of actions on clusters; for example, to create new clusters, start or stop clusters, or open a page of detail settings for a cluster.

- Display details about a cluster.

Display the list of clusters then, on the Server Cluster collection page, click the name of the cluster.

This displays the settings page for that cluster. You can use this page to view or change the configuration and local topology of a server cluster. If you saved your administrative configuration after creating the server cluster, you can also view runtime information about the server cluster.

- Display the list of servers that are members of a cluster.

Display details about a cluster then, on the Server Cluster settings page, click **Cluster members**.

This displays the Cluster Members collection page, which lists the members of the cluster, states the nodes on which members reside, and states whether members are started, stopped or encountering problems. You can use this page to perform a number of actions on cluster members; for example, to create new cluster members, or open a page of detail settings for a cluster member.

- Create new members of a cluster

Display the Cluster Members collection page, then click **Cluster members**. The Cluster members page lists members of a cluster, states the nodes on which members reside, and states whether members are started, stopped or encountering problems.

- Start a cluster.

To be able to run applications and services, a cluster must be started.

Display the list of clusters then, on the Server Cluster collection page, select the checkbox for the cluster, then click **Start** or **Ripplestart**.

- Monitor the running of clusters and their members.

You must monitor the status of run-time components to ensure that after starting they remain operational as needed.

You can browse messages displayed in the cluster logs. You can also use the **Troubleshooting** → **Logs and Trace** page of the administrative console to monitor the status of run-time components.

- Tune the behavior of the workload management run time

If your application is experiencing problems with timeouts or your network experiences extreme latency, change the timeout interval for the `com.ibm.CORBA.RequestTimeout` property. Or, if the workload management state of the client is refreshing too soon or too late, change the interval for the `com.ibm.websphere.wlm.unusable.interval` property.

- Stop a cluster.

Stopping a server cluster prevents it being used to run applications and services.

Display the list of clusters then, on the Server Cluster collection page, select the checkbox for the cluster, then click **Stop** or **Immediate Stop**.

- Delete a cluster member.

Deleting a cluster member removes the server from the cluster, and deletes the server. You cannot delete a cluster member without deleting the server.

Display the list of cluster members then, on the Cluster Members collection page, select the checkbox for the cluster member, then click **Delete**.

- Delete a cluster.

If you delete a server cluster, the action deletes the cluster and all its cluster members. When you delete a cluster, there is no option to keep certain cluster members or applications that you have installed on any part of the cluster

Display the list of clusters then, on the Server Cluster collection page, select the checkbox for the cluster, then click **Delete**.

Doing more with bus topologies

The `SCA.SYSTEM` bus used for SCA modules is a standard service integration bus, as used in WebSphere Application Server. Besides the `SCA.SYSTEM` bus, you can also create other service integration buses to support service integration logic or other applications.

For example, the `SCA.APPLICATION.cell_name.Bus` is provided and used to define JMS queue destinations and other JMS resources for modules deployed with JMS bindings.

You can create other buses for use as in WebSphere Application Server; for example, for applications acting as service requesters and providers within WebSphere ESB, or to link a bus to WebSphere MQ. You can also use a WebSphere ESB deployment manager to manage separate application servers, for use with applications and modules deployed onto WebSphere Application Server.

Managing service applications

You can manage service applications from the WebSphere ESB administrative console. Service applications provide services, and have an associated Service Component Architecture (SCA) module.

The kind of SCA modules that WebSphere ESB supports are called mediation modules. Mediation modules provide a simple way to change the format, content or target of service requests and responses.

Getting started with service applications

After deploying service applications you can view and manage components of your service applications.

You can view and manage the applications and the associated Service Component Architecture (SCA) modules. The kind of SCA modules that WebSphere ESB supports are called mediation modules.

You can list all the SCA modules that you have deployed, and display details of how the SCA modules connect to service requesters and service providers.

Overview of the application integration environment:

A service application has an associated Service Component Architecture (SCA) module. You deploy service applications, to WebSphere ESB, within EAR (Enterprise ARchive) files.

Deploying a service application

The process of deploying an EAR file containing a service application is the same as the process of deploying any other EAR file. However, after you have deployed an EAR file containing an SCA module, you can view details about the following.

- The service application.
- The SCA module associated with the service application.
 - You can see how an SCA module is connected to service requesters and service providers. SCA modules are connected to service requesters through exports, and to service providers through imports.

Viewing SCA module details

The SCA module details that you can view include some of the following. The precise details you can display depend upon the SCA module.

- SCA module name.
- SCA module description.
- Associated application name.
- SCA module imports.
 - Interfaces.
 - Import interfaces are abstract definitions that describe how an SCA module accesses a service.

Note: WebSphere ESB supports WSDL interfaces, it does not support Java interfaces.

- Bindings.
 - Import bindings are concrete definitions. They specify the physical mechanism by which an SCA module accesses a service. For example, using SOAP/HTTP.
- SCA module exports.
 - Interfaces.
 - Export interfaces are abstract definitions that describe how service requesters access an SCA module.

Note: WebSphere ESB supports WSDL interfaces, it does not support Java interfaces.

- Bindings.

- Export bindings are concrete definitions. They specify the physical mechanism by which a service requester accesses an SCA module, and indirectly, a service.

If a default import binding connects two SCA modules then you can change the binding's target using WebSphere ESB. Modifying SCA import bindings lets you invoke different service providers.

Learning about service applications:

Service applications provide services, and have an associated Service Component Architecture (SCA) module. SCA modules encapsulate services, so you can make changes to services without impacting users of the service. The kind of SCA modules that WebSphere ESB supports are called mediation modules.

Mediation modules:

Mediation modules are Service Component Architecture (SCA) modules that can change the format, content or target of service requests.

Mediation modules operate on messages that are in flight between service requesters and service providers. They allow you to route messages to different service providers. They also let you transform messages: you can amend message content or form. In addition, mediation modules can provide functions such as message logging, and error processing that is tailored to your requirements.

Components of mediation modules

Among the items that mediation modules contain are the following:

- Imports.
 - Imports define interactions between Service Component Architecture (SCA) modules and service providers.
 - Imports allow SCA modules to call external services as if they were local.
 - Mediation module imports can be viewed from WebSphere ESB and if the import binding is an SCA binding, then it can be modified to point to another SCA module.
- Exports.
 - Exports define interactions between Service Component Architecture (SCA) modules and service requesters.
 - Exports allow an SCA module to offer a service. Exports define the external interfaces (access points) of an SCA module.
 - Mediation module exports can be viewed from WebSphere ESB.
- Service Component Architecture (SCA) components.
 - SCA components, or service components, are SCA building blocks. You build SCA modules such as mediation modules, using SCA components. You can create and customize SCA modules and components graphically, using WebSphere Integration Developer.
 - Typically, mediation modules contain a specific type of SCA component called a mediation flow component. A mediation module can contain, at most, one mediation flow component.
 - A mediation flow component can contain one mediation primitive, a number of mediation primitives or no mediation primitives. WebSphere ESB supports a supplied set of mediation primitives that provide functionality for message

routing and transformation. One of the mediation primitives that WebSphere ESB supports allows you to invoke custom logic.

- A mediation module does not have to contain a mediation flow component. The purpose of a mediation module that does not contain a mediation flow component is to transform service requests from one protocol to another. For example, a service request might be made using SOAP/JMS but need transforming to SOAP/HTTP before sending on.

Note: You can view mediation modules from WebSphere ESB. You can also make limited changes to mediation modules from WebSphere ESB. However, you cannot view or change SCA components or mediation primitives from WebSphere ESB. Use WebSphere Integration Developer to customize SCA components and mediation primitives.

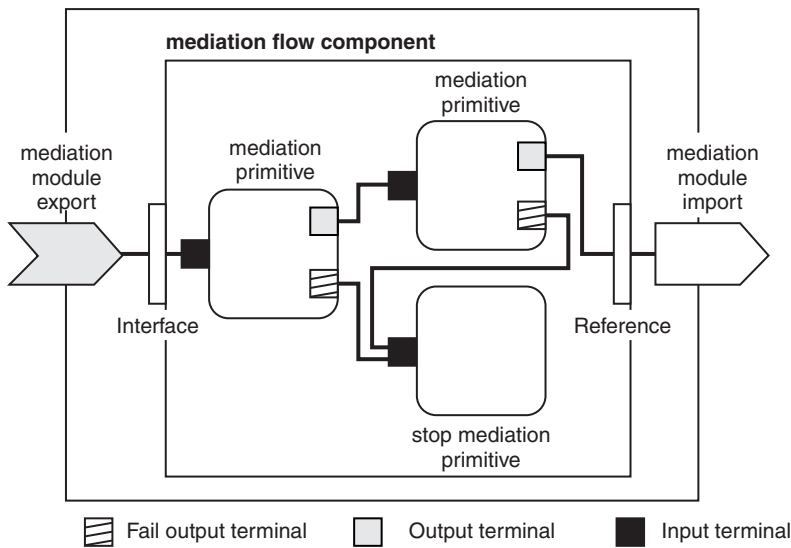


Figure 15. Simplified example of a mediation module. The mediation module contains one mediation flow component. The mediation flow component contains mediation primitives.

Deploying mediation modules

Mediation modules are created using WebSphere Integration Developer, and deployed to WebSphere ESB inside an enterprise archive (EAR) file. Therefore, a mediation module is deployed to WebSphere ESB in the same way you deploy enterprise applications.

WebSphere Integration Developer packages mediation modules inside Java archive (JAR) files, and the JAR files are then stored inside EAR files.

Logically, mediation modules can be thought of as one entity. In reality, SCA modules are defined by a number of XML files stored in one JAR file.

- EAR file.
 - Contains JAR file.
 - Contains Mediation module.

Example of EAR file, containing a mediation module

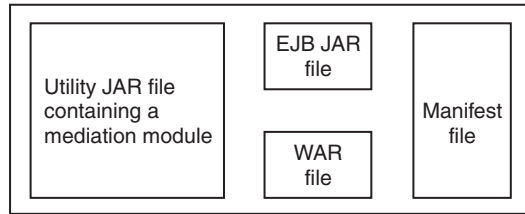


Figure 16. Simplified example of an EAR file containing a mediation module. The EAR file contains JAR files. The utility JAR file contains a mediation module.

Imports and import bindings:

Imports let a Service Component Architecture (SCA) module access external services, (services that are outside the SCA module) as if they were local. Imports define interactions between SCA modules and service providers. Import bindings define the specific way that an external service is accessed.

SCA modules are not required to have imports, if they do not need to access external services. Mediation modules are types of SCA modules.

Interfaces and bindings

An SCA module import needs at least one interface, and an SCA module import has only one binding.

- Interfaces.
 - Import interfaces are abstract definitions. They define access points.
 - Import interfaces are defined using Web Services Description Language (WSDL), an XML language for describing Web services.
 - An SCA module can have many import interfaces.
- Bindings.
 - Import bindings are concrete definitions. They specify the physical mechanism that SCA modules use to access an external service.

Supported import bindings

WebSphere ESB supports the following import bindings.

- Web Service Bindings
 - SOAP/HTTP
 - SOAP/JMS
- SCA Bindings
 - SCA modules can have SCA bindings. SCA bindings connect SCA modules to other SCA modules.
- Java Message Service (JMS) 1.1 Bindings
 - JMS allows interoperability with the WebSphere family.
 - JMS can exploit various transport types, including: TCP/IP and HTTP(S).
 - There are predefined JMS bindings that support JMS text messages containing Business Object (BO) XML. The predefined JMS bindings also support JMS object messages containing serialized Java Business Objects.

- You can use JMS custom bindings to support other types of JMS message. However, custom bindings require some coding to translate the message.
- WebSphere Adapter Bindings
 - WebSphere Adapters enable interaction with Enterprise Information Systems (EIS).

Exports and export bindings:

Exports let a Service Component Architecture (SCA) module offer a service to others; they define interactions between SCA modules and service requesters. Export bindings define the specific way that an SCA module is accessed by others.

Mediation modules are types of SCA modules.

Interfaces and bindings

An SCA module export needs at least one interface.

- Interfaces.
 - Export interfaces are abstract definitions. They define access points.
 - Export interfaces are defined using Web Services Description Language (WSDL), an XML language for describing Web services.
 - An SCA module can have many export interfaces.
- Bindings.
 - Export bindings are concrete definitions. They specify the physical mechanism that service requesters use to access a service.
 - Typically, an SCA module export has one binding specified. An export with no binding specified is interpreted by the runtime as an export with an SCA binding.

Supported export bindings

WebSphere ESB supports the following export bindings.

- Web Service Bindings
 - SOAP/HTTP
 - SOAP/JMS
- SCA Bindings
 - SCA modules can have SCA bindings. SCA bindings connect SCA modules to other SCA modules.
- Java Message Service (JMS) 1.1 Bindings
 - JMS allows interoperability with the WebSphere family.
 - JMS can exploit various transport types, including: TCP/IP and HTTP(S).
 - There are predefined JMS bindings that support JMS text messages containing Business Object (BO) XML. The predefined JMS bindings also support JMS object messages containing serialized Java Business Objects.
 - You can use JMS custom bindings to support other types of JMS message. However, custom bindings require some coding to translate the message.
- WebSphere Adapter Bindings
 - WebSphere Adapters enable interaction with Enterprise Information Systems (EIS).

Service application features of the administrative interfaces:

WebSphere ESB allows you to view and change aspects of service applications using the administrative console.

Service applications provide services, and have an associated Service Component Architecture (SCA) module. The type of SCA modules that are supported by WebSphere ESB are mediation modules.

Viewable SCA module details

After you have deployed an EAR (Enterprise ARchive) file containing an Service Component Architecture (SCA) module, you can view SCA module details. You can list all your SCA modules, and their associated applications, and you can view details about a particular SCA module.

The SCA module details you can view include some of the following.

- SCA module name.
- Associated application.
- SCA module imports.
 - Interfaces.
 - Bindings.
- SCA module exports.
 - Interfaces.
 - Bindings.

Modifiable SCA module details

After you have deployed an EAR file containing an SCA module you can change the following SCA module details using the administrative console.

- Import bindings of type SCA.
 - Changing import bindings lets you change service interactions.
 - SCA bindings connect SCA modules to other SCA modules.
 - One SCA module can interact with a second SCA module, and be changed to interact with a third SCA module.

SCA modules can have different types of bindings. WebSphere ESB supports many different types of SCA module bindings. For example, web service bindings and SCA bindings. However, you can only change SCA import bindings from WebSphere ESB.

Note: An export with no binding specified is interpreted by the runtime as an export with an SCA binding.

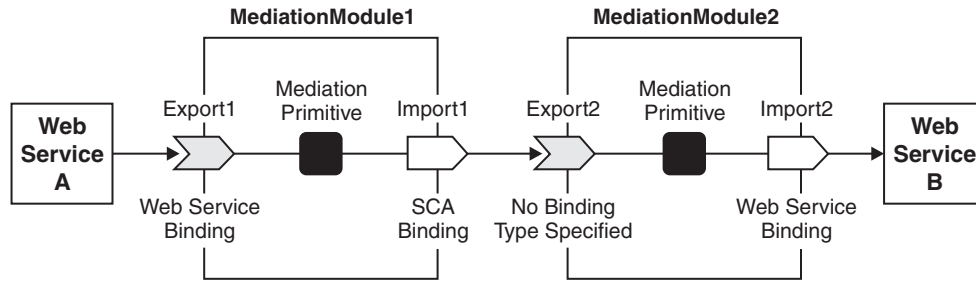


Figure 17. Example showing one mediation module interacting with another mediation module. MediationModule1 connects to MediationModule2

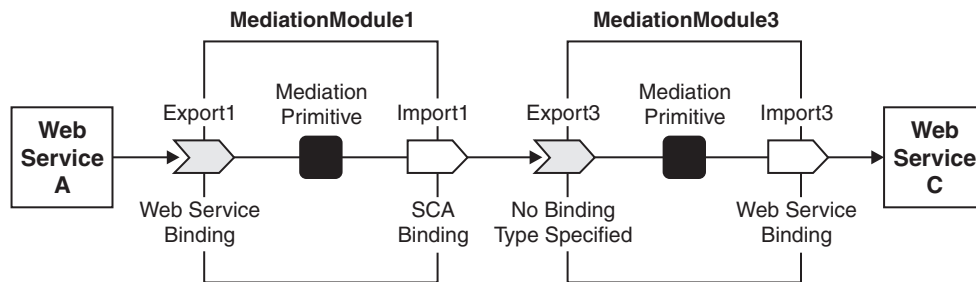


Figure 18. Example showing one mediation module interacting with another mediation module. MediationModule1 connects to MediationModule3

Installing a module on a production server

This topic describes the steps involved in taking an application from a test server and deploying it into a production environment.

Before deploying a service application to a production server, assemble and test the application on a test server. After testing, export the relevant files as described in Preparing to deploy to a server and bring the files to the production system to deploy. See the information centers for WebSphere Integration Developer and WebSphere Application Server Network Deployment, version 6 for more information.

1. Copy the module and other files onto the production server.
The modules and resources (EAR, JAR, RAR, and WAR files) needed by the application are moved to your production environment.
2. Run the serviceDeploy command to create an installable EAR file.
This step defines the module to the server in preparation for installing the application into production.
 - a. Locate the JAR file that contains the module to deploy.
 - b. Issue the command using the JAR file from the previous step as input.
3. Install the EAR file from step 2. How you install the applications depends on whether you are installing the application on a stand alone server or a server in a cell.

Note: You can either use the administrative console or a script to install the application. See the WebSphere Application Server information center for additional information.

4. Save the configuration. The module is now installed as an application.

5. Start the application.

The application is now active and work should flow through the module.

Monitor the application to make sure the server is processing requests correctly.

Installing a mediation module EAR file with the console:

Installing a mediation module consists of moving the installable EAR file, for the mediation module, onto a server or cluster that will host the mediation module. Installed mediation modules that start and run properly are considered *deployed*.

If you have exported your mediation module to a JAR file then use the `serviceDeploy` command to create an installable EAR file from the mediation module JAR file. For more information about creating an installable EAR file for a mediation module, see “Installing a module on a production server” on page 195.

To enable WebSphere ESB to use the functions provided by a mediation module to integrate applications and services, you need to install the EAR file for the module into a server or cluster, then start the deployed module.

This topic describes how to use the administrative console to install a mediation module EAR file. Alternatively, you can also use other methods, like the `install` or `installinteractive` command with the `wsadmin` tool, in the same way as you install enterprise application files into WebSphere Application Server.

Important: After you start performing the steps below, click **Cancel** to exit if you decide not to install the application. Do not simply move to another administrative console page without first clicking **Cancel** on an application installation page.

To use the administrative console to install a mediation module EAR file, complete the following steps:

1. Click **Applications** → **Install New Application** in the console navigation tree. The first of two Preparing for application installation pages is displayed.
2. On the first Preparing for application installation page, complete the following substeps:
 - a. Specify the full path name of the mediation module EAR file (.ear file). The EAR file that you are installing can be either on the client machine (the machine that runs the administrative console Web browser) or on the server machine (the machine to which the client is connected). If you specify an EAR file on the client machine, then the administrative console uploads the EAR file to the machine on which the console is running and proceeds with application installation.
 - b. Click **Next**.
3. On the second Preparing for application installation page, complete the following substeps:
 - a. Select whether to generate default bindings.

Using the default bindings causes any incomplete bindings in the application to be filled in with default values. Existing bindings are not altered.

You can customize default values used in generating default bindings.
 - b. Click **Next**. If security warnings are displayed, click **Continue**. The Install New Application pages are displayed. If you chose to generate default

bindings, and your application does not use a resource adapter, you can proceed to the Summary step (last step below).

4. On the Step: Select installation options panel, provide values for the following settings. For more information about the settings on this page, see Select installation options settings. Default values are used if you do not specify a value.
 - a. For Directory to install application, specify the directory into which the application EAR file will be installed. The default value is the value of *APP_INSTALL_ROOT/cell_name*, where the *APP_INSTALL_ROOT* variable is *install_root/installedApps*. For example, for WebSphere ESB installed on Windows into *C:\Program Files\WESB*, the default location is *C:\Program Files\WESB\profile_name\installedApps\cell_name*.

Note: If an installation directory is not specified when an application is installed in a stand-alone profile, the application is installed in *APP_INSTALL_ROOT/base_cell_name*. If you add the stand-alone server to a deployment manager cell, the cell name of the new server configuration becomes the cell name of the deployment manager node. If the *-includeapps* option is used for the *addNode* utility, then the applications that are installed prior to the *addNode* operation still use the installation directory *APP_INSTALL_ROOT/base_cell_name*. However, an application that is installed after the stand-alone server has been added to the network configuration uses the default installation directory *APP_INSTALL_ROOT/network_cell_name*. To move the application to the *APP_INSTALL_ROOT/network_cell_name* location upon running the *addNode* operation, you should explicitly specify the installation directory as *\${APP_INSTALL_ROOT}/\${CELL}* during installation. In such a case, the application files can always be found under *APP_INSTALL_ROOT/current_cell_name*.

- b. For Distribute application, specify whether WebSphere ESB expands or deletes application binaries in the installation destination. The default is to enable application distribution. As a result, when you save changes in the console, application binaries for newly installed applications are expanded to the directory specified. The binaries are also deleted when you uninstall and save changes to the configuration. If you disable this option, then you must ensure that the application binaries are expanded appropriately in the destination directories of all nodes where the application is expected to run.

Important: If you disable this option and you do not copy and expand the application binaries to the nodes, a later saving of the configuration or manual synchronization does not move the application binaries to the nodes for you.

- c. For Use Binary Configuration, specify whether the server or cluster uses the binding, extensions, and deployment descriptors located with the application deployment document, the *deployment.xml* file (default), or those located in the EAR file.
 - d. For **Application name**, type a name for the application. Application names must be unique within a cell and cannot contain characters that are not allowed in object names. For a list of characters that are not allowed in object names, see Object names.

- e. For **Create MBeans for resources**, specify whether to create MBeans for various resources (such as servlets or JSP files) within an application when the application is started. The default is to create MBean instances.
- f. For **Enable class reloading**, specify whether to enable class reloading when application files are updated. The default is not to enable class reloading. Enabling class reloading sets `reloadEnabled` to `true` in the `deployment.xml` file for the mediation module. If a mediation module's class definition changes, the server run time stops and starts the application to reload application classes.
- g. For **Reload interval in seconds**, specify the number of seconds to scan the application's file system for updated files. The default is the value of the `reload interval` attribute in the IBM extension (`META-INF/ibm-application-ext.xmi`) file of the EAR file. To enable reloading, specify a value greater than zero (for example, 1 to 2147483647). To disable reloading, specify zero (0).

The reload interval specified here takes effect only if class reloading is enabled.

- h. For **Deploy Web services**, specify whether the Web services deploy tool `wsdeploy` runs during application installation. The tool generates code needed to run applications using Web services. The default is not to run the `wsdeploy` tool. You must enable this setting if the EAR file contains modules using Web services and has not previously had the `wsdeploy` tool run on it, either from the **Deploy** menu choice of an assembly tool or from a command line.
 - i. For **Validate Input off/warn/fail**, specify whether WebSphere ESB examines the application references specified during application installation or updating and, if validation is enabled, warns you of incorrect references or fails the operation. An application typically refers to resources using data sources for container managed persistence (CMP) beans or using resource references or resource environment references defined in deployment descriptors. The validation checks whether the resource referred to by the application is defined in the scope of the deployment target of that application. Select **off** for no resource validation, **warn** for warning messages about incorrect resource references, or **fail** to stop operations that fail as a result of incorrect resource references.
 - j. For **Process embedded configuration**, specify whether the embedded configuration should be processed. An embedded configuration consists of files such as `resource.xml` and `variables.xml`. When selected or `true`, the embedded configuration is loaded to the application scope from the `.ear` file. If the `.ear` file does not contain an embedded configuration, the default is `false`. If the `.ear` file contains an embedded configuration, the default is `true`.
5. On the Step: Map modules to servers panel, for every module select a target server or cluster from the Clusters and Servers list. Select the check box beside Module to select the mediation module.

If the application uses a WebSphere Adapter, specify target servers or clusters for each RAR file. Also map all other modules that use the resource adapters defined in the RAR modules to the same targets.

Note: When installing a RAR file onto a server, WebSphere ESB looks for the manifest (MANIFEST.MF) for the connector module. It looks first in the `connectorModule.jar` file for the RAR file and loads the manifest from the `_connectorModule.jar` file. If the class path entry is in the manifest from the `connectorModule.jar` file, then the RAR uses that class path. To

ensure that the installed connector module finds the classes and resources that it needs, check the Class path setting for the RAR using the console. For more information about the Class path setting, see the Resource Adapter settings and WebSphere relational resource adapter settings for the administrative console.

You can specify Web servers as targets that route requests to the application. The plug-in configuration file `plugin-cfg.xml` for each Web server is generated based on the applications which are routed through it. If you want a Web server to serve the application, use the **Ctrl** key to select an application server or cluster and the Web server together in order to have the plug-in configuration file `plugin-cfg.xml` for that Web server generated based on the applications which are routed through it.

6. If your application defines resource references, for **Step: Map resource references to resources**, specify JNDI names for the resources that represent the logical names defined in resource references. Each resource reference defined in the application must be bound to a resource defined in your WebSphere ESB configuration before clicking on **Finish** on the Summary panel.
 - a. **Optional:** Specify login configuration name and authentication properties for the resource.
 - b. Click **OK** to save the values and return to the mapping step.
7. If your application uses Web modules, for **Step: Map virtual hosts for web modules**, select a virtual host from the list that should map to a Web module defined in the application.

The port number specified in the virtual host definition is used in the URL that is used to access artifacts such as servlets and JSP files in the Web module. Each Web module must have a virtual host to which it maps. Not specifying all needed virtual hosts will result in a validation error displaying after you click **Finish** on the Summary panel.

8. If the application has security roles defined in its deployment descriptor then, for **Step: Map security roles to users/groups**, specify users and groups that are mapped to each of the security roles.

Select **Role** to select all of the roles or select individual roles. For each role, you select one of the following choices for how security should be applied:

Option	Description
Everyone	This is equivalent to no security.
All authenticated	Anyone who authenticates with a valid user name and password is a member of the role.
Mapped users	Individual users are listed as members of the role.
Mapped groups	Groups are the most convenient way to add the users, every member of the identified groups becomes a member of the role.

For **Mapped users** or **Mapped groups**, to select specific users or groups from the user registry, complete the following sub-steps:

- a. Select a role and click **Lookup users** or **Lookup groups**.
- b. On the Lookup users/groups panel displayed, enter search criteria to extract a list of users or groups from the user registry.
- c. Select individual users or groups from the results displayed.

- d. Click **OK** to map the selected users or groups to the role selected on the **Step: Map security roles to users/groups** panel.
9. If the application has Run As roles defined in its deployment descriptor, for **Step: Map RunAs roles to user**, specify the Run As user name and password for every Run As role. Run As roles are used by enterprise beans that must run as a particular role while interacting with another enterprise bean. Select **Role** to select all of the roles or select individual roles. After selecting a role, enter values for the user name, password, and verify password and click **Apply**.
10. If your application contains resource environment references, for **Step: Map resource environment references to resources**, specify JNDI names of resources that map to the logical names defined in resource environment references. If each resource environment reference does not have a resource associated with it, after you click **Finish** a validation error is displayed.
11. If your application defines **Run-As Identity** as *System Identity*, for **Step: Replace RunAs System to RunAs Roles**, you can optionally change it to *Run-As role* and specify a user name and password for the Run As role specified. Selecting *System Identity* implies that the invocation is done using the WebSphere Application Server security server ID and should be used with caution as this ID has more privileges.
12. If your application has resource references that map to resources that have an Oracle database doing backend processing, for **Step: Specify the isolation level for Oracle type provider**, specify or correct the isolation level to be used for such resources when used by the application. Oracle databases support ReadCommitted and Serializable isolation levels only.
13. On the Summary panel, verify the cell, node, and server onto which the application modules will install:
 - a. Beside **Cell/Node/Server**, click **Click here**.
 - b. Verify the settings.
 - c. Click **Finish**.

Several messages are displayed, indicating whether your application file is installing successfully.

If you receive an OutOfMemory exception and the source application file does not install, your system might not have enough memory or your application might have too many modules in it to install successfully onto the server. If lack of system memory is not the cause of the exception, package your application again so the .ear file has fewer modules. If lack of system memory and the number of modules are not the cause of the exception, check the options you specified on the Java Virtual Machine page of the application server running the administrative console. Then, try installing the application file again.

Windows During installation certain application files are extracted in the directory represented by the configuration session and, when the configuration is saved, these files are saved in the WebSphere Application Server configuration repository. On Windows machines, there is a limit of 256 characters for file paths. Therefore, the application installation might fail if the path for application files in the configuration session or in the configuration repository exceeds the limit of 256 characters. You might see FileNotFound exceptions with *path name too long* in the message. To overcome such problems, make application names and module URI names shorter in length, which helps reduce the file path length. Then, try installing the application file again.

After the application file installs successfully, complete the following actions:

1. Associate any shared libraries that the application needs to the application.
2. Save the changes to your configuration. The application is registered with the administrative configuration and application files are copied to the target directory, which is *install_root/installedApps/cell_name* by default or the directory that you designate. When installed into a Network Deployment profile, files are copied to remote nodes when the configuration on the deployment manager synchronizes with the configuration on individual nodes.
3. If the module is deployed on a server cluster, click **Rollout Update** on the Enterprise Applications page to propagate the changed configuration on all members of the cluster. Rollout Update sequentially updates the configuration on the nodes that contain cluster members.

To enable WebSphere ESB to use the functions provided by a mediation module to integrate applications and services, the deployed module must be started. You can start the module manually or configure it to start automatically. You can also administer the module in other ways; for example, to change the configuration of the module, to stop or update the module, and otherwise manage its activity.

Deploying applications using ANT tasks:

This topic describes how to use ANT tasks to automate the deployment of applications to WebSphere Process Server. By using ANT tasks, you can define the deployment of multiple applications and have them run unattended on a server.

This task assumes the following:

- The applications being deployed have already been developed and tested.
- The applications are to be installed on the same server or servers.
- You have some knowledge of ANT tasks.
- You understand the deployment process.

Information about developing and testing applications is located in the WebSphere Integration Developer information center.

The reference portion of the information center for WebSphere Application Server Network Deployment, version 6 contains a section on application programming interfaces. ANT tasks are described in the package `com.ibm.websphere.ant.tasks`. For the purpose of this topic, the tasks of interest are `ServiceDeploy` and `InstallApplication`.

If you need to install multiple applications concurrently, develop an ANT task before deployment. The ANT task can then deploy and install the applications on the servers without your involvement in the process.

1. Identify the applications to deploy.
2. Create a JAR file for each application.
3. Copy the JAR files to the target servers.
4. Create an ANT task to run the `ServiceDeploy` command to create the EAR file for each server.
5. Create an ANT task to run the `InstallApplication` command for each EAR file from step 4 on the applicable servers.
6. Run the `ServiceDeploy` ANT task to create the EAR file for the applications.
7. Run the `InstallApplication` ANT task to install the EAR files from step 6.

The applications are correctly deployed on the target servers.

Example of deploying an application unattended

This example shows an ANT task contained in a file myBuildScript.xml.

```
<?xml version="1.0">
<project name="OwnTaskExample" default="main" basedir=".">
  <taskdef name="servicedeploy"
    classname="com.ibm.websphere.ant.tasks.ServiceDeployTask" />
  <target name="main" depends="main2">
    <servicedeploy scaModule="c:/synctest/SyncTargetJAR"
      ignoreErrors="true"
      outputApplication="c:/synctest/SyncTargetEAREAR"
      workingDirectory="c:/synctest"
      noJ2eeDeploy="true"
      cleanStagingModules="true"/>
  </target>
</project>
```

This statement shows how to invoke the ANT task.

```
${WAS}/bin/ws_ant -f myBuildScript.xml
```

Tip: Multiple applications can be deployed unattended by adding additional project statements into the file.

Use the administrative console to verify that the newly installed applications are started and processing the workflow correctly.

Managing mediation modules

You can list the mediation modules that have been deployed to WebSphere ESB. You can also view information associated with individual mediation modules and make changes to some import bindings.

After deploying service applications, you can view and manage the associated Service Component Architecture (SCA) modules. Mediation modules are types of Service Component Architecture (SCA) modules.

Working with mediation modules:

You can list the mediation modules that have been deployed to WebSphere ESB. You can also start or stop mediation modules, and view details of a mediation module or its application.

To display the mediation modules that you have deployed, use the administrative console to complete the following steps.

Note: Mediation modules are types of Service Component Architecture (SCA) modules.

List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.

This displays, in the content pane, the mediation modules that have been deployed to WebSphere ESB. You can also see the applications that the mediation modules are associated with, and whether the applications are running.

Displaying details of a mediation module:

You can display information on mediation modules that have been deployed to WebSphere ESB.

To display details about the mediation modules that you have deployed, use the administrative console to complete the following steps.

1. List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.
2. Select the particular SCA module that you are interested in.

This displays, in the content pane, the SCA module name and description; and the name of the associated enterprise application. Also displayed are expandable lists of imports and exports.

Displaying details of the application for a mediation module:

You can display details about the application used to deploy a mediation module to WebSphere ESB.

The application used to deploy a mediation module defines a range of configuration properties that affect the use of the mediation module and associated components. When you installed the application, you specified most, if not all, of its property values.

After installing an application, you might want to review the properties and, if needed, change some of the values.

To display details about the application used to deploy a mediation module, use the administrative console to complete the following steps.

1. List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.
2. For the SCA module that you are interested in, click the application name.
The application name is listed in the column labelled Application.

This displays, in the content pane, the application details page that provides the application's configuration properties and, if appropriate, local topology. From this page, you can modify property values and link to additional console pages to further review and change the application's configuration.

You can review and, if needed, change the configuration properties for the application, as described in *Configuring an application*.

Starting and stopping mediation modules:

You can start a mediation module that is not running (has a status of Stopped) or stop an module that is running (has a status of Started). To change the status of a mediation module, you start or stop the application used to deploy the module.

Before you can start or stop the application for a mediation module, you must have deployed the mediation module into WebSphere ESB, as described in "Installing a module on a production server" on page 195. This installs the application onto an appropriate server (or server cluster).

To use the services of a mediation module and associated components, you start the associated application. By default, the application starts automatically when the server starts.

You can start and stop applications manually using the following:

- Administrative console
- wsadmin startApplication and stopApplication commands
- Java programs that use ApplicationManager or AppManagement MBeans

To start or stop a mediation module, you can use the administrative console to complete the following steps.

1. List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.
2. Select the check box for the SCA module that you want started or stopped.
3. Click the Start or Stop button

Option	Description
Start	Runs the application and changes the state of the application to <i>Started</i> . The status is changed to <i>partially started</i> if not all servers on which the application is deployed are running.
Stop	Stops the processing of the application and changes the state of the application to <i>Stopped</i> .

To restart a running application, select the application you want to restart, click **Stop** and then click **Start**.

The status of the application changes and a message stating that the application started or stopped displays at the top the page.

You can change whether or not an application starts automatically when the server on which it resides starts. For more information about starting and stopping WebSphere applications, see Starting and stopping applications.

Working with imports:

You can list the imports of mediation modules that have been deployed to WebSphere ESB. You can also display import interfaces and change SCA import bindings.

To list the imports of mediation modules that you have deployed, use the administrative console to complete the following steps.

1. List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.
2. Select the particular SCA module that you are interested in.
3. List the SCA module imports. In the content pane, under Module components, expand **Imports**.

A list of imports is displayed, in the content pane. If there are no imports then a information message is displayed.

Displaying an import interface:

You can display the import interfaces of mediation modules that have been deployed to WebSphere ESB.

To display the import interfaces of mediation modules that you have deployed, use the administrative console to complete the following steps.

1. List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.
2. Select an SCA module.
3. List the SCA module imports. In the content pane, under Module components, expand **Imports**.
4. Display import details. In the content pane, under Module components expand the import you require.
5. Display import interfaces. Expand **Interfaces**.
6. Select an interface.

The WSDL (Web Services Description Language) interface is displayed, in the content pane.

Displaying an import binding:

You can display the import bindings of mediation modules that have been deployed to WebSphere ESB.

To display the import bindings of mediation modules, use the administrative console to complete the following steps.

1. List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.
2. Select an SCA module.
3. List the SCA module imports. In the content pane, under Module components, expand **Imports**.
4. Display import details. In the content pane, under Module components expand the import you require.
5. Display import bindings. Expand **Binding**. If the binding is a JMS binding or an Adapter binding then the binding type is displayed. The binding type is not selectable.
6. **Optional:** Select a binding. If the binding is a Web service binding or an SCA Import binding then you can select the binding to get import binding details.

The import binding details are displayed, in the content pane. The details displayed depend on the type of binding.

If the binding is a Web service binding then the service and port names are displayed. If the binding is an SCA Import binding then the details displayed include the current target module. Also displayed are the current target exports and the current export's interfaces. The target modules displayed are the SCA modules that have been deployed to WebSphere ESB and they are displayed in a drop down menu. If you select a different target module then the list of target exports, and export interfaces, changes. If you select a different target export then the list of export interfaces changes.

Changing an import binding:

In some cases, you can change the import bindings of mediation modules from WebSphere ESB. You can change import bindings if they are SCA import bindings, but not if they are other types of bindings.

To change SCA import bindings of mediation modules, use the administrative console to complete the following steps.

1. List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.
2. Select an SCA module.
3. List the SCA module imports. In the content pane, under Module components, expand **Imports**.
4. Display import details. In the content pane, under Module components expand the import you require.
5. Display import bindings. Expand **Binding**.
6. Select an SCA import binding. SCA import bindings are indicated using the identifier [SCA].
7. Select a new target SCA module. Select a module from the **Target** drop down menu. Selecting a different SCA module changes the exports and export interfaces that are displayed.
8. Select an export. Select an export from the **Export** drop down menu.
9. Save your changes. Click **OK**. Then save your changes to the master configuration.

Changes the SCA import binding for a particular SCA module import.

WebSphere ESB issues a warning for each import interface that is not satisfied by an export interface. WebSphere ESB compares the WSDL (Web Services Description Language) port type names of import and export. If the port type names are not the same then a warning is issued, but you are allowed to ignore the warning. However, if the port type names do match then WebSphere ESB assumes that the operations provided are equivalent and no warning is issued.

Working with exports:

You can list the exports of mediation modules that have been deployed to WebSphere ESB. You can also display export interfaces and export bindings.

To list the exports of mediation modules that you have deployed, use the administrative console to complete the following steps.

1. List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.
2. Select the particular SCA module that you are interested in.
3. List the SCA module exports. In the content pane, under Module components, expand **Exports**.

A list of exports is displayed, in the content pane. If there are no exports then an information message is displayed.

Displaying an export interface:

You can display the export interfaces of mediation modules that have been deployed to WebSphere ESB.

To display the export interfaces of mediation modules that you have deployed, use the administrative console to complete the following steps.

1. List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.
2. Select an SCA module.

3. List the SCA module exports. In the content pane, under Module components, expand **Exports**.
4. Display export details. In the content pane, under Module components expand the export you require.
5. Display export interfaces. Expand **Interfaces**.
6. Select an interface.

The WSDL (Web Services Description Language) interface is displayed, in the content pane.

Displaying an export binding:

You can display the export bindings of mediation modules that have been deployed to WebSphere ESB.

To display the export bindings of mediation modules, use the administrative console to complete the following steps.

1. List the SCA modules. In the navigation pane, expand **Applications** → **SCA Modules**.
2. Select an SCA module.
3. List the SCA module exports. In the content pane, under Module components, expand **Exports**.
4. Display export details. In the content pane, under Module components expand the export you require.
5. Display export bindings. Expand **Binding**. If the binding is a JMS binding or an Adapter binding then the binding type is displayed. The binding type is not selectable. If an export has no binding specified then the runtime assumes that the binding is of type SCA.
6. Select a binding. If the binding is a Web service binding then you can select the binding to get export binding details.

The export binding details are displayed, in the content pane. A Web service binding displays service and port names.

Managing the throughput of SCA requests

For each SCA module deployed on WebSphere ESB, requests being processed are held on queue points and in the data store for messaging engines. You can display the data for SCA requests, and if appropriate take further action to manage the throughput of SCA requests.

When an SCA module is running in WebSphere ESB, requests normally flow through the enterprise service bus without needing to be managed. Sometimes, you might want to check the throughput of a request, check the contents of a request, or if some problem has occurred delete a request. You might also want to take other actions such as to monitor the overall throughput of requests, or change the reliability setting for requests.

Requests are handled as messages by the service integration technologies of the underlying WebSphere Application Server. For this reason, actions to manage requests are managed by using the WebSphere Application Server tasks to act on service integration messages.

This topic provides an overview of the main tasks that you might consider using, and links into the WebSphere Application Server tasks for more detailed information.

- Listing messages on a message point

SCA requests that are being processed are held on queue points of the `SCA.SYSTEM.cell_name.Bus`. You can list the SCA requests either through a queue destination for a component of the SCA module, or through the messaging engine that hosts a queue point; for example: **Service integration** → **Buses** → **SCA.SYSTEM.localhostNode01Cell.Bus** → **Destinations** → **StockQuoteService_Export** → **Queue points** → **StockQuoteService_Export@localhostNode01.server1-SCA.SYSTEM.localhostNode01Cell.Bus** → **Runtime** → **Messages**
- Resolving locked messages on a message point

If a problem occurs, an SCA request might remain locked on the queue point where it is being processed. You can display the message **State** property that indicates whether or not the request is locked, and if appropriate take further action to resolve the problem.
- Deleting messages on a message point

Under exceptional circumstances, you might need to delete one or messages that exist on a message point for a selected bus destination or messaging engine. You should not normally need to delete messages on a message point. This task is intended as part of a troubleshooting procedure.
- Viewing data in the data store for a messaging engine.

A messaging engine maintains requests as volatile (nonpersistent) and durable (persistent) data in its data store.

You can use the database tools for the data store to view request data in the data store for a messaging engine. For example, if the messaging engine uses the default Cloudscape database, you can use the CloudView tool to view request data.
- Changing message reliability for a destination

Request messages have a quality of service attribute that specifies the reliability of message delivery. You can select a reliability to suit your requirements for assured delivery, and system performance. The administrator can specify the reliability setting on bus destinations, or the reliability can be specified by individual producers (typically under application control through an API call).

Viewing data in a data store:

A messaging engine persists both volatile and durable data in its data store, including messages, transaction states, and communication channel states. You can use the database tools to view data in the data store for a messaging engine.

Before you can use the CloudView tool to view data in a Cloudscape data store for a messaging engine, you must have stopped the messaging engine.

Volatile data is lost when a messaging engine stops, in either a controlled or an uncontrolled manner. Durable data is available after the server restarts.

In some cases, you might want to view the data in a data store; for example, to examine the messages being processed by the messaging engine.

You can use the database tools for the data store to view data in the data store for a messaging engine. For example, if the messaging engine uses the default Cloudscape database, you can use the CloudView tool to view request messages.

1. Start the CloudView tool. For example, on Windows complete the following sub-steps:
 - a. Open a command window
 - b. Change directory to *install_root/cloudscape/bin/embedded*
 - c. Type *cvview*
2. Open the data store for the messaging engine. Use the CloudView tool to complete the following sub-steps:
 - a. Click **File** → **Open**
 - b. Browse to, then select, the database file.

For a messaging engine, the database is stored in the directory *install_root/profiles/profile_name/databases/com.ibm.ws.sib* and has the name of the messaging engine; for example, for the default standalone profile on Windows, the database file for the messaging engine *localhostNode01.server1-SCA.SYSTEM.localhostNode01Cell.Bus* (for server1 on the SCA.SYSTEM bus) is:

install_root/profiles/default/databases/com.ibm.ws.sib/localhostNode01.server1-SCA.SYSTEM.localhostNode01Cell.Bus

- c. Click **Open**
3. Use the CloudView controls to view data.
 - a. Expand Tables in the navigation pane.
 - b. Click a table name.
 - c. In the content pane, click the Data tab.
4. **Optional:** Display more help about using the CloudView tool.

To view help about using CloudView, click the Help button or the menubar option **Help** → **Cview help...**

Changing message reliability for a bus destination:

Messages have a quality of service attribute that specifies the reliability of message delivery. You can select a reliability to suit your requirements for assured delivery, and system performance.

The administrator can specify the reliability setting on bus destinations, or the reliability can be specified by individual producers (typically under application control through an API call). The administrator can specify whether the default reliability for the destination can be overridden by a producer, and the maximum reliability that attached producers can request.

To browse or change the message reliability setting of a destination, use the administrative console to complete the following steps:

1. In the navigation pane, click **Service integration** → **Buses**.
2. In the content pane, click the name of the bus on which the destination exists.
3. Click **Destinations**
4. Click the destination name. This displays the details page for the destination.
5. Review the reliability properties. The following properties control the message reliability for the destination:

Default reliability

The reliability assigned to a message produced to this destination when an explicit reliability has not been set by the producer.

Maximum reliability

The maximum reliability of messages accepted by this destination.

These properties can have values from the following list:

Best effort nonpersistent

Messages are discarded when a messaging engine stops or fails.

Messages may also be discarded if a connection used to send them becomes unavailable and as a result of constrained system resources.

Express nonpersistent

Messages are discarded when a messaging engine stops or fails.

Messages may also be discarded if a connection used to send them becomes unavailable.

Reliable nonpersistent

Messages are discarded when a messaging engine stops or fails.

Reliable persistent

Messages may be discarded when a messaging engine fails.

Assured persistent

Messages are not discarded.

For more information about using these properties to control message reliability, see Message reliability levels.

6. Review whether producers can override the default reliability setting.

Enable producers to override default reliability

Select this option to enable producers to override the default reliability that is set on the destination.

7. **Optional:** Change the destination properties to suit your needs.
You can further refine the configuration of a destination by setting other properties to suit your needs, as described in *Configuring bus destinations*.
8. Click **OK**.
9. Save your changes to the master configuration.

Doing more with mediation modules

Besides using the WebSphere administrative console to manage mediation modules, you can perform other tasks such as those to manage resources used by modules, to manage applications used to deploy modules, and others using commands to manage mediation modules.

The more routine tasks for managing mediation modules are described in “Managing mediation modules” on page 202.

This set of topics provides links into the WebSphere Application Server topics for information about tasks involving applications used to deploy mediation modules.

For information about doing more with mediation modules, see the following sub-topics:

Administering resources for mediation modules:

Mediation modules make use of resources provided by the service integration technologies of WebSphere Application Server. Mediation modules can also make use of a range of resources, including those provided by the Java Message Service

(JMS) and common event infrastructure. To administer the resources for mediation modules, you can use the WebSphere administrative console, commands, and scripting tools.

For more information about managing resources for mediation modules, see the related topics.

Service integration technologies

Service integration resources, such as bus destinations, enable a mediation module to use service integration technologies. Queue destinations are used by the SCA runtime exploited by the mediation module as a robust infrastructure to support asynchronous interactions between components and modules. When you install a mediation module into WebSphere ESB, the destinations used by a module are defined on a member of the service integration bus called `SCA.SYSTEM.cell_name.Bus`. These bus destinations are used to hold messages that are being processed for components of the mediation module that use asynchronous interactions:

Queue `sca/module_name`

This is the destination used to buffer asynchronous requests sent to module `module_name`

Queue `sca/module_name/export/export_name`

This is the destination used to buffer asynchronous requests routed to module export `export_name`.

Queue `sca/module_name/exportlink/export_name`

This is the destination used by the export to send asynchronous requests into the module. Requests are routed to the component target linked to the export.

Queue `sca/module_name/component/component_name`

This is the destination used to buffer asynchronous requests sent to component `component_name`

Queue `sca/module_name/component/component_name/source/source_name`

This is the destination used to buffer asynchronous requests routed to the component source import `source_name`.

Queue `sca/module_name/component/component_name/target/target_name`

This is the destination used to buffer asynchronous requests routed to the component target export `target_name`.

Queue `sca/module_name/import/import_name`

This is the destination used to buffer asynchronous requests sent to import `import_name`.

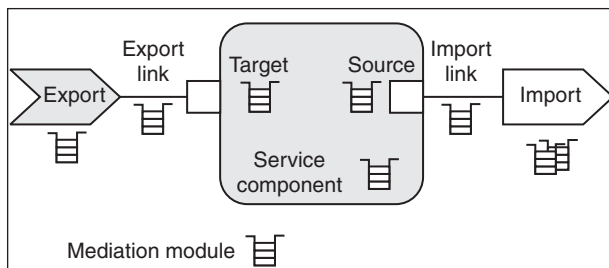
Queue `sca/module_name/importlink/import_name`

This is the destination used by the import to send asynchronous requests out of the module. Requests are routed to the module export linked to the import.

Queue `sca/module_name/import/sca/dynamic/import/scaimport` [for SCA binding]

Queue `sca/module_name/import/sca/dynamic/import/wsimport` [for Web service binding]

Queue `sca/contextStore/module_name`



For each of the destinations, a queue point is also created defined on the messaging engine of the bus member.

You can deploy and use mediation modules without needing to manage these resources. However, you might want to adjust the configuration of the resources (for example, to modify the maximum messaging quality of service used) or to use them in locating messages for troubleshooting.

Java Message Service (JMS)

JMS resources enable a mediation module to use asynchronous messaging as a method of communication based on the Java Message Service (JMS) programming interface. For mediation modules, JMS support is provided by the default messaging provider of WebSphere Application Server. To manage use of the Java Message Service, you can administer the following resources:

JMS connection factory

A JMS connection factory is used to create connections to the associated JMS provider of JMS destinations, for both point-to-point and publish/subscribe messaging. Use connection factory administrative objects to manage JMS connection factories for the default messaging provider.

JMS queue

A JMS queue is used as a destination for point-to-point messaging. Use JMS queue destination administrative objects to manage JMS queues for the default messaging provider.

JMS topic

A JMS topic is used as a destination for publish/subscribe messaging. Use topic destination administrative objects to manage JMS topics for the default messaging provider.

JMS activation specification

A JMS activation specification is associated with one or more message-driven beans and provides the configuration necessary for them to receive messages.

common event infrastructure (CEI)

CEI resources enable a mediation module to use standard formats and mechanisms for managing event data. To manage use of the common event infrastructure, you can administer the following resources:

Data Store Profile

Defines properties used by the default data store. The default data store is the data store supplied by the Common Event Infrastructure.

Emitter Factory Profile

This profile defines the options for an event emitter.

Event Bus Transmission Profile

This profile defines the EJB entry into the event bus.

Event Group Profile

This profile defines a list of events which are determined through selector expressions. JMS queues and a JMS topic can be associated with each event group. If the event server distribution service is enabled and an event matches an event group the event is distributed to any topic or queues configured for that particular event group.

Event Server Profile

This profile defines the properties for the event server.

Filter Factory Profile

This profile defines the properties of a filter. The filter uses the filter configuration string to determine whether an event will be passed to the bus.

JMS Transmission Profile

The database schema that contains the event tables.

Using asynchronous messaging:

These topics describe how to use asynchronous messaging with WebSphere ESB, to enable enterprise applications to use JMS resources and message-driven beans.

WebSphere ESB supports asynchronous messaging as a method of communication based on the Java Message Service (JMS) programming interface. This JMS support is provided by one or more *JMS providers*, and associated services and resources, that you configure for use by enterprise applications. You can deploy EJB 2.1 applications that use the JMS 1.1 interfaces and EJB 2.0 applications that use the JMS 1.0.2 interfaces. This support is provided by the WebSphere Application Server on which WebSphere ESB is built.

You can use the WebSphere administrative console to administer the WebSphere ESB support for asynchronous messaging. For example, you can configure messaging providers and their resources, and can control the activity of messaging services.

For more information about implementing WebSphere enterprise applications that use asynchronous messaging, see the following sub-topics:

For more information about implementing WebSphere enterprise applications that use asynchronous messaging, see the following sub-topics in the infocenter. Alternatively, you can see *Administering applications and their environment* and other PDF books available through the WebSphere Application Server Network Deployment library Web page at <http://www-306.ibm.com/software/webservers/appserv/was/library/>.

- Learning about messaging with WebSphere Application Server
Use this topic to learn about the use of asynchronous messaging for enterprise applications with WebSphere ESB.
- Installing and configuring a JMS provider
This topic describes the different ways that you can use JMS providers with WebSphere ESB. A JMS provider enables use of the Java Message Service (JMS) and other message resources in WebSphere ESB.

- Using the default messaging provider
This topic is the entry-point into a set of topics about enabling WebSphere applications to use messaging resources provided by the default messaging provider. The default messaging provider is installed and runs as part of WebSphere Application Server, and is based on service integration technologies.
- Maintaining Version 5 default messaging resources
This topic is the entry-point into a set of topics about maintaining messaging resources provided for WebSphere Application Server, version 5 applications by the default messaging provider.
- Using JMS resources of WebSphere MQ
This topic is the entry-point into a set of topics about enabling WebSphere applications to use JMS resources provided by WebSphere MQ.
- Using JMS resources of a generic provider
This topic is the entry-point into a set of topics about enabling WebSphere applications to use JMS resources provided by a generic messaging provider (other than a WebSphere default messaging provider or WebSphere MQ).
- Administering support for message-driven beans
Use these tasks to manage resources used to support message-driven beans. These tasks are in addition to the tasks for administering resource adapters, JMS providers and the resources they provide.
- Troubleshooting WebSphere messaging
Use this overview task to help resolve a problem that you think is related to the WebSphere Messaging. To identify and resolve problems that you think are related to WebSphere Messaging, you can use the standard WebSphere ESB troubleshooting facilities.

Using commands to manage service applications:

You can manage service applications using commands. The commands can be used within scripts.

You must use the wsadmin tool to run service application commands.

You can use the wsadmin tool in different ways. You can use the tool interactively, as an individual command or in a script. Running multiple commands in a script can be useful if you are administering multiple machines.

WebSphere ESB has commands that let you display SCA modules, and their imports and exports. You can also change import bindings if they are SCA bindings, but not if they are other types of bindings.

1. List the SCA administration commands. `$AdminTask help SCAAdminCommands`
2. Display detailed help about a given command. `$AdminTask help command_name`

```
$AdminTask help listSCAModules
```

Managing mediation modules:

You can list the mediation modules that have been deployed to WebSphere ESB from the command line. You can also view information associated with individual mediation modules and make changes to some import bindings.

You must use the wsadmin tool to run WebSphere ESB commands.

You can run commands individually or in a script. Running multiple commands in a script is useful if you are administering multiple machines, or producing regular reports.

Listing mediation modules:

You can use a command to list the mediation modules that have been deployed to WebSphere ESB.

You must use the wsadmin tool to run the following command.

To list the mediation modules that you have deployed, use the wsadmin tool to complete the following step.

Mediation modules are types of Service Component Architecture (SCA) modules. Therefore, to list the mediation modules you have deployed, you list the SCA modules.

List the deployed SCA modules. `$AdminTask listSCAModules`

Lists the SCA modules that have been deployed to WebSphere ESB, and the applications they are associated with. The output is returned in the format: *moduleName:application name*. This makes it easier for scripts to parse the output and extract names, for use in subsequent commands.

Displaying details of a mediation module:

You can use a command to display attributes of mediation modules.

You must use the wsadmin tool to run the following commands.

To display the description of a mediation module, use the wsadmin tool to complete the following steps.

In order to show the description of a particular mediation module, you need to know the mediation module name.

Note: Mediation modules are types of Service Component Architecture (SCA) modules.

The listSCAModules command lists SCA modules that have been deployed to WebSphere ESB.

1. List the deployed SCA modules. `$AdminTask listSCAModules`
2. Display the details of a particular SCA module. `$AdminTask showSCAModule {-moduleName moduleName}`

Note: In addition to specifying the *moduleName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

Displays the name and description of a particular SCA module.

```
$AdminTask showSCAModule {-moduleName myModule -applicationName myApplication}
```

Listing imports:

You can use a command to list the imports of any mediation module that has been deployed to WebSphere ESB.

You must use the wsadmin tool to run the following commands.

To list the imports of a mediation module, use the wsadmin tool to complete the following steps.

In order to list the imports of a mediation module you need to know the name of the mediation module.

Note: Mediation modules are types of Service Component Architecture (SCA) modules.

The listSCAModules command lists all SCA modules that have been deployed to WebSphere ESB, and the listSCAImports command lists the imports for a particular SCA module. It is possible for an SCA module not to have any imports.

1. List the deployed SCA modules. `$AdminTask listSCAModules`
2. List the imports of a particular SCA module. `$AdminTask listSCAImports {-moduleName moduleName}`

Note: In addition to specifying the *moduleName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

Lists the imports for a particular SCA module.

```
$AdminTask listSCAImports {-moduleName myModule -applicationName myApplication}
```

Displaying details of an import:

You can use a command to display import details of a mediation module that has been deployed to WebSphere ESB.

You must use the wsadmin tool to run the following commands.

To display the import details of mediation module, use the wsadmin tool to complete the following steps.

In order to show the details of a particular mediation module import, you need to know the mediation module name and the import name.

Note: Mediation modules are types of Service Component Architecture (SCA) modules.

The listSCAModules command lists all SCA modules that have been deployed to WebSphere ESB, and the listSCAImports command lists the imports for a particular SCA module. It is possible for an SCA module not to have any imports.

1. List the deployed SCA modules. `$AdminTask listSCAModules`
2. List the imports for a particular SCA module. `$AdminTask listSCAImports {-moduleName moduleName}`

Note: In addition to specifying the *moduleName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

3. Show the details of a particular SCA module import. `$AdminTask showSCAImport {-moduleName moduleName -import importName}`

Note: In addition to specifying the *moduleName* and *importName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

Displays the import details for a particular SCA module import.

```
$AdminTask showSCAImport {-moduleName myModule -applicationName  
myApplication -import myImport}
```

Displaying an import binding:

You can use a command to display the import bindings of a mediation module that has been deployed to WebSphere ESB.

You must use the wsadmin tool to run the following commands.

To display the import bindings of a particular mediation module, use the wsadmin tool to complete the following steps.

In order to show the import bindings of a particular mediation module import, you need to know the mediation module name and the import name.

Note: Mediation modules are types of Service Component Architecture (SCA) modules.

The `listSCAModules` command lists all SCA modules that have been deployed to WebSphere ESB, and the `listSCAImports` command lists the imports for a particular SCA module. It is possible for an SCA module not to have any imports.

1. List the deployed SCA modules. `$AdminTask listSCAModules`
2. List the imports for a particular SCA module. `$AdminTask listSCAImports {-moduleName moduleName}`

Note: In addition to specifying the *moduleName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

3. Show the import binding for a particular import. `$AdminTask showSCAImportBinding {-moduleName moduleName -import importName}`

Note: In addition to specifying the *moduleName* and *importName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

Displays the import binding for a particular SCA module import.

```
$AdminTask showSCAImportBinding {-moduleName myModule -applicationName  
myApplication -import myImport}
```

Changing an import binding:

You can use a command to change the SCA import bindings of mediation modules that have been deployed to WebSphere ESB.

You must use the wsadmin tool to run the following commands.

You might change an SCA import binding if you wanted a particular mediation module to invoke a different mediation module. If you change an import binding you must ensure that the import and export match, that is, that the operations provided are equivalent. This might involve reviewing the WSDL. Use the wsadmin tool to complete the following steps.

Note: You can only modify Service Component Architecture (SCA) module import bindings if they are SCA bindings, but not if they are other types of bindings. An SCA binding connects one SCA module to another SCA module. Mediation modules are types of SCA module.

To modify the binding of a particular mediation module import, you need to know the names of the source and target mediation modules, and the specific import and export.

The listSCAModules command lists all SCA modules that have been deployed to WebSphere ESB. The listSCAImports command lists all imports for a particular SCA module and the listSCAExports command lists all exports for a particular SCA module.

1. List the deployed SCA modules. `$AdminTask listSCAModules`
2. List the imports for a particular SCA module. `$AdminTask listSCAImports {-moduleName moduleName}`

Note: In addition to specifying the *moduleName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

3. Modify an SCA module import binding, of type SCA. `$AdminTask modifySCAImportSCABinding -moduleName moduleName -import importName -targetModule targetModuleName -targetExport targetExportName`

Note: You also have the option of specifying the *applicationName* and *targetApplicationName*. Providing an *applicationName* and a *targetApplicationName* improves performance.

Changes the SCA import binding for a particular SCA module import.

WebSphere ESB issues a warning for each import interface that is not satisfied by an export interface. WebSphere ESB compares the WSDL port type names of import and export, if they are not the same then a warning is issued. However, if the port type names do match, then WebSphere ESB assumes that the operations provided are equivalent and no warning is issued.

```
$AdminTask modifySCAImportSCABinding {-moduleName myModule -applicationName myApplication -import myImport -targetModule myTargetModule -targetApplicationName myTargetApplication -targetExport myTargetExport}
```

Listing Exports:

You can use a command to list the exports of any mediation module that has been deployed to WebSphere ESB.

You must use the wsadmin tool to run the following commands.

To list the exports of a mediation module, use the wsadmin tool to complete the following steps.

In order to list the exports of a particular mediation module, you need to know the mediation module name.

Note: Mediation modules are types of Service Component Architecture (SCA) modules.

The listSCAModules command lists all SCA modules that have been deployed to WebSphere ESB, and the listSCAExports command lists the exports for a particular SCA module. It is possible for an SCA module not to have any exports.

1. List the deployed SCA modules. `$AdminTask listSCAModules`
2. List the exports of a particular SCA module. `$AdminTask listSCAExports {-moduleName moduleName}`

Note: In addition to specifying the *moduleName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

Lists the exports for a particular SCA module.

```
$AdminTask listSCAExports {-moduleName myModule -applicationName myApplication}
```

Displaying details of an export:

You can use a command to display export details of a mediation module that has been deployed to WebSphere ESB.

You must use the wsadmin tool to run the following commands.

To display the export details of mediation module, use the wsadmin tool to complete the following steps.

In order to show the details of a particular mediation module export, you need to know the mediation module name and the export name.

Note: Mediation modules are types of Service Component Architecture (SCA) modules.

The listSCAModules command lists all SCA modules that have been deployed to WebSphere ESB, and the listSCAExports command lists the exports for a particular SCA module. It is possible for an SCA module not to have any exports.

1. List the deployed SCA modules. `$AdminTask listSCAModules`
2. List the exports for a particular SCA module. `$AdminTask listSCAExports {-moduleName moduleName}`

Note: In addition to specifying the *moduleName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

3. Show the details of a particular SCA module export. `showSCAExport -moduleName moduleName -export exportName`

Note: In addition to specifying the *moduleName* and *exportName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

Displays the export details for a particular SCA module export.

```
$AdminTask showSCAExport {-moduleName myModule -applicationName  
myApplication -export myExport}
```

Displaying an export binding:

You can use a command to display the export bindings of a mediation module that has been deployed to WebSphere ESB.

You must use the wsadmin tool to run the following commands.

To display the export bindings of a particular mediation module, use the wsadmin tool to complete the following steps.

In order to show the export bindings of a particular mediation module export, you need to know the mediation module name and the export name.

Note: Mediation modules are types of Service Component Architecture (SCA) modules.

The listSCAModules command lists all SCA modules that have been deployed to WebSphere ESB, and the listSCAExports command lists the exports for a particular SCA module.

1. List the deployed SCA modules. `$AdminTask listSCAModules`
2. List the exports for a particular SCA module. `$AdminTask listSCAExports {-moduleName moduleName}`

Note: In addition to specifying the *moduleName*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

3. Show the export binding for a particular export. `$AdminTask showSCAExportBinding {-moduleName moduleName -export exportName}`

Note: In addition to specifying the *moduleName* and *export*, you have the option of specifying the *applicationName*. Providing an *applicationName* improves performance.

Shows the export binding for a particular SCA module export. The information displayed depends upon the type of binding. If an export has no binding specified then the runtime assumes that the binding is of type SCA.

```
$AdminTask showSCAExportBinding {-moduleName myModule -applicationName  
myApplication -export myExport}
```

Administering enterprise applications:

Use the console's Enterprise Application page (viewed by clicking **Applications > Enterprise Applications**) to view and administer enterprise applications installed on the server.

To view the values specified for an application's configuration, click the application name from the list. The application details page opens and displays the application's configuration properties and, if appropriate, local topology. From this page, you can modify existing values and link to additional console pages for configuring the application.

To administer an enterprise application, select it by clicking the check box next to its name and then use one of the following buttons:

Table 11. Buttons for administering enterprise applications

Button	Resulting action
Start	Attempts to run the application. After the application starts successfully, the state of the application changes to one of the following: <ul style="list-style-type: none"> • Started—The application has started on all deployment targets • Partial Start—The application is still starting on one or more of the deployment targets
Stop	Attempts to stop the processing of the application. After the application stops successfully, the state of the application changes to one of the following: <ul style="list-style-type: none"> • Stopped—The application has stopped on all deployment targets • Partial Stop—The application is still stopping on one or more of the deployment targets
Install	Opens a wizard to help you deploy an enterprise application or module (such as a .jar, .war, or .ear file) onto a server.
Uninstall	Deletes the application from the WebSphere Application Server configuration repository and deletes the application binaries from the file system of all nodes where the application modules are installed after the configuration is saved.
Update	Opens a wizard to help you update application files deployed on a server. You can update the full application, a single module, a single file, or part of the application. If a new file or module has the same name as a file or module already on the server, the new file or module replaces the existing one. Otherwise, it is added to the deployed application.
Remove File	Deletes a file from the deployed application or module. This button deletes the file from the configuration repository and from the file system of all nodes where the file is installed.
Export	Opens the Export Application EAR files page so you can export an enterprise application to an EAR file. Use the Export action to back up a deployed application and to preserve its binding information.
Export DDL	Opens the Export Application DDL files page so you can export DDL files in the EJB modules of an enterprise application.

For more information on administering applications, see the WebSphere Application Server for z/OS Information Center.

Administering relationships

The relationship manager is a tool for manually manipulating relationship data to correct errors found in automated relationship management or provide more complete relationship information. In particular, it provides a facility for retrieving as well as modifying relationship instance data.

The relationship manager allows you to configure, query, view, and perform operations on relationship runtime data, including participants and their data. You create relationship definitions with the relationship editor. At run time, instances of the relationships are populated with the data that associates information from different applications. This relationship instance data is created when the maps or other WebSphere Process Server components run and need a relationship instance. The relationship service exposes a set of application programming interfaces (API's) to retrieve relationship metadata and to create, retrieve, and manipulate the instance data. The data is stored in the relationship tables specified in the relationship definition. The relationship manager provides a graphical user interface to interact with the relationships and relationship instances.

For each relationship instance, the relationship manager can display a hierarchical listing of its participants. Each participant fills a role in the relationship and has instance data, properties, and key attributes. The relationship tree also provides detailed information about each of the participants in the relationship instance, such as the type of entity, its value, and the date it was last modified. A relationship instance ID is automatically generated when the relationship instance is saved in the relationship table. The relationship manager will display this instance ID at the top level of the relationship tree.

You can use the relationship manager to manage entities at all levels: the relationship instance, participant instance, and attribute data and property data levels. For example, you can use the relationship manager to:

- Create and delete relationship instances
- Modify the contents of a relationship instance, such as adding and deleting participants
- Add a participant's data and copy and paste the data of a participant from another relationship into the relationship instance, thus creating a new participant (as long as the participant types are identical)
- Activate and deactivate participants
- Retrieve participants based on instance ID's or data
- Salvage a situation when problems arise. For example, when corrupt or inconsistent data from a source application has been sent to the generic and destination application relationship table, you can use the relationship manager to rollback the data to a point in time when you know the data is reliable.

For more information on relationships, see the WebSphere Integration Developer Information Center and the relationship services and administrative console in the WebSphere Process Server Information Center.

Viewing relationship types

Perform this task to view information related to the relationship type, including the relationship name, display name, static or identity attributes, and roles.

Security role required: To perform this task, you must be logged in as a monitor, an operator, or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.

Viewing relationship details

Perform this task to view detailed information for the relationship type, including the relationship name, display name, property values, role type attributes, and static and identity attributes.

Security role required: To perform this task, you must be logged in as a monitor, an operator, or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. You can view the relationship details in two ways:
 - a. Click the relationship name.
 - b. In the **Select** column, click the radio button next to the relationship name; and click **Details**.
5. To go back to the list of relationship details, click **Relationships** from the path at the top of the page.

Querying relationship instances

Use this task to perform relationship-based instance queries.

Security role required: To perform this task, you must be logged in as a monitor, an operator, or an administrator.

Select a query parameter (**All**, **By ID**, or **By property**) to retrieve all or a subset of the instance data for a relationship. The return is the result set of that query and is displayed based on how you set the **Preferences** fields on the relationship instances page.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. In the **Select** column, click the radio button next to the relationship name; and click **Query**.
5. Choose one of the query options by selecting the appropriate tab.

Option	Description
All tab	Retrieve a list of all instances in the relationship. You can select to display all activated, all inactivated, or all activated and inactivated relationship instance data.

Option	Description
By ID tab	Retrieve relationship instances in the range of the starting and ending instance identifiers. If one field is left blank, it will return only the single instance. The query will return all of the roles for the instances it finds.
By property tab	Retrieve relationship instances by specific property values.

6. Once you have selected the query parameter, you then have the following options.
 - Click **Apply** to display the result data from the query.
 - Click **OK** to display the result data from the query.
 - Click **Reset** to clear your selections or return the entry to the most recent set of changes made.
 - Click **Cancel** to discard any changes made and return to the list of relationship types.

Viewing relationship instance details

Perform this task to view detailed information for the selected relationship instance, including the relationship name, relationship instance ID, role-based information, and property values.

Security role required: To perform this task, you must be logged in as a monitor, an operator, or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. In the **Select** column, click the radio button next to the relationship name; and click **Query**.
5. Select one of the three query options.
 - Select the **All** tab to retrieve a list of all instances in the relationship. You can select to display all activated, all inactivated, or all activated and inactivated relationship instance data.
 - Select the **By ID** tab to retrieve relationship instances in the range of the starting and ending instance identifiers. If one field is left blank, it will return only the single instance. The query will return all of the roles for the instances it finds.
 - Select the **By property** tab to retrieve relationship instances by specific property values.
6. Once you have selected the query option and filled in the necessary information, click either **Apply** or **OK**.
7. You can view the relationship instance details in two ways:
 - Click the relationship instance ID.
 - In the **Select** column, click the radio button next to the relationship instance ID; and click **Details**.

Editing relationship instance details

Perform this task to edit the information for the selected relationship instance.

Security role required: To perform this task, you must be logged in as an operator or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. In the **Select** column, click the radio button next to the relationship name; and click **Query**.
5. Select one of the three query options, fill in the query information, and click either **Apply** or **OK**.
6. You can view the relationship instance details in two ways:
 - Click the relationship instance ID.
 - In the **Select** column, click the radio button next to the relationship instance ID; and click **Details**.
7. Modify the property values, as necessary.
8. When you are finished making changes, you have the following options.
 - Click **Apply** to save the changes locally.
 - Click **OK** to save the changes locally.
 - Click **Reset** to clear your changes or return the entry to the most recent set of changes made.
 - Click **Cancel** to discard any changes made and return to the list of relationship instances.
9. Click **Apply Changes** to save any changes made to this point in the database.

Creating new relationship instances

Perform this task to create a new relationship instance.

Security role required: To perform this task, you must be logged in as an operator or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. In the **Select** column, click the radio button next to the relationship name; and click **Query**.
5. Select one of the three query options.
6. Once you have selected a query option and filled in the necessary information, click either **Apply** or **OK**.
7. Click **Create**.
8. Add the value information for the property values if you want values other than the default values, and click either **Apply** or **OK** to save the new relationship instance locally.

Note: You must also create a role instance for the relationship instance, as you cannot have a relationship instance without a role instance.

9. Once you have created both a relationship instance and a role instance, click **Apply Changes** to save any changes made to this point in the database.

Deleting relationship instances

Perform this task to delete a selected relationship instance.

Security role required: To perform this task, you must be logged in as an operator or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. In the **Select** column, click the radio button next to the relationship name; and click **Query**.
5. Select one of the three query options, fill in the necessary information, and click either **Apply** or **OK**.
6. Click the radio button next to the relationship instance ID you want to delete.
7. Click **Delete** to delete the relationship instance locally.
8. Click **Apply Changes** to save any changes made to this point in the database.

Rolling back relationship instance data

Perform this task to rollback the instance data for a relationship to a specified date and time.

Security role required: To perform this task, you must be logged in as an operator or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. In the **Select** column, click the radio button next to the relationship name; and click **Rollback**.
5. Select the time period for the rollback by filling in the **From date** and **To date** fields.
6. Click **Apply** or **OK**.

Viewing role types associated with a relationship

Perform this task to view descriptions of the roles for a relationship, including the role name, display name, key (for the business object), role object type (business object name), and managed attribute setting.

Security role required: To perform this task, you must be logged in as a monitor, an operator, or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.

3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. Click the role names in the **Roles** column of the relationship.

Note: To set display preferences for this collection page, click **Preferences**. Modify the field values, as desired, and click **Apply**.

Querying relationship instances based on roles

Use this task to perform different role-based instance queries.

Security role required: To perform this task, you must be logged in as a monitor, an operator, or an administrator.

Select a query parameter (**By key**, **By date**, **By property**, or **Advanced**) to retrieve all or a subset of the instance data for a relationship based on the chosen role. The return is the result set of that query and is displayed based on how you set the **Preferences** fields on the relationship instances page.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. Click the role names in the **Roles** column of the relationship.
5. In the **Select** column, click the radio button next to the role name; and click **Query**.
6. Choose one of the query options by selecting the appropriate tab.

Option	Description
By key tab	Retrieve relationship instances based on values for key attributes of selected participants. To retrieve this information, fill in the value of the ID and click Apply or OK .
By date tab	Retrieve relationship instances with roles that were created or modified between the specified dates. To retrieve this information, fill in the From date and To date fields with the desired dates, and click OK or Apply .
By property tab	Retrieve relationship instances by specific role property values. To retrieve this information, perform these steps: <ol style="list-style-type: none"> 1. Select the role property name and type parameters from the drop-down list. 2. Enter the value parameter for the specified role property in the Property value field, and click OK or Apply.

Option	Description
Advanced tab	Request a more refined search by combining fields from the other query panels. Fill in the information as described in the other options, and click OK or Apply .

Note: You also have the following options on each tab:

- Click **Reset** to clear your selections or return the entry to the most recent set of changes made.
- Click **Cancel** to discard any changes made and return to the list of roles.

Viewing role details

Perform this task to view detailed information for the role, including the relationship name, role name, display name, property values, keys, role object type, and managed attribute setting.

Security role required: To perform this task, you must be logged in as a monitor, an operator, or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. Click the role names in the **Roles** column of the relationship.
5. You can view the role details in two ways:
 - Click the role name.
 - In the **Select** column, click the radio button next to the role name; and click **Details**.

Viewing role instance details

Perform this task to view detailed information for the selected role instance, including the role name, role element, key attributes, property values, status, and logical state.

Security role required: To perform this task, you must be logged in as a monitor, an operator, or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. In the **Select** column, click the radio button next to the relationship name; and click **Query**.
5. Select one of the query options, and click **Apply** or **OK**.
6. Click the role names in the **Role instances** column of the relationship.
7. You can view the role instances in two ways:
 - Click the role name.

- In the **Select** column, click the radio button next to the role name; and click **Instances**.
8. You can view the instance details in two ways:
 - Click the role instance name.
 - In the **Select** column, click the radio button next to the role instance; and click **Details**.

Creating new role instances

Perform this task to create a new role instance for a relationship.

Security role required: To perform this task, you must be logged in as an operator or an administrator.

Note: Additional role instances cannot be added to identity relationship instances.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. In the **Select** column, click the radio button next to the relationship name; and click **Query**.
5. Select one of the query options, fill in the necessary information, and click **Apply** or **OK**.
6. Click the role names in the **Role instances** column of the relationship.
7. Click the radio button next to the role name, and click **Create**.
8. Add the value for the ID and the value information for the property values, and click either **Apply** or **OK** to save the new role instance locally.

You can only set the key value information when creating the role instance. You cannot change this information after you have applied the changes back to the database. However, you can edit the property values later.

Note: You also have the following options:

- Click **Reset** to clear your selections or return the entry to the most recent set of changes made.
 - Click **Cancel** to discard any changes made and return to the list of role instances.
9. Click **Apply Changes** to save any changes made to this point in the database.

Deleting role instances

Perform this task to delete a selected role instance of a relationship.

Security role required: To perform this task, you must be logged in as an operator or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. In the **Select** column, click the radio button next to the relationship name; and click **Query**.

5. Select one of the query options, fill in the necessary information, and click **Apply** or **OK**.
6. Click the role names in the **Role instances** column of the relationship.
7. In the **Select** column, click the radio button next to the role name; and click **Instances**.
8. In the **Select** column, click the radio button next to the role instance; and click **Delete** to delete the role instance locally.
9. Click **Apply Changes** to save any changes made to this point in the database.

Editing role instance properties

Perform this task to edit the information for the selected role instance.

Security role required: To perform this task, you must be logged in as an operator or an administrator.

1. Ensure that the administrative console is running.
2. In the navigation pane, click **Integration Applications > Relationship Manager**.
3. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
4. In the **Select** column, click the radio button next to the relationship name; and click **Query**.
5. Select one of the query options, fill in the necessary information, and click **Apply** or **OK**.
6. Click the role names in the **Role instances** column of the relationship.
7. In the **Select** column, click the radio button next to the role name; and click **Instances**.
8. In the **Select** column, click the radio button next to the role instance; and click **Details**.
9. Edit the information as necessary, and click **Apply** or **OK** to save these changes locally.

Note: You also have the following options:

- Click **Reset** to clear your changes or return the entry to the most recent set of changes made.
 - Click **Cancel** to discard any changes made and return to the list of role instances.
10. Click **Apply Changes** to save any changes made to this point in the database.

Tutorial: Relationship manager administration

This tutorial demonstrates the basic functions of the WebSphere Process Server relationship manager. Relationships are used to correlate identifiers from different environments for the same item of data. For example, in one environment, states are identified by two-letter abbreviations (AZ, TX). In another environment, different abbreviations are used (Ariz., Tex.). A relationship would be created to correlate "AZ" in the first environment to "Ariz" in the second environment.

The sample relationship referenced here correlates customer ID's. Many business applications maintain databases of customers, and most of these applications assign their own ID to each customer. In an enterprise environment, the same customer likely has a different ID in each business application. In this tutorial, a relationship is defined to correlate customer ID's. The relationship name is

"SampleCustID". Two roles are defined for this relationship. One role is for the Customer Information System (CIS), and the other role is for the General Ledger (GL) application. This relationship was created by the relationship services sample along with the roles and a small amount of sample data.

The relationship manager is designed to add, modify, and remove role instances of a relationship instance as well as add, modify, and remove relationship instances. WebSphere Integration Developer should be used to create and deploy new relationship definitions. The definitions are stored as XML files that are deployed as part of a J2EE application to a particular server.

Objectives of this tutorial

After completing this tutorial, you will be able to change the values of relationship instances.

Time required to complete this tutorial

This tutorial requires approximately 10 minutes to complete.

Prerequisites

This tutorial uses a relationship that is created by the relationship services technical sample. Before following the steps of this tutorial, go to the samples gallery and perform the steps described in the relationship services sample to create the required relationship and roles.

Changing the values of a relationship instance

One of your customers has a customer ID of A004 in your CIS application. This same customer has a customer ID of 801 in your GL application. However, due to a data entry error, the relationship instance that correlates the customer ID's of this customer currently has a value of 901 instead of 801 for the GL customer ID. This tutorial takes you through the steps to correct this entry in the relationship.

1. Open the WebSphere Process Server administrative console.
2. If security is enabled, log in as a user with administrator privileges.
3. In the navigation pane, click **Integration Applications > Relationship Manager**.
4. Open the relationships page for the server you want to manage by clicking **Relationships** next to that relationship services MBean.
A relationship named SampleCustID should be visible.
5. Locate the **Roles** column, and click the link in this column for the SampleCustID relationship.
Three roles should be available: MyCISCustomer_0, MyGLCustomer_0, and GenericCustomer_0.
6. Search for a particular value on the MyGLCustomer_0 role to locate the relationship that must be modified. Select the radio button next to MyGLCustomer_0, and click **Query**.
7. Under the **By Key** tab, enter the value 901 in the **Value** field under Key attributes; and click **Apply**.

This locates the relationship instance for the requested customer.

8. Click the relationship instance ID.

This page shows the actual relationship data for customer ID 901 in the GL application. Since a query was done against a particular role, only data for that role was returned.

9. Click the MyGLCustomer_0 link in the **Name** column.
This displays all of the roles that match the query for this relationship instance.

Note: The role should have customerNumber=901 and no property values. If any other data appears, you need to look at the role instance and record any data you want to keep.
10. In the **Select** column, click the radio button next to the role instance; and click **Delete**.
11. Click **Create** to create a new role instance for this relationship instance.
12. Enter 801 as the **Value** for customerNumber, and click **OK**.
You should see a new unsaved instance in the table.
13. Click **Apply Changes** to save all the updates into the relationship database tables and return the changed instance.

You now have the correct customer ID value in the relationship instance for the GL application.

Managing WebSphere Process Server failed events

The administrator can managed WebSphere Process Server failed events using the failed event manager available in the administrative console.

What is a failed event?

In the context of WebSphere Process Server, an event is a request that is received by a WebSphere Process Server application. It can come from an external source (such as an inbound application adapter) or an external invocation to a web service. The event is comprised of a reference to the business logic it wants to operate and its data, stored in a Service Data Object (a business object). When an event is received, it is processed by the appropriate WebSphere Process Server application business logic.

A single thread of execution can branch off into multiple branches (or threads); the individual branches are linked to the main invoking event by the same session context.

If this business logic in one of these branches cannot execute completely due to system failure, component failure, or component unavailability, the event moves into the failed state. If multiple branches fail, a failed event is created for each. The WebSphere Process Server Recovery subsystem handles the following types of failed events:

- Event failures that occur during an asynchronous invocation of a Service Component Architecture (SCA) operation
- Event failures that are caused by a runtime exception (in other words, any exception that is not declared in the methods used by the business logic)

The Recovery subsystem collects these types of failed events and makes them available for administrative purposes through the failed event manager interface.

Failed events typically have source and destination information associated with them. The source and destination are based on the failure point (the location where the invocation fails), regardless of the type of interaction. Consider the following example, where Component A is asynchronously invoking Component B. The request message is sent from A to B, and the response message is sent from B to A.

- If the exception occurs during the initial request, Component A is the source and Component B is the destination for the purposes of the failed event manager.
- If the exception occurs during the response, Component B is the source and Component A is the destination for the purposes of the failed event manager.

This is true for all asynchronous invocations. (The failed event manager does not handle failures from synchronous invocations.)

How are failed events managed?

An administrator uses the failed event manager available in the administrative console to browse and manage all WebSphere Process Server failed events. Failed events can be resubmitted or deleted from the system.

Common tasks for managing failed events include:

- Browsing all failed events
- Searching for failed events by specific criteria
- Editing data for a failed event
- Resubmitting failed events
- Deleting failed events

To access the failed event manager, click **Integration Applications > Failed Event Manager**.

Role-based access for failed event manager

The failed event manager uses role-based access control to the failed event data and tasks. Only the administrator and operator roles are authorized to perform tasks within the failed event manager. Users logged in as either administrator or operator can view all data associated with failed events and can perform all tasks.

Note: This security infrastructure is inherited from the base WebSphere Application Server product. For more information about security, see the information centers for WebSphere Application Server and WebSphere Process Server.

Finding failed events

Before you can edit, resubmit, or delete failed events, you must identify them. Use the search functionality in the failed event manager to find all failed events on the server, or to find a specific subset of failed events.

This topic provides instructions for finding all failed events on the server, with references to topics for conducting other searches based on source, destination, date, business object type, exception text, or a combination of those criteria.

Security role required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running.

2. Click **Integration Applications > Failed Event Manager** to enter the failed event manager.
3. From the **Failed events on this server** box, click **Get all failed events**.
The Search Results page opens, displaying a list of all the WebSphere Process Server failed events on the server.

Searching for failed events by destination

Use the Search page's **By Destination** tab to find only those failed events that are associated with a specific destination module, component, or method. The failed event manager determines the destination based on the point of failure, regardless of the type of interaction.

When performing a search, note the following:

- The values for the fields are case sensitive.
- The fields accept the asterisk (*) wildcard character.
- If you leave any field on this tab blank, the blank field is treated as a wild card. The failed event manager will search in all components, modules, or methods.
- You can search on a single destination criteria or on multiple criteria. Searching on two or more of the destination criteria provides a more refined list of failed events.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Search by destination**.
The Search page opens with the **By Destination** tab selected.
3. Specify the search criteria you want to use. You can use any combination of the following fields to customize your search:
 - The **Destination module** field—Use this field to specify the failed event's destination module.
 - The **Destination component** field—Use this field to specify the failed event's destination component.
 - The **Destination method** field—Use this field to specify the failed event's destination method.
4. Click **OK** to begin the search.
The Search Results page opens and displays a list of all failed events that were destined for the specified module, component, or method.

Related concepts

"Managing WebSphere Process Server failed events" on page 232

The administrator can managed WebSphere Process Server failed events using the failed event manager available in the administrative console.

Searching for failed events by source

Use the Search page's **By Source** tab to find only those failed events that originated from a specific source module, component, or both. The failed event manager determines the source based on the point of failure, regardless of the type of interaction.

When performing a search, note the following:

- The values for the fields are case sensitive.
- The fields accept the asterisk (*) wildcard character.
- If you leave either field on this tab blank, the blank field is treated as a wildcard. The failed event manager will search in all components or modules.
- To get the most refined list of failed events, use both the **Source module** and **Source component** fields.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Search by source**.
The Search page opens with the **By Source** tab selected.
3. Specify the search criteria. You can use one or both of the following fields:
 - The **Source module** field—Use this field to specify the module that the failed event originated from.
 - The **Source component** field—Use this field to specify the component that the failed event originated from.
4. Click **OK** to begin the search.
The Search Results page opens and displays a list of all failed events that originated from the specified module, component, or both.

Related concepts

“Managing WebSphere Process Server failed events” on page 232

The administrator can managed WebSphere Process Server failed events using the failed event manager available in the administrative console.

Searching for failed events by date

Use the Search page’s **By Date** tab to find only those events that failed during a specific time period.

When performing a search, note the following:

- The format for the date and time are locale-specific. An example of the appropriate format is provided with each field.

Note: The values you supply must match the required format exactly. If you provide an incorrectly formatted value, the failed event manager displays a warning and substitutes the default value for that field.

- The time is always local to the server. It is not updated to reflect the local time of individual machines running the administrative console.
- You must specify a value for both fields on this tab.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Search by date**.
3. Use the **From Date** field to specify the starting date and time. Because the required format for the value varies by locale, the failed event manager provides a locale-appropriate example above this field. Ensure the value you enter is formatted in the same manner as the example provided. (For instance,

the required format for the en_US locale is *MM/DD/YY HH:MM Meridiem*; therefore, a correctly formatted value for this field looks like 11/10/05 4:30 PM.
)

4. Use the **To Date** field to specify the ending date and time. Because the required format for the value varies by locale, the failed event manager provides a locale-appropriate example above this field. Ensure the value you enter is formatted in the same manner as the example provided. (For instance, the required format for the en_US locale is *MM/DD/YY HH:MM Meridiem*; therefore, a correctly formatted value for this field looks like 11/17/05 4:30 PM.
)
5. Click **OK** to begin the search.
The Search Results page opens and displays a list of all failed events that originated during the specified time period.

Searching for failed events by business object type

Use the Search page's **By Type** tab to find only those failed events that are associated with a specific business object.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Search by business object type**.

The Search page opens with the **By Type** tab selected.

3. Specify the business object type you want to search against by using one of the following:
 - The **Select the business object type** menu—Use this drop-down menu to select the type of business object associated with the failed events. This menu contains a list of all business object types found in the failed events on the server.
 - The **Other business object type** field—Use this field to specify the type of business object associated with the failed events. The field accepts the asterisk (*) wildcard character. All values are case sensitive.
4. Click **OK** to begin the search.
The Search Results page opens and displays a list of all failed events that are associated with the specified business object type.

Searching for failed events by exception

Use the Search page's **By Exception** tab to find only those failed events that are associated with a specific exception. You can specify part or all of the exception text.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Search by exception text**.
The Search page opens with the **By exception** tab selected.
3. In the **Exception text** field, type the text associated with the exception you want to search against.

You can specify all or part of the exception text, as well as the asterisk (*) wildcard character to make the search easier. The values in this field are case sensitive.

Note: If you leave the **Exception text** field blank, it is treated as a wild card; all failed events are returned.

4. Click **OK** to begin the search.

The Search Results page opens and displays a list of all failed events that are associated with the specified exception text.

Performing an advanced search for failed events

Use the Search page's **Advanced** tab to perform a more refined search for failed events by using a combination of the criteria found on the other search tabs: source, destination, date, business object type, and exception text.

Note the following:

- Unless otherwise noted below, all fields accept the asterisk (*) wildcard character.
- Leaving a field blank causes it to be treated as a wild card.

The advanced search is not optimized; executing an advanced search on a large set of failed events can reduce performance.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Advanced search**.
The Search page opens with the **Advanced** tab selected.
3. Specify the search criteria you want to use. You can use any combination of the following fields to customize your search:
 - The **Destination module** field—Use this field to specify the failed event's destination module.
 - The **Destination component** field—Use this field to specify the failed event's destination component.
 - The **Destination method** field—Use this field to specify the failed event's destination method.
 - The **Source module** field—Use this field to specify the module that the failed event originated from.
 - The **Source component** field—Use this field to specify the component that the failed event originated from.
 - The **From Date** field—Use this field to specify the starting date and time if you want to search within a specific time period. This field does not accept the asterisk (*) wildcard character.
 - The **To Date** field—Use this field to specify the ending date and time if you want to search within a specific time period. This field does not accept the asterisk (*) wildcard character.
 - The **Business object type** field—Use this field to specify the type of business object associated with the failed events.
 - The **Exception text** field—Use this field to specify the text associated with the exception you want to search against.

4. Click **OK** to begin the search.

The Search Results page opens and displays a list of all failed events that meet the specified criteria.

Working with data in failed events

Each failed event has data associated with it; often, that data can be edited before an event is resubmitted. There are two basic types of data for a failed event: data about the event, and business data.

Data about the failed event

Each failed event has the following data associated with it:

- The unique message ID and session ID for the event
- The service invocation type between SCA components
- The names of the module and component from which the event originated (the source). The failed event manager determines the source of an event based on the location where the invocation failed.
- The names of the destination module, component and method for the event. The failed event manager determines the event's destination based on the location where the invocation failed.
- The time the event failed
- The exception thrown when the event failed

This data cannot be edited. In addition, failed events can have associated trace and expiration data, both of which can be edited.

Business data

Events typically include business data. Business data can be encapsulated in a business object, or it can be simple data that is not part of a business object. Business data is edited with the business data editor available in the failed event manager.

Related tasks

“Browsing data in failed events”

“Editing trace or expiration data in a failed event” on page 239

The Failed Event Details page enables you to set or modify values for the trace control and expiration date associated with a failed event.

“Editing business data in a failed event” on page 240

Browsing data in failed events

Each failed event has two types of data associated with it:

- Failed event data—Information about the failed event itself, including the source and destination for the event, the time it failed, the exception it failed with, its message and session IDs, and its trace and expiration settings.
- Business data—Information contained in the event. The business data can be encapsulated in a business object, or it can be simple data that is not part of a business object.

Security role required: You must be logged as administrator or operator to perform this task.

1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.

2. From the failed event manager's Search Results page, click the ID (found in the Message ID column) of the failed event whose data you want to browse.

The Failed Event Details page opens and displays all of the information about the event.

3. If you want to browse the business data associated with the failed event, click **Edit business data**.

The Business Data Editor collection page opens, displaying the business data associated with the failed event. Each parameter name in the hierarchy is a link. If the parameter is a simple data type, clicking its name will open up a form so you can edit the parameter's value. If the parameter is a complex data type, clicking its name will expand the hierarchy further.

Related tasks

"Finding failed events" on page 233

"Editing trace or expiration data in a failed event"

The Failed Event Details page enables you to set or modify values for the trace control and expiration date associated with a failed event.

"Editing business data in a failed event" on page 240

Editing trace or expiration data in a failed event

The Failed Event Details page enables you to set or modify values for the trace control and expiration date associated with a failed event.

Important: Any edits you make to the trace or expiration data are only saved locally until you resubmit the event. If you perform any other action before resubmitting the event, all edits are lost.

Failed events can be resubmitted with trace to help you monitor the event processing. Tracing can be set for a service or a component, and it can be sent to a log or to the Common Event Infrastructure (CEI) server. When you view the failed event data on the Failed Event Details page, the default trace value `SCA.LOG.INFO;COMP.LOG.INFO` is shown for the event. If you resubmit the event with this default setting, no trace occurs when the session calls an SCA service or executes a component.

Some failed events also have an expiration. If a user has specified an expiration with the asynchronous call that sends the event, that data persists even if the event fails, and the expiration time appears in the **Resubmit Expiration Time** field on the Failed Event Details page. Expired failed events cannot be resubmitted successfully. To prevent a second failure, you can edit the expiration date for the event to ensure that it is not expired when it is resubmitted.

Security role required: You must be logged in as administrator or operator to perform this task.

1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.
2. From the failed event manager's Search Results page, click the ID (found in the Message ID column) of the failed event whose data you want to edit.
The Failed Event Details page opens.

3. If the event has an expiration date that causes it to expire before it is resubmitted, edit the expiration in the **Resubmit expiration time** field.

The expiration time shown is local to the server. The value for this field must be formatted according to your specified locale. An example of the correct format for your locale is provided above the field.

4. If you want to enable tracing for the failed event, specify a new value in the **Trace Control** field. For detailed information about trace values, see the Monitoring topics in the WebSphere Process Server Information Center.
5. Do one of the following:
 - If the edited data is correct and you want to resubmit the event, click **Resubmit** to make the changes at a server level.
 - If you want to remove the changes you made, click **Undo local changes**.
The edited failed event is resubmitted for processing and is removed from the failed event manager.

Editing business data in a failed event

The failed event manager provides a business data editor so you can edit the business data associated with a failed event before you resubmit it. For each failed event, the editor displays the associated business data in a hierarchical format; the navigation tree at the top of the table is updated as you navigate through the parameters to give you a clear picture of where you are in the hierarchy.

Business data can be encapsulated into a business object, or it can be simple data that is not part of a business object. A failed event can have both simple data and a business object associated with it.

You can edit only simple data types (for example, String, Long, Integer, Date, Boolean). If a data type is complex (for example, an array or a business object), you must navigate through the business data hierarchy until you reach the simple data types that make up the array or business object. Complex data is denoted by an ellipsis (...) in the Parameter Value column.

Important: Any edits you make to business data are saved locally. Changes are not made to the corresponding business data in the server until you resubmit the failed event.

Security role required: You must be logged in as administrator or operator to perform this task.

1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.
2. From the failed event manager's Search Results page, click the ID (found in the Message ID column) of the failed event whose data you want to edit.

The Failed Event Details page opens.

3. From the Failed Event Details page, click **Edit business data** to access the Business Data Editor collection page.

This page displays a hierarchical view of all of the data associated with the failed event.

4. Navigate through the business data hierarchy by clicking on the name of each parameter (these appear as links in the Parameter Name column). When you have located the parameter whose value you want to edit, click its name.

If the parameter has an editable value, the Business Data Editor page opens.

5. In the **Parameter value** field, specify the new value for the parameter.
6. Click **OK**.

The change is saved locally and you are returned to the Business Data Editor collection page.

7. If you want to remove the changes you made, click **Undo local business data changes**.

All of the edits are removed and the business data is returned to its original state.

8. If the edited business data is correct, click **Resubmit** to make the changes at a server level.

The edited failed event is resubmitted for processing and is removed from the failed event manager.

Resubmitting failed events

If you want to try to execute the event again, you must resubmit it from the failed event manager. You can resubmit an event without changes, or you can edit the business data parameters before resubmitting it.

When a failed event is resubmitted, the processing resumes only for the failed branch, not for the entire event.

Tracing is available for resubmitted events to help monitor the event's processing. Tracing can be set for a service or a component, and its output can be sent to a log or to the Common Event Infrastructure (CEI) server.

You can also use the event's unique message ID to track its success or failure. If a resubmitted event fails again, it is returned to the failed event manager with its original message ID and an updated failure time.

Resubmitting an unchanged failed event

You can resubmit one or more unchanged failed events to be processed again. Processing resumes only for the failed branch, not for the entire event.

Security role required: You must be logged in as administrator or operator to perform this task.

1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.
2. From the Search Results page, select the check box next to each failed event you want to resubmit.
3. Click **Resubmit**.

Each selected event is resubmitted for processing and is removed from the failed event manager.

Resubmitting a failed event with trace

You can monitor the resubmission of a failed event to determine whether it executes successfully. The failed event manager provides optional tracing for all failed events.

Tracing can be set for a service or a component, and it can be output to a log or to the Common Event Infrastructure (CEI) server. For detailed information about setting and viewing trace, see the Monitoring topics in the WebSphere Process Server Information Center.

Security role required: You must be logged in as administrator or operator to perform this task.

1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.

2. From the Search Results page, select the check box next to each failed event you want to resubmit.
3. Click **Resubmit with trace**.
4. From the Resubmit with Trace page, specify the level of trace you want to use in the **Trace control** field.
By default, the value is `SCA.LOG.INFO;COMP.LOG.INFO`. With this setting, no trace occurs when the session calls an SCA service or executes a component.
5. Click **OK** to resubmit the failed event and return to the Search Results page.

To view the trace log for a resubmitted event, open the corresponding component logger or use the CEI log viewer.

Deleting failed events

If you do not want to resubmit a failed event, or if you have failed events that have expired, use the failed event manager to delete them from the server. The failed event manager provides three options for deleting failed events.

Security role required: You must be logged in as administrator or operator to perform this task.

1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.
2. From the failed event manager's Search Results page, do one of the following:
 - If you want to delete one or more specific failed events, select the check box next to each event and then click **Delete**.
 - If you want to delete only those failed events that have expired, click **Delete expired events**. Note that this deletes only the expired events in the current set of search results.
 - If you want to delete all failed events on the server, click **Clear all on server**.

Using the Common Event Infrastructure

The Common Event Infrastructure is a core component of WebSphere Process Server.

The Common Event Infrastructure provides facilities for the run-time environment to persistently store and retrieve events from many different programming environments. Events are represented using the Common Base Event model a standard, XML-based format that defines the structure of an event.

The Common Event Infrastructure provides WebSphere Process Server with standard formats and mechanisms for managing event data. The following facilities are provided:

- Standard interfaces and services for WebSphere applications to create event objects, store them, send them, and retrieve them later.
- Facilities that pass event objects to registered applications either directly, in the context of the producing (source) application, or indirectly through Java Message Service (JMS). There are event emitters for Business Process Execution Language (BPEL) -based processes and for Enterprise JavaBeans (EJB) invocations based on deployment descriptor extensions.
- The Common Base Event browser for browsing stored events.

Administering the Common Event Infrastructure

You can perform the following administrative tasks to control the operation of the Common Event Infrastructure components at run-time.

Logging and tracing in the Common Event Infrastructure

You can enable logging and tracing in order to debug problems with an application using the Common Event Infrastructure.

The Common Event Infrastructure components use the JSR47 Java logging framework, which is available in WebSphere Process Server and client environments. For more information about using the logging framework, refer to the WebSphere Application Server troubleshooting documentation.

The following table indicates the logger names used by the Common Event Infrastructure components.

Table 12. Logger names

Component	Logger name
Root logger name	com.ibm.events
Event catalog	com.ibm.events.catalog
Event server subcomponents	com.ibm.events.access com.ibm.events.bus com.ibm.events.distribution com.ibm.events.server
Default data store plug-in	com.ibm.events.datastore
Event emitter	com.ibm.events.emitter
Notification helper	com.ibm.events.notification
Configuration	com.ibm.events.configuration
Installation	com.ibm.events.install
Miscellaneous utilities	com.ibm.events.util

DB2 database maintenance

If you are using a DB2 event database, you should periodically perform database maintenance by running the provided scripts.

The DB2 versions currently supported for running these maintenance scripts are

- DB2 Universal Version 8.1
- DB2 Universal Version 8.2.1

Updating database statistics:

To enable the DB2 database to optimize queries and find free space, update the database statistics using the runstats script.

It is recommended that you update the database statistics regularly, and especially under any of these circumstances:

- Events have been purged from the database
- A large number of events have been inserted into the database

- Tables have been reorganized using the reorg script
- Indexes have been added or removed from a table

The runstats script is located in the *install_root/event/dbscripts/db2* directory.

To update the database statistics, run the following command:

```
runstats.sh db_alias db_user [db_password]
```

The parameters are as follows:

db_alias

The database alias. The event database must be catalogued on the DB2 client; if you are running the script on the DB2 server, the database is already catalogued.

db_user

The database user ID to use. This parameter is required.

db_password

The database password. This parameter is optional. If you do not specify the password on the command line, the DB2 database will prompt you for it.

For example, the following command updates the DB2 database statistics where the event database name is *event*, the database user ID is *dbadmin*, and the password is *mypassword*:

```
runstats.sh event dbadmin mypassword
```

Reorganizing database tables:

After events are purged from a DB2 event database, reorganize the database tables using the reorg script.

The reorg script is located in the *profile_root/event/dbscripts/db2* directory.

To reorganize the event database tables, run the following command:

```
reorg.sh db2_alias db_user [db_password]
```

The parameters are as follows:

db2_alias

The database alias. The event database must be catalogued on the DB2 client; if you are running the script on the DB2 server, the database is already catalogued.

db_user

The database user ID to use. This parameter is required.

db_password

The database password. This parameter is optional. If you do not specify the password on the command line, the DB2 database prompts you.

For example, the following command reorganizes the event database tables using the database user ID *dbadmin*, the password *mypassword*, and the database name is *event*:

```
reorg.sh event dbadmin mypassword
```


After you run the reorg script, you should update the database statistics using the runstats script. For more information, see *Updating database statistics*.

Purging events from the event database:

You can use the provided scripts to rapidly purge large numbers of events from the event database.

The default data store plug-in provides a set of utilities you can use to periodically perform a rapid purge of large numbers of old events from the event database. These utilities are distinct from the **eventpurge.jacl** event server command, which deletes events matching specified criteria.

The rapid purge capability uses the concept of **buckets**. A bucket is a set of tables used to store events in the event database. The default data store plug-in uses two buckets:

- The **active bucket** is the bucket containing the most recent events; new events are stored in the active bucket. The active bucket cannot be purged using the rapid purge utility.
- The **inactive bucket** contains older events. Events stored in the inactive bucket can be queried, deleted, or modified, but typically no new events are stored in the inactive bucket. The inactive bucket can also be purged by the rapid purge utility.

Each event is stored in only one bucket. From the perspective of an event consumer, the distinction between the active and inactive buckets is invisible; a consumer can query, modify, or delete a specific event without knowing which bucket the event is stored in. The advantage of this approach is that the inactive bucket can be rapidly purged using database-specific interfaces without affecting the active bucket; normal event traffic can continue even while the purge operation is taking place.

After the inactive bucket is purged, you can then swap the buckets so that the active bucket becomes inactive and the inactive bucket becomes active. Swapping buckets is possible only when the inactive bucket is empty.

Note: Although new events are generally stored only in the active bucket, under some circumstances events might be stored in the inactive bucket immediately after the buckets are swapped. The data store plug-in checks periodically to determine which bucket is currently marked as active, but until the next check takes place, some events might continue to be stored in the inactive bucket. Similarly, events sent as part of a batch are all stored in the same bucket, even if that bucket becomes inactive while the batch is still being processed.

If you want to use the fast purge capability, it is your responsibility to determine how frequently to swap buckets or purge the inactive bucket, depending upon event traffic, storage space, archival requirements, or other considerations.

Viewing or changing the active bucket status:

The active bucket status indicates which bucket is currently active and which is currently inactive.

To view or change the active bucket status, use the **eventbucket.jacl** script (found in the *profile_root/event/bin* folder):

```
wsadmin -f eventbucket.jacl [-status] [-change]
```

This command has the following options:

-status

Use this option to see information about the current bucket configuration, including the active bucket setting and the bucket check interval (the frequency with which the data store plug-in checks to determine which bucket is active).

-change

Use this option to swap the active and inactive buckets. The inactive bucket must be empty before you can use this option.

Purging the inactive bucket:

The method used to purge the inactive bucket varies depending on the database software.

Note: The rapid purge utility is not supported for Cloudscape databases.

Purging the inactive bucket for a DB2 database (z/OS systems):

On z/OS systems, the rapid purge utility for a DB2 event database is implemented using the DB2 load utility.

To purge the inactive bucket:

1. Use the **eventbucket.jacl** command to identify the inactive bucket (bucket 0 or bucket 1).
2. Upload the appropriate utility control file. These files are generated during database configuration and are located in the *profile_path/event/dbscripts/db2zos* directory (*profile_path* is the path to the directory containing the profile for the WebSphere Process Server runtime environment. Upload one of the following files:
 - If bucket 0 is inactive, upload fastpurge00.ctl.
 - If bucket 1 is inactive, upload fastpurge01.ctl.

Note: The control file must be uploaded with a fixed record format and a logical record length of 80.

3. On the z/OS host, go to the ISPF DB2I Primary Option Menu and select the **Utilities** option.
4. Specify the following information:

Field	Value
Function	EDITJCL
Utility	LOAD
Statement Data Set	The name of the data set containing the uploaded control file
LISTDEF	NO
Template	NO

5. Press the Enter key to continue to the next panel.
6. In the recdsn entry field, specify the name of the data set containing the uploaded control file.
7. Press the Enter key. The JCL script to purge the inactive bucket is generated.

8. Press the Enter key to clear the output messages.
9. Edit the generated JCL script as needed.
10. Submit the JCL script.

Changing the bucket check interval:

The bucket check interval is specified in the DataStoreEjb.jar file.

This value specifies how frequently the data store plug-in checks to determine which bucket is active. The default value is 5 minutes (300 seconds). A shorter interval reduces the likelihood that events will be stored in the inactive bucket after swapping, but might decrease performance.

To change the bucket check interval:

1. In the WebSphere Process Server administrative console, navigate to **Enterprise Applications > EventServer > EJB Modules > DataStoreEjb.jar > View Deployment Descriptor**.
 - a. Look for the DefaultDataStoreEJB parameters.
 - b. Expand the `<session id="DataStoreHelperEJB">` menu, and look for the `<env-entry-value>`.

This is the bucket check interval value, which is set in number of seconds.
2. Modify the value of the BucketCheckInterval environment variable to specify the bucket check interval in seconds.

Working with events

Java objects are used as representations of the Common Base Event specification.

The Common Event Infrastructure represents events as Java objects. Specifically, each event is an instance of a class implementing the `org.eclipse.hyades.logging.events.cbe.CommonBaseEvent` interface, which is a Java representation of the Common Base Event specification. The `org.eclipse.hyades.logging.events.cbe` package is part of the Eclipse-based Hyades environment, which is a set of standards and open-source tools for testing, tracing, and monitoring. For more information, see <http://www.eclipse.org/hyades/>.

The typical life cycle of an event is as follows:

1. To send an event, an event source creates a new instance of `CommonBaseEvent`, populates it with property data, and then submits it to an emitter.
2. The emitter optionally uses the content completion mechanism (if implemented) to populate the event with required property data. The emitter then validates the event and checks it against the currently configured filter criteria. If the event is valid and passes the filter criteria, the emitter sends the event to the event server. For more information about event processing by the emitter, see *"Sending events"* on page 255.
3. If persistence is enabled, the event server stores the event in a persistent data store.
4. If publishing is enabled, the event server publishes the event to one or more Java Message Service (JMS) destinations. Event consumers subscribing to these destinations then receive notifications of the new event. The event consumers then use the Notification Helper to convert the received JMS messages back into a `CommonBaseEvent` instance.

An event consumer might also submit a query to retrieve the event from the data store. Typically, a consumer uses the query interface to retrieve historical events, especially during startup processing.

After receiving the event, an event consumer reads the event property data and processes the event.

5. When it is no longer needed, the event can be purged from the data store.

The Common Base Event specification, which is based on the XML Schema definition language, defines two kinds of event property data:

- Properties represented by simple data types, encoded in XML as attributes of the `CommonBaseEvent` element. These include properties such as *globalInstanceId*, *severity*, and *msg*. The `CommonBaseEvent` Java class represents these values as strings or integers, as appropriate.
- Properties represented by complex data types and encoded in XML as subelements of the `CommonBaseEvent` element. These include properties such as *situation*, *sourceComponentId*, and *extendedDataElements*, each of which has nested properties of its own. These complex types are represented by specialized Java classes defined in the `org.eclipse.hyades.logging.events.cbe` package. For example, the *sourceComponentId* property is represented by an instance of `ComponentIdentifier`.

The `CommonBaseEvent` interface defines getter and setter methods for each property, as well as helper methods to simplify creation of complex properties. An event source uses the setter methods (or the helper methods) to populate an event with property data before submitting it to an emitter; an event consumer uses the getter methods to retrieve the property data from a received event.

For more information about the XML Schema specification, see <http://www.w3.org/XML/Schema>.

Creating an event object:

New events are created by using an *event factory*.

To create new events in your event source, you use an *event factory*, which is an object that returns new instances of `CommonBaseEvent` or of the specialized classes representing complex property data types.

There are two ways you can access an event factory:

- You can create a new event factory using the *event factory factory*. Use this approach if no appropriate event factory is already available. When you create a new event factory, you can optionally specify a content handler to provide automatic content completion.
- You can use an existing event factory that has been bound into a Java Naming and Directory Interface (JNDI) namespace. Use this approach if an administrator has provided an event factory for you to use; this ensures that any events you create conform to the appropriate business rules, because the event factory might be configured with a content handler.

Creating a new event factory:

An *event factory* is used to create new events.

To create a new event factory, use the *event factory factory*, implemented as the class `EventFactoryFactory`. This class has no instances; instead, it provides two

static methods used to create event factories. The choice of which method to use depends upon whether you want to use a content handler to implement automatic content completion. For more information, see “*Completing event content automatically*” on page 251.

To create a generic event factory with no content handler, use the `createEventFactory()` static method of `EventFactoryFactory`:

```
EventFactory eventFactory =  
    (EventFactory) EventFactoryFactory.createEventFactory();
```

To create an event factory with a content handler, use the `createEventFactory(ContentHandler)` method, specifying the content handler you want to use:

```
EventFactory eventFactory =  
    (EventFactory) EventFactoryFactory.createEventFactory(contentHandler);
```

In either case, the returned object is an event factory you can use to create new events.

Getting an event factory by JNDI lookup:

Use a JNDI lookup to retrieve event factories, which use JNDI for event sources.

If an administrator has bound an existing event factory into JNDI for event sources to use, perform a standard JNDI lookup to retrieve the event factory:

```
import javax.naming.*  
import org.eclipse.hyades.logging.events.cbe.*  
  
Context context = new InitialContext();  
EventFactory eventFactory =  
    (EventFactory) context.lookup("com/ibm/events/EventFactory");
```

The returned object is the provided event factory; if the event factory is configured with a content handler, an instance of the content handler is also created locally. For more information about content handlers and JNDI, see “*Completing event content automatically*” on page 251.

Creating and populating an event:

After obtaining an event factory, you can create event objects and populate them with property data.

Most event properties are defined as optional by the Common Base Event specification, but the following properties are required:

- *version* (a string attribute)
- *creationTime* (an XML Schema `dateTime` attribute; see <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/#dateTime>)
- *sourceComponentId* (a complex `ComponentIdentification` element)
- *situation* (a complex `Situation` element)

Note: The *version* attribute is defined as optional by the Common Base Event specification, but if it is not specified, the default value 1.0 is assumed. Because the Common Event Infrastructure supports only version 1.0.1 of the specification, this value must be specified.

If you try to send an event that is missing any of these properties, the emitter rejects the event and throws an `EventsException` exception.

The following code fragment creates an event and populates it with the minimal required property data:

```
CommonBaseEvent event = eventFactory.createCommonBaseEvent();

event.setVersion("1.0.1");           // set version

long currentTime = System.currentTimeMillis(); // get current time
event.setCreationTimeAsLong(currentTime);    // and set creationTime

// set sourceComponentId (a complex type)
event.setSourceComponentId("Windows",      // application
                           "svchost.exe",  // component
                           "tlntsvr.exe",  // subcomponent
                           "http://www.ibm.com/namespaces/autonomic/Windows",
                                       // componentType
                           "win386_svc",   // componentIdType
                           "9.45.72.138",  // location
                           "IPV4"         // locationType
                           );

// create situation object
Situation situation = eventFactory.createSituation();

// set situationType to AvailableSituation (a complex type)
situation.setAvailableSituation("EXTERNAL", // reasoningScope
                                "NOT AVAILABLE", // availabilityDisposition
                                "STARTABLE",    // operationDisposition
                                "FUNCTION_PROCESS"); // processingDisposition

// set situation
event.setSituation(situation);
```

This example first uses an event factory to create a new event instance, *event*. First it sets the *version* property; then it retrieves the current system time and uses the `setCreationTimeAsLong(long)` method to set the value of the *creationTime* property. An alternative is to use the `setCreationTime(String)` method, which sets the creation time using the XML `dateTime` format (for example, "2004-07-29T13:12:00-05:00").

The next required property, *sourceComponentId*, is a complex property represented by an instance of `ComponentIdentification`, which has properties of its own. However, it is not necessary to directly instantiate or interact with this object (although it is possible to do so). Instead, the next statement in the example uses a helper method, `setSourceComponentId()`, to specify the nested properties; the helper method uses these values to create an instance of `ComponentIdentification`, which it then uses to set the value of the *sourceComponentId* property of the event.

Similar helper methods exist for setting other complex properties (for example, `setMsgDataElement()`, `addAssociatedEvent`, and `addExtendedDataElement()`). Many of these methods exist in multiple versions with different signatures, making it possible to specify property values in different ways. Refer to the Javadoc API documentation for complete information on these methods.

The last required property in the example, *Situation*, is another complex property. In this case the situation object must be instantiated directly using an event factory; the example then uses a helper method to set the *situationType* property, which is itself a complex subelement.

In an actual application, a useful event needs to include more information than is shown here, but this is the minimum required by the Common Base Event specification and the Common Event Infrastructure. The event is now valid and can be submitted to an emitter.

Completing event content automatically:

By setting properties and policies, event content can be automatically completed.

In some situations, you might want some event property data to be automatically set for every event you create. This is a way to fill in certain standard values that do not change (such as the application name), or to set some properties based on information available from the runtime environment (such as creation time or thread information). You can also set policies that govern event content according to business rules; for example, you might require that any event with a particular extension name have its severity set to a certain value.

You can do this by creating a *content handler*. A content handler is an object that automatically sets the property values of each event based on any arbitrary policies you want to use. The Common Event Infrastructure does not restrict how a content handler can modify event data, so long as the event still conforms to the Common Base Event specification.

To ensure that all event sources comply with the same policies, you can create an event factory associated with a content handler (using `EventFactoryFactory`) and then bind the created event factory into a JNDI namespace. Instead of creating their own event factories, event sources can then perform JNDI lookups to access the event factory that already exists, without any knowledge of the content handler. If your business rules later change, you can modify the content handler in one place.

An event source does not need to do anything to enable content completion. If an event factory is associated with a content handler, each event it creates carries with it a reference to that content handler. When the event is submitted to an emitter, the event calls the `completeEvent()` method of the content handler, passing a reference to itself. This ensures that the correct policies are applied to the event after the event source is finished setting event-specific properties, but before the event is validated and processed by the emitter.

Note: When an event is transmitted from one process to another, the reference to the content handler is not transmitted with it. This is because content completion relies upon the environment where the event originates, and the necessary information might not be available elsewhere. This restriction does not affect calls between applications that are local to one another (for example, a call to an enterprise bean using its local interface).

To create a content handler, follow these steps:

1. Create a new Java class implementing the `org.eclipse.hyades.logging.events.cbe.ContentHandler` interface. This interface defines a single method called `completeEvent(CommonBaseEvent)`; the parameter is the event whose content is to be completed. In your implementation of this method, you can use the getter and setter methods of `CommonBaseEvent` to process the event property data in accordance with any policies that apply.

Note: When an event source uses JNDI to retrieve an event factory, the content handler is returned along with the event factory. For this reason, the content handler must be serializable.

The following example is a simple content handler that automatically sets the extension name of each event:

```
import java.io.Serializable;
import org.eclipse.hyades.logging.events.cbe.*;

public class BusinessContentHandler
    implements ContentHandler, Serializable {

    public void completeEvent(CommonBaseEvent event)
        throws CompletionException {
        event.setExtensionName("business");
    }
}
```

2. Associate the content handler with an event factory. To do this, specify the content handler when creating the event factory:

```
EventFactory eventFactory =
    (EventFactory) EventFactoryFactory.createEventFactory(contentHandler);
```

The returned event factory is permanently associated with the specified content handler.

Retrieving data from a received event:

An event source uses methods of the `CommonBaseEvent` to retrieve event property data.

When an event source receives an event, it can then use the getter methods of `CommonBaseEvent` to retrieve the event property data. For example, the following code fragment retrieves a single event and then reads the content of the `msg` property.

```
CommonBaseEvent event = eventAccess.queryEventByGlobalInstanceId(eventId);
String eventMessage = event.getMsg();
```

If the property you want to retrieve is a complex property (a subelement of `CommonBaseEvent` in the Common Base Event specification), the returned value is an instance of the specialized class representing the complex data type. You can then use the getter methods of the returned object to retrieve the property data from that object. For example, the following code fragment retrieves the value of `componentId`, which is a complex property; it then retrieves the content of the nested `component` property, which is a string, to read the name of the source component.

```
CommonBaseEvent event = eventAccess.queryEventByGlobalInstanceId(eventId);
ComponentIdentification componentId = event.getSourceComponentId();
String componentName = componentId.getComponent();
```

Converting XML events:

An event source can convert XML-formatted events from other applications.

In addition to creating new events, an event source might convert events received from other applications in XML format. Similarly, an event consumer might convert events to XML format for forwarding to another application. The `org.eclipse.hyades.logging.events.cbe.EventFormatter` class provides methods you can use to convert between `CommonBaseEvent` instances and XML.

Using `EventFormatter`, you can convert an instance of `CommonBaseEvent` into a string containing either an XML document or an XML fragment. Similarly, you can convert from an XML document or fragment into an instance of `CommonBaseEvent`.

For more information about `EventFormatter`, refer to the Javadoc documentation in the `org.eclipse.hyades.logging.events.cbe` package.

Accessing event instance metadata:

Accessing event instance metadata is done through the Java-based Eclipse Modeling Framework.

The `org.eclipse.hyades.logging.events.cbe` package, which provides the classes and interfaces required for working with event objects, is based on the Eclipse Modeling Framework (EMF). EMF is a Java framework used to generate application code based on a structured data model; it also provides interfaces in the generated code that can be used to access metadata describing the data model. (Refer to the Eclipse Modeling Framework documentation at <http://www.eclipse.org/emf> for more information about EMF.)

By using these interfaces, EMF-compatible tools can interact with `CommonBaseEvent` event data without any prior knowledge of the data model or access to the implementation. This makes it possible for development tools to generate code that transfers data from other data models into the `CommonBaseEvent` model. Application developers can then focus on writing code that uses the data rather than code that builds the data.

For example, consider an event source that monitors network events and describes its own data model in terms of EMF. With access to both data models, a development tool could display the fields of the event source data model alongside the fields of the `CommonBaseEvent` data model. A developer could then use a graphical interface to indicate how the fields in the event source model are mapped to fields in the `CommonBaseEvent` model; for example, a field called `Workstation.name` in the event source data model might correspond to the `CommonBaseEvent.sourceComponentId.location` field in the `CommonBaseEvent` data model. Because both data models are described using standard EMF interfaces, the tool could generate code that handles the transfer of data between the two models.

The following code fragment is a simple example of how a development tool might use EMF interfaces to query information about the `CommonBaseEvent` data model and then use that information to interact with an event instance. This example could be part of a simple event consumer; it iterates through all of the fields of an event instance and, for each one, prints the name and value of the field.

```
// event is a valid CommonBaseEvent instance

// Get list of event instance structural features (fields)
List features = event.eClass().getEAllStructuralFeatures();

// iterate through list; print names and values
for (int i = 0 ; i < features.size() ; i++)
{
    EStructuralFeature feature = (EStructuralFeature)features.get(i);
    Object value = eObj.eGet(feature);
    System.out.println(feature.getName() + ":" + value);
}
```

The `CommonBaseEvent` data model is described in the EMF files `cbe.ecore` and `cbe.genmodel`. These files are included with the Common Event Infrastructure SDK; you can import them into an Eclipse-based development environment and then use EMF to generate code that interacts with `CommonBaseEvent` objects.

Creating an event source

An event source interacts with the event server *through* an emitter object.

An *event source* is any application that uses an emitter to send events to the event server. The following applications are examples of event sources:

- An adapter or monitor that generates events related to monitored resources
- An application that generates notification events
- An application that forwards events from other sources

An event source is implemented in the Java programming language, using either the Java 2 Platform, Standard Edition (J2SE) or the Java 2 Platform, Enterprise Edition (J2EE). An event source must submit valid events conforming to the Common Base Event model. Each event is represented as a Java object.

Emitters and emitter factories

An event source does not interact directly with the event server; instead, it interacts with an object called an emitter (an implementation of the `com.ibm.events.emitter.Emitter` interface). An emitter is a local object that provides methods for sending events.

In general, the emitter handles the details of event transmission; the developer of an event source does not need to be concerned about the event server location, the filter settings, or the underlying transmission mechanism. Details such as these are governed by the *emitter factory*, an object configured by an administrator and bound in a Java Naming and Directory Interface (JNDI) namespace. An emitter factory is an instance of `com.ibm.events.emitter.EmitterFactory` and is used to create emitter objects. It also defines the behavior of the emitters it creates; it includes settings for the following:

- The preferred *transaction mode*. This setting specifies whether the emitter attempts to send each event in a new transaction or within the current transaction. An event source can change this setting for a particular emitter or event submission, but the profile specifies the default value. (This setting is valid only in a J2EE container; the J2SE platform does not provide transaction controls.)
- The preferred *synchronization mode*. This setting specifies whether events are sent using synchronous or asynchronous transmission. *Synchronous transmission* means that the `sendEvent()` method does not return control to the caller until the event has been processed; *asynchronous transmission* means that the method returns immediately after the event is submitted, and the caller has no further information about event processing. An event source can change this setting for an emitter or for an event submission, but the default value is specified by the profile.
- The *transmission profiles* to use. A transmission profile is a configuration object that defines a specific transmission mechanism for sending events to the event server. An emitter factory profile can specify two transmission profiles, one for synchronous transmission and one for asynchronous transmission. An event source cannot change the transmission profiles used by an emitter.

- The filter configuration to use for the emitter. The filter configuration defines what filtering plug-in is used to filter events submitted to the emitter. The Common Event Infrastructure includes a default filter plug-in, but you can also implement your own filter plug-in if you want to use a different filtering engine.

An administrator can create multiple emitter factory profiles, each one defining a different emitter configuration. An event source obtains an emitter using the emitter factory associated with an existing emitter factory profile; therefore, all emitters created by a particular emitter factory will have the same default behavior. For more information, see *Obtaining an emitter*.

Note: If your event source is running with Java 2 security enabled, and you want to generate your own globally unique identifiers (GUIDs), you must modify your policy file to enable correct processing. Add the following entries:

```
permission java.io.FilePermission "${java.io.tmpdir}${/}guid.lock",
    "read, write, delete";
permission java.net.SocketPermission "*", "resolve";
```

Obtaining an emitter:

Before you can obtain an emitter, there must be at least one emitter factory profile configured.

For each emitter factory profile, an emitter factory is automatically created and is accessible using the JNDI name of the emitter factory profile.

To obtain an emitter, follow these steps:

1. Perform a JNDI lookup specifying the name of the emitter factory you want to use for your emitter. This is the JNDI name specified by an administrator when defining an emitter factory profile.
2. Call the *getEmitter()* method of the emitter factory. The returned object is an emitter configured according to the options defined in the emitter factory profile you specified. If the emitter factory is unable to obtain an emitter, it throws an *EmitterException* exception.

Note: If your event source is a J2EE client application running in a secure environment, and the emitter profile you are using specifies asynchronous transmission profiles, you must specify a JMS user name and password in order to get an emitter. To do this, use the *getEmitter(String, String)* method, passing the JMS user name and password you want to use. For more information, refer to the Javadoc documentation for the *com.ibm.events.emitter* class.

The following code fragment obtains an emitter configured with the profile *Default*:

```
import javax.naming.*
import com.ibm.events.emitter.*

Context context = new InitialContext();
EmitterFactory emitterFactory =
    (EmitterFactory) context.lookup("com/ibm/events/configuration/emitter/Default");
Emitter emitter = emitterFactory.getEmitter();
```

Sending events:

An event source sends events in the form of Java objects.

Specifically, each event is an instance of a class implementing the `org.eclipse.hyades.logging.events.cbe.CommonBaseEvent` interface, which is a Java representation of the Common Base Event specification. For more information, see *The Common Base Event model*.

To send an event, use the `sendEvent()` methods of the Emitter interface. When you submit an event to an emitter, the following occurs:

1. The emitter calls the `complete()` method of the event, triggering optional content completion. See *“Completing event content automatically” on page 251* for more information.
2. The emitter assigns a sequence number and global instance identifier to any event that does not already have them.
3. The emitter validates the event to ensure that it conforms to the Common Base Event specification.

Note: The current Common Base Event specification allows only one extended data element with a given name at each level of the event containment hierarchy, but this restriction will not be included in future versions of the specification and is not enforced by the Common Event Infrastructure.

4. If filtering is active, the emitter checks the event against the current filter criteria to determine whether the event should be sent or discarded.
5. Finally, if the event is valid and passes the filter criteria, the emitter sends the event to the event server for persistence and distribution to event consumers.

If the event is not valid, or if the emitter encounters a problem when trying to send the event to the event server, an exception is thrown.

Note: If WebSphere security is enabled, the application user ID must be mapped to the `eventCreator` or `eventAdministrator` role to send events using synchronous EJB transmission.

Sending an event with the current emitter settings:

Events can be sent with current emitter settings, if transmission modes do not need to be specified.

If you do not need to specify a particular transmission mode or transaction mode, you can send an event using the current emitter settings. These settings are initially defined by an administrator in the emitter factory profile, but they can later be changed by event consumers.

To send an event using the current emitter settings, use the `sendEvent(CommonBaseEvent)` method.

```
String eventId = emitter.sendEvent(event);
```

In this example, `emitter` is an Emitter instance, and `event` is a `CommonBaseEvent` instance.

The returned value, `eventId`, is the globally unique identifier of the event (the value of the `globalInstanceId` field of `CommonBaseEvent`). If the event does not have a global instance identifier when you submit it, the emitter assigns one automatically.

Note: Submitting an event to an emitter does not guarantee that the event is sent to the event server, because the filter settings might cause the event to be discarded. A successful call to `sendEvent()` means only that the event was successfully processed by the emitter.

Overriding the current emitter settings:

Emitter factory profile settings can be changed by event consumers.

When sending an event, you can specify options that override the current transaction mode, synchronization mode, or both, currently configured for the emitter. These settings are initially defined by an administrator in the emitter factory profile, but they can later be changed by event consumers.

An emitter might not support all synchronization and transaction modes. The available modes are subject to the following limitations:

- The synchronization modes supported by an emitter are defined by the emitter factory profile. You can find out which modes are supported by a particular emitter by calling the `isSynchronizationModeSupported()` method; see the Javadoc API documentation for `com.ibm.events.emitter.Emitter` for more information.
- Transactions are supported only in a J2EE container.

If you attempt to use a mode that is not supported, the emitter throws a `TransactionModeNotSupportedException` or `SynchronizationModeNotSupportedException` exception.

To override the emitter settings, use the `sendEvent(CommonBaseEvent, int, int)` method.

```
String eventId = emitter.sendEvent(event,  
                                  synchronizationMode,  
                                  transactionMode);
```

The parameters are as follows:

event

The event object (an instance of `CommonBaseEvent`) you want to send.

synchronizationMode

An integer constant defined by the interface `SynchronizationMode`. This should be one of the following constants:

- `SynchronizationMode.ASYNCHRONOUS` (send the event asynchronously)
- `SynchronizationMode.SYNCHRONOUS` (send the event synchronously)
- `SynchronizationMode.DEFAULT` (use the current emitter setting)

transactionMode

An integer constant defined by the interface `TransactionMode`:

- `TransactionMode.NEW` (send the event in a new transaction)
- `TransactionMode.SAME` (send the event in the current transaction)
- `TransactionMode.DEFAULT` (use the current emitter setting)

The event is sent with the options you specify. These options apply only to the single event being sent; no changes are made to the emitter settings, and subsequent event submissions are not affected.

The returned value, `eventId`, is the globally unique identifier of the event (the value of the `globalInstanceId` field of `CommonBaseEvent`). If the event does not have a `globalInstanceId` when you submit it, the emitter assigns one automatically.

Note: Submitting an event to an emitter does not guarantee that the event is sent to the event server, because the filter settings might cause the event to be discarded. A successful call to `sendEvent()` means only that the event was successfully processed by the emitter.

The following example overrides the emitter setting to send an event in a new transaction, but does not override the synchronization mode:

```
String eventId = sendEvent(event,
                          SynchronizationMode.DEFAULT,
                          TransactionMode.NEW);
```

Sending multiple events:

If your event source needs to send multiple events in a batch, you can improve performance by sending them with a single call to the `sendEvents()` method.

Batching events in this way can also be useful for logical groups of events that should be sent only if an underlying transaction has completed successfully. All of the submitted events are sent as part of a single transaction.

- To send multiple events with the current emitter settings, use the `sendEvents(CommonBaseEvent[])` method:
`String[] eventIds = emitter.sendEvents(events);`
In this example, `emitter` is an `Emitter` instance, and `events` is an array of `CommonBaseEvent` instances.
- To send multiple events and override the current emitter settings, use the `sendEvents(CommonBaseEvent, int, int)` method, specifying the synchronization mode and transaction mode you want to use:

```
String eventId = emitter.sendEvent(event,
                                  synchronizationMode,
                                  transactionMode);
```

The returned value, `eventIds`, is an array containing the globally unique identifiers of the sent events.

Each event is validated and checked against the current filter criteria. All of the valid events that pass the filter criteria are then sent using the appropriate mechanism:

- If you are using synchronous event transport, the events are sent using a single EJB call. If an error occurs during the EJB call, an exception is thrown and none of the events are sent.
- If you are using asynchronous event transport, all of the events are sent using a single JMS message. If an error occurs during JMS processing, an exception is thrown and none of the events are sent.

Changing the emitter settings:

The settings in an emitter factory profile can be changed by an event source.

An event source can make changes to the transaction mode and synchronization mode configured for the emitter. These settings are initially defined by the emitter

factory profile. In addition, an event source can query the current transaction mode to determine what setting is currently in effect for the emitter.

Changing the synchronization mode:

An event source can change the synchronization mode being used by an emitter.

This change remains in effect for subsequent event submissions, but it does not change the preferred synchronization mode defined in the emitter factory profile.

The synchronization modes supported by an emitter are defined by the emitter factory profile. You can find out which modes are supported by a particular emitter by calling the `isSynchronizationModeSupported()` method; see the Javadoc API documentation for `com.ibm.events.emitter.Emitter` for more information. If you attempt to use a mode that is not supported, the emitter throws a `SynchronizationModeNotSupportedException` exception.

To change the synchronization mode, use the `setSynchronizationMode(int)` method.
`emitter.setSynchronizationMode(synchronizationMode);`

The *synchronizationMode* is an integer constant defined by the interface `SynchronizationMode`:

- `SynchronizationMode.ASYNCHRONOUS` (send the event asynchronously)
- `SynchronizationMode.SYNCHRONOUS` (send the event synchronously)
- `SynchronizationMode.DEFAULT` (send the event using the current emitter settings)

Changing the transaction mode:

An event source can change the transaction mode being used by an emitter.

This transaction mode change remains in effect for subsequent event submissions, but it does not change the transaction mode defined in the emitter factory profile.

Note: Transactions are supported only in a J2EE container.

To change the transaction mode, use the `setTransactionMode(int)` method.
`emitter.setTransactionMode(transactionMode);`

The *transactionMode* is an integer constant defined by the interface `TransactionMode`:

- `TransactionMode.NEW` (send the event in a new transaction)
- `TransactionMode.SAME` (send the event in the current transaction)
- `TransactionMode.DEFAULT` (send the event using the current emitter settings)

Querying the transaction mode:

An event source can query the transaction mode being used by an emitter.

Note: Transactions are supported only in a J2EE container.

To query the current transaction mode, use the `getTransactionMode()` method.
`int transactionMode = emitter.getTransactionMode();`

The returned value is an integer corresponding to one of the transaction mode constants:

- TransactionMode.NEW
- TransactionMode.SAME

Freeing emitter resources:

If your event source has finished sending events with a particular emitter, you should free the resources the emitter is using.

To free the emitter resources, use the `close()` method:

```
emitter.close();
```

This method releases all resources being used by the emitter.

Filtering events:

An emitter can optionally be configured to filter events at the source.

Event filtering provides a mechanism for reducing event traffic by screening out events that are not important. Each time an event source submits an event to an emitter, the emitter checks the event against the current filter criteria. If the event passes the filter criteria, the emitter sends the event to the event server; otherwise, the emitter discards the event. In any case, an event source cannot change the filter settings, which are configured by an administrator.

The emitter filter is implemented as a separate component called a *filter plug-in*. The Common Event Infrastructure includes a default filter plug-in, which provides filtering of submitted events based on XPath event selectors. If you want to use a different filter mechanism, you can implement your own filter plug-in.

In the Common Event Infrastructure configuration, each emitter factory is associated with a *filter factory*. A filter factory is an object used to create instances of a filter plug-in. When you create an emitter using an emitter factory, the emitter is automatically associated with an instance of the specified filter plug-in, which provides filtering of events submitted to that emitter.

Filtering events with the default filter plug-in:

A default emitter filter plug-in is included with the Common Event Infrastructure.

The Common Event Infrastructure includes a default emitter filter plug-in, which can be configured with an XPath event selector defining which events are sent to the event server and which are discarded. For example, the filter settings might specify that only events with severity greater than 20 (harmless) should be sent.

To filter events using the default filter plug-in, follow these steps:

1. In the WebSphere administrative console, navigate to the **Common Event Infrastructure Provider > Filter Factory Profile** page.
2. Create a new filter factory profile. For more information, see the online help for the administrative console.
3. In the **Filter Configuration String** field, type an XPath event selector describing events you want to use for filtering events. Events matching this event selector are sent to the event server; events that do not match are discarded by the emitter.

4. Navigate to the **Common Event Infrastructure Provider > Emitter Factory Profile** page.
5. Create a new emitter factory profile, or go to an existing emitter factory profile. For more information, see the online help for the administrative console.
6. In the **Filter Factory JNDI Name** field, type the JNDI name of the new filter factory profile you created.

Event sources can now use the new emitter factory to create instances of an emitter using the new filter configuration. If you later want to adjust the filter settings for event sources using that emitter factory, you can modify the event selector specified in the filter factory.

You can use tracing to find out what events are being discarded by the default filter plug-in; for more information, see *“Logging and tracing in the Common Event Infrastructure” on page 243*.

Note: The default filter plug-in uses the Apache XPath component to process XPath event selectors. If Java 2 security is enabled, you must modify your policy file to include entries allowing the required access:

```
permission java.util.PropertyPermission "*" , "read";
permission java.io.FilePermission
    "${was.install.root}${/}java${/}jre${/}lib${/}jxpath.properties",
    "read";
```

Implementing a filter plug-in:

You can implement a custom filter plug-in.

If you want to use your own filtering engine as an emitter filter, you can implement a custom filter plug-in by following these steps:

1. Develop your filter plug-in as a Java class implementing the interface `com.ibm.events.filter.Filter`. This interface defines the following methods:

isEnabled(CommonBaseEvent)

Returns a boolean value indicating whether the specified event passes the filter criteria. Each time an event is submitted to an emitter, the emitter calls this method, passing the submitted event. If the return value is `true`, the emitter sends the event to the event server for persistence and distribution. If the return value is `false`, the emitter discards the event.

getMetaData()

Returns information about the filter plug-in, such as the provider name and version number.

close() Frees all resources used by the filter plug-in. This method is called when the `close()` method of an emitter is called.

2. Develop a filter factory class that implements the interface `com.ibm.events.filter.FilterFactory`. This interface defines a single method, `getFilter()`, which returns an instance of your filter class (an implementation of the `Filter` interface).
3. Bind an instance of your filter factory into a JNDI namespace. During initialization, an emitter performs a JNDI lookup to access the filter factory.
4. In the WebSphere Process Server administrative console, modify your emitter factory profile or create a new profile. In the **Filter Factory JNDI Name** field,

specify the JNDI name of your FilterFactory implementation. For more information about emitter factory profiles, see the online help for the administrative console.

When you create an emitter using the emitter factory profile that specifies your filter factory, the new emitter uses an instance of your filter implementation. You can now send events using the standard emitter interfaces, and your filter plug-in is used.

Creating and populating an event using the ECSEmitter class:

Through the Java Naming and Directory Interface, you can indirectly access the event factory.

If an event factory is bound into the Java Naming and Directory Interface namespace, you can access the event factory indirectly. You can use the `com.ibm.websphere.cem.ECSEmitter` class to create and populate a Common Base Event. This class provides the following methods:

- `createCommonBaseEvent` method. If you use this method, you need provide only the extension name and the situation properties for the Common Base Event. All other properties are set automatically.
- `addUserDataEvent` method. If you use this method, all of the mandatory properties are set automatically. The extension name is set to `ECS:UserDataEvent` and the situation is set to `ReportSituation`. You can set extended data elements for the Common Base Event by passing a set of properties.

You can create and populate a Common Base Event in one of the following ways.

- Use the `createCommonBaseEvent` method to create and populate an event.

The following code fragment starts a new event correlation sphere, `newECSID`, and then uses the `createCommonBaseEvent` method to create an event object:

```
ECSEmitter myEmitter = new ECSEmitter("JNDI Emitter Factory Name", "newECSID");
CommonBaseEvent myEvent = myEmitter.createCommonBaseEvent("myEventType");
// get situation object
Situation mySituation = myEvent.getSituation();
// set situation properties
mySituation.setCategoryName("ReportSituation");
mySituation.setReportSituation("EXTERNAL", "STATUS");
// add other information to the the event
// send the event
myEmitter.sendEvent(myEvent);
```

This example uses the constructor method of the `ECSEmitter` class to create an emitter, passing the JNDI name of an existing Common Event Infrastructure emitter and the identification of a new event correlation sphere.

The new emitter is then used to create a Common Base Event that contains a situation object that is accessed using the `getSituation` call. The `setCategoryName` and `setReportSituation` methods are used to set the mandatory data in the situation object to emit an event with a `ReportSituation`. To create other situation types, use different category names in the `setCategoryName` call and different setter method calls for the situation.

All other mandatory information is provided automatically by the runtime environment. If mandatory information is set explicitly in the Common Base Event, this information is not overwritten with the default information. The event is now valid and can be submitted to an emitter using the `sendEvent` method.

In an actual application, a useful event needs to include more information than is shown in this example, but these properties are the minimum required by the Common Base Event specification and the Common Event Infrastructure.

- Use the `addUserDataEvent` method to create and populate an event.

The following code fragment uses the `addUserDataEvent` method to create an event object in the current event correlation sphere.

```
ECSEmitter myEmitter = new ECSEmitter("JNDI Emitter Factory Name", null);
// prepare a set of user data properties
Properties myUserData = new Properties();
myUserData.setProperty("UserData1", "UserDataValue1");
myUserData.setProperty("UserData2", "UserDataValue2");
// create and send the event
myEmitter.addUserDataEvent(myUserData);
```

This example uses the constructor method of the `ECSEmitter` class to create an emitter, passing the JNDI name of an existing Common Event Infrastructure emitter. An event correlation sphere identifier is not passed (`null`) and therefore a new event correlation sphere is not started. If an event correlation sphere exists, the user data event is added to this correlation sphere.

A set of user data properties is then prepared. Name and value pairs are added to a property list.

The last step in the example creates and sends a Common Base Event using the `addUserDataEvent` method of the new emitter. The `extensionName` property of the new Common Base Event is set to `ECS:UserDataEvent`, the `situation` is set to `ReportSituation`, and all other mandatory information is provided automatically by the runtime environment.

Creating an event consumer

An *event consumer* is any application that receives events from the event server.

An *event consumer* might be an application that receives asynchronous event notifications, or it might be an application that queries and processes historical event data from the persistent data store. The event consumer receives events in the form of Java objects; it can then use the `CommonBaseEvent` interface to retrieve event property data, or convert the event to another supported format (such as XML) for forwarding to another application.

An event consumer can receive events in either of two ways:

- It can use the Java Message Service (JMS) interface to subscribe to a queue or topic, receiving event notifications asynchronously as JMS messages. This is the most efficient approach for an event consumer that needs to process new and changed events as they arrive at the event server.
- It can use the Event Access interface to query historical events from the persistent data store, retrieving the requested events synchronously. This is useful for startup processing; by querying the data store for historical events, an event consumer can determine current state information before beginning to receive new events through JMS.

In addition to receiving events, an event consumer can also modify events, delete events, and purge old events from the data store.

Using the Java Message Service interface:

Using the Java Message Service (JMS) interface, you can develop event consumers that receive event notifications asynchronously from JMS queues or topics.

An event consumer can be implemented as a standard Java class or as a Message-Driven Bean (MDB).

By using the JMS interface, you can implement your event consumer using standard Java tools and programming models, and you can avoid the performance disadvantages of directly querying the event data store. Instead of interacting with the Common Event Infrastructure directly, your event consumer subscribes to JMS destinations (queues and topics) and receives event notifications in the form of JMS messages.

The Common Event Infrastructure organizes events in event groups, which are logical collections of events defined in the Common Event Infrastructure configuration. A particular event consumer typically needs to receive only events from specific event groups.

The configuration profile for each event group associates that event group with one or more JMS destinations through which notifications related to that event group are distributed. The relationships between event groups and JMS destinations are as follows:

- An event group can be associated with multiple queues.
- An event group can be associated with only one topic. (Multiple event consumers can subscribe to the same topic, so publishing the same event group to more than one topic is redundant.)
- A JMS destination (queue or topic) should typically be associated with only one event group.

To receive messages from an event group, a JMS consumer subscribes to the appropriate destination. Each time an event matching the associated event group is created, modified, or deleted, a notification is delivered in the form of a JMS message containing an event notification. The content of the notification depends upon its type:

- For a new or modified event, the notification includes the complete event data, which can be converted into a `CommonBaseEvent` instance.
- For a deleted event, the notification includes the global instance identifier of the event that has been deleted.

In addition to the standard JMS interfaces, a JMS event consumer interacts with a facility called the Notification Helper. The Notification Helper translates between Common Event Infrastructure entities (events and event groups) and equivalent JMS entities (messages and destinations). The Notification Helper provides the following functions:

- It identifies the JMS topic or queues associated with a specified event group. Your event consumer can then use the appropriate destination to create subscriptions.
- It converts event notifications for new and changed events into instances of `CommonBaseEvent`.
- It can provide filtering of events at the consumer. Each Notification Helper can be associated with an event selector specifying which events should be returned to consumers. When a consumer uses the Notification Helper to convert an event notification into an event instance, the event instance is returned only if it matches the specified event selector.

Note: The notification helper uses the Apache XPath component to process XPath event selectors. If Java 2 security is enabled, you must modify your policy file to include the following entries:

```
permission java.util.PropertyPermission "*", "read";
permission java.io.FilePermission
    "${was.install.root}${/}java${/}jre${/}lib${/}jxpath.properties",
    "read";
```

Developing an event consumer as a message-driven bean (MDB):

A J2EE event consumer is implemented as a message-driven bean, which is associated with a JMS destination and connection factory at deployment time. To receive events, follow these steps:

1. Obtain a notification helper. A JMS event consumer uses a notification helper to identify JMS destinations associated with an event group, to convert received JMS messages into event notifications, and to perform filtering of received events. To obtain a notification helper, use a notification helper factory, which is an instance of NotificationHelperFactory that has been bound into a JNDI namespace. The following code fragment uses a notification helper factory to obtain a notification helper.

```
// Get notification helper factory from JNDI
InitialContext context = new InitialContext();
Object notificationHelperFactoryObject =
    context.lookup("com/ibm/events/NotificationHelperFactory");
NotificationHelperFactory nhFactory = (NotificationHelperFactory)
    PortableRemoteObject.narrow(notificationHelperFactoryObject,
        NotificationHelperFactory.class);
```

```
// Create notification helper
NotificationHelper notificationHelper =
    nhFactory.getNotificationHelper();
```

2. **Optional:** Specify the event selector. If you want to filter received events, you can use the setEventSelector() method to set an event selector on the notification helper. Your event consumer can then use the notification helper to check received events against the event selector. The following code fragment sets an event selector specifying events with severity greater than 30 (warning).
notificationHelper.setEventSelector("CommonBaseEvent[@severity > 30]");

3. Convert received messages into event notifications.

In the onMessage() method of your listener, use the notification helper to convert each received JMS message into an array containing an event notification. (If the event does not match the event selector specified on the Notification Helper, the array is empty.) An event notification is an instance of a class implementing the EventNotification interface.

```
public void onMessage(Message msg) {
    EventNotification[] notifications =
        notificationHelper.getEventNotifications(msg);
    // ...
}
```

4. Check the notification type and retrieve the event data as appropriate. Each event notification has a field representing the notification type (an integer whose value is one of the notification type constants defined by the NotificationHelper interface). Three notification types are currently supported:

Notification type	Description
CREATE_EVENT_NOTIFICATION_TYPE	A new event has been created in the event group associated with the destination. This means either that a new event has been sent, or that an existing event has changed so that it now matches the event group definition. The notification also contains the complete event data.
REMOVE_EVENT_NOTIFICATION_TYPE	An event stored in the event database has been removed from the event group associated with the destination. This means either that an event has been deleted from the event database, or that an existing event has changed so that it no longer matches the event group definition. The notification also contains the global instance identifier of the deleted event.
UPDATE_EVENT_NOTIFICATION_TYPE	An event stored in the event database has been updated in a way that does not change its membership in the event group associated with the destination. The notification also contains the complete event data.

Use the `getNotificationType()` method of `EventNotification` to check the notification type of each received notification. Based on the notification type, you can determine whether your event consumer should process the notification further, and what kind of event data the notification contains:

- If the notification type is `CREATE_EVENT_NOTIFICATION_TYPE` or `UPDATE_EVENT_NOTIFICATION_TYPE`, your consumer can use `EventNotification.getEvent()` to attempt to retrieve the new or updated event. This method is valid only for notifications of new or updated events.
- If the notification is `REMOVE_EVENT_NOTIFICATION_TYPE`, your consumer can use `EventNotification.getGlobalInstanceId()` to retrieve the global instance identifier of the deleted event. This method is valid only for notifications of deleted events.

```
for (int i = 0; i < notifications.length; i++)
{
    int notifType = notifications[i].getNotificationType();

    if(notifType == NotificationHelper.CREATE_EVENT_NOTIFICATION_TYPE)
    {
        CommonBaseEvent event = notifications[i].getEvent();
        if (event != null) {
            // process the new event
            // ...
        }
    }

    else if(notifType == NotificationHelper.UPDATE_EVENT_NOTIFICATION_TYPE)
    {
        CommonBaseEvent event = notifications[i].getEvent();
        if (event != null) {
            // process the updated event
            // ...
        }
    }

    else if(notifType == NotificationHelper.REMOVE_EVENT_NOTIFICATION_TYPE)
    {
        String eventId = notifications.[i].getGlobalInstanceId();
    }
}
```

```

        // process the event deletion
        // ...
    }
}

```

In its deployment descriptor, a message-driven bean must be associated with a listener port, which specifies a JMS destination and connection factory. You must create a listener port for your event consumer before deploying the MDB, specifying the destination and connection factory associated with the event group from which you want to receive events (these are defined in the event group profile).

Note: Do not use the `CommonEventInfrastructure_ListenerPort` listener port when deploying your MDB. This listener port is used by the event server and is not intended for use by event consumers.

Developing a non-MDB event consumer:

An event consumer can also be created using a non-message driven bean.

To write an event consumer that is not a message-driven bean, follow these steps:

1. Obtain a notification helper. A JMS event consumer uses a notification helper to identify JMS destinations associated with an event group, to convert received JMS messages into event notifications, and to perform filtering of received events. To obtain a notification helper, use a notification helper factory, which is an instance of `NotificationHelperFactory` that has been bound into a JNDI namespace. The following code fragment uses a notification helper factory to obtain a notification helper.

```

// Get notification helper factory from JNDI
InitialContext context = new InitialContext();
Object notificationHelperFactoryObject =
    context.lookup("com/ibm/events/NotificationHelperFactory");
NotificationHelperFactory nhFactory = (NotificationHelperFactory)
    PortableRemoteObject.narrow(notificationHelperFactoryObject,
        NotificationHelperFactory.class);

// Create notification helper
NotificationHelper notificationHelper =
    nhFactory.getNotificationHelper();

```

2. **Optional:** Specify the event selector. If you want to filter received events, you can use the `setEventSelector()` method to set an event selector on the notification helper. Your event consumer can then use the notification helper to check received events against the event selector. The following code fragment sets an event selector specifying events with severity greater than 30 (warning).

```
notificationHelper.setEventSelector("CommonBaseEvent[@severity > 30]");
```

3. Use the notification helper to find the JMS destination to subscribe to.

Each event group can be associated with a single JMS topic and any number of JMS queues. You can query the notification helper to find out what destinations are associated with a particular event group. To find the topic associated with an event group, use the `getJmsTopic(String)` method of `NotificationHelper`, specifying the name of the event group:

```
MessagePort msgPort = notificationHelper.getJmsTopic("critical_events");
```

To find the queues associated with an event group, use the `getJmsQueues(String)` method:

```
MessagePort[] msgPorts = notificationHelper.getJmsQueues("critical_events");
```

The returned object is either a single MessagePort object representing a JMS topic or an array of MessagePort objects representing JMS queues. A MessagePort instance is a wrapper object containing the JNDI names of the destination and its connection factory.

4. Connect to the destination. Use the getter methods of MessagePort to retrieve the JNDI names of the destination and connection factory. You can then use the standard JMS interfaces to connect to the destination. The following code fragment subscribes to a JMS topic:

```
String connectionFactoryName = msgPort.getConnectionFactoryJndiName();
String destinationName = msgPort.getDestinationJndiName();

// create connection and session
ConnectionFactory connectionFactory =
    (ConnectionFactory) context.lookup(connectionFactoryName);
Connection connection = connectionFactory.createConnection();
Session session = connection.createSession(false,
    Session.CLIENT_ACKNOWLEDGE);

// Create consumer and register listener
Topic topic = (Topic) context.lookup(destinationName);
MessageConsumer consumer = session.createConsumer(topic);
consumer.setMessageListener(this);
connection.start();
```

5. Convert received messages into event notifications.

In the onMessage() method of your listener, use the notification helper to convert each received JMS message into an array containing an event notification. (If the event does not match the event selector specified on the Notification Helper, the array is empty.) An event notification is an instance of a class implementing the EventNotification interface.

```
public void onMessage(Message msg) {
    EventNotification[] notifications =
        notificationHelper.getEventNotifications(msg);
    // ...
}
```

6. Check the notification type and retrieve the event data as appropriate. Each event notification has a field representing the notification type (an integer whose value is one of the notification type constants defined by the NotificationHelper interface). Three notification types are currently supported:

Notification type	Description
CREATE_EVENT _NOTIFICATION_TYPE	A new event has been created in the event group associated with the destination. This means either that a new event has been sent, or that an existing event has changed so that it now matches the event group definition. The notification also contains the complete event data.
REMOVE_EVENT _NOTIFICATION_TYPE	An event stored in the event database has been removed from the event group associated with the destination. This means either that an event has been deleted from the event database, or that an existing event has changed so that it no longer matches the event group definition. The notification also contains the global instance identifier of the deleted event.
UPDATE_EVENT _NOTIFICATION_TYPE	An event stored in the event database has been updated in a way that does not change its membership in the event group associated with the destination. The notification also contains the complete event data.

Use the `getNotificationType()` method of `EventNotification` to check the notification type of each received notification. Based on the notification type, you can determine whether your event consumer should process the notification further, and what kind of event data the notification contains:

- If the notification type is `CREATE_EVENT_NOTIFICATION_TYPE` or `UPDATE_EVENT_NOTIFICATION_TYPE`, your consumer can use `EventNotification.getEvent()` to attempt to retrieve the new or updated event. This method is valid only for notifications of new or updated events.
- If the notification is `REMOVE_EVENT_NOTIFICATION_TYPE`, your consumer can use `EventNotification.getGlobalInstanceId()` to retrieve the global instance identifier of the deleted event. This method is valid only for notifications of deleted events.

```
for (int i = 0; i < notifications.length; i++)
{
    int notifType = notifications[i].getNotificationType();

    if(notifType == NotificationHelper.CREATE_EVENT_NOTIFICATION_TYPE)
    {
        CommonBaseEvent event = notifications[i].getEvent();
        if (event != null) {
            // process the new event
            // ...
        }
    }

    else if(notifType == NotificationHelper.UPDATE_EVENT_NOTIFICATION_TYPE)
    {
        CommonBaseEvent event = notifications[i].getEvent();
        if (event != null) {
            // process the updated event
            // ...
        }
    }

    else if(notifType == NotificationHelper.REMOVE_EVENT_NOTIFICATION_TYPE)
    {
        String eventId = notifications.[i].getGlobalInstanceId();
        // process the event deletion
        // ...
    }
}
```

Querying events from the event server:

An event consumer can synchronously retrieve historical events from the persistent data store by querying the event server.

The persistent data store is implemented as a separate component called a *data store plug-in*. The Common Event Infrastructure includes a default data store plug-in, which supports event queries based on a subset of XPath syntax. If you want to use a different data store, you can implement your own data store plug-in.

To query the event server, use the event access interface.

1. Create an event access bean.

This bean is a Java 2 Platform, Enterprise Edition (J2EE) stateless session bean. The bean interface provides methods for querying the event server. An event consumer uses an instance of the event access bean for all synchronous event queries.

2. Query events.

You can query events in the following ways:

- Specify a global instance identifier to retrieve a specific single event.
- Specify an event group to retrieve events associated with that event group. You can optionally refine the query by specifying an additional event selector. This action retrieves only those events that match both the event group and the event selector.
- Specify a known event and an association type to retrieve events that are associated with the known event.
- Query and purge events.

Creating an event access bean:

Getting event access starts with creating an instance of the event access bean.

The event access interface is implemented as a stateless session bean using the Enterprise JavaBeans architecture. To query the event server using the event access interface, an event source must first create an instance of the event access session bean. The event access bean can be either local or remote.

To create an instance of the event access session bean, use the appropriate home interface: `com.ibm.events.access.EventAccessHome` or `com.ibm.events.access.EventAccessLocalHome`.

```
// use home interface to create remote event access bean
InitialContext context = new InitialContext();
Object eventAccessHomeObj = context.lookup("ejb/com/ibm/events/access/EventAccess");
EventAccessHome eventAccessHome = (EventAccessHome)
    PortableRemoteObject.narrow(eventAccessHomeObj,
        EventAccessHome.class);
eventAccess = (EventAccess) eventAccessHome.create();
```

Querying events by global instance identifier:

Events can be queried by a primary key.

The Common Base Event specification defines a `globalInstanceId` event property that can be used as a primary key for event identification. The content of this property is a globally unique identifier that is generated either by the application or by the emitter. Although the Common Base Event specification defines the `globalInstanceId` property as optional, the event emitter automatically assigns an identifier to any event that does not already have an identifier.

You can retrieve a specific single event from the event server by querying with the `globalInstanceId` property of the event that you want to retrieve. This query can be useful for testing purposes (to confirm that events are stored in the event database), or to retrieve an event associated with one that was received previously.

To query an event by the global instance identifier, use the `queryEventByGlobalInstanceId` method of the event access bean.

1. **Optional:** Create an event access bean.
2. Call the `queryEventByGlobalInstanceId(String)` method of the `EventAccess` bean, specifying the global instance identifier of the event that you want to retrieve.

```
CommonBaseEvent event = eventAccess.queryEventByGlobalInstanceId(eventId);
```

The returned object is the event with the specified global instance identifier. If there is no matching event in the persistent data store, the returned object is null.

Querying events by event group:

To make specified events easier to access, they can be associated with event groups.

You can associate an event with one or more *event groups*. An event group is a logical grouping of events that match a particular event selector. Event groups are defined in the event infrastructure configuration. For more information about event groups, see *Default configuration*.

You can use the event access interface to retrieve events that belong to a specified event group. You can restrict the query results by specifying an additional event selector. You can also query events without retrieving them.

You can query event groups in the following ways:

Querying a limited number of events from an event group:

You can query a limited number of events from an event group.

To query a limited number of events from an event group, use the `queryEventsByEventGroup(String, String, Boolean, int)` method of the `EventAccess` bean.

1. If you need to create an event access bean, see *“Creating an event access bean”* on page 270.
2. Call the `EventAccess.queryEventsByEventGroup(String, String, boolean, int)` method.

```
CommonBaseEvent[] events = eventAccess.queryEventsByEventGroup(eventGroup,
                                                                eventSelector,
                                                                ascendingOrder,
                                                                maxEvents);
```

The parameters of this method are as follows:

eventGroup

A string containing the name of the event group you want to query events from. This must be the name of an existing event group defined in the event infrastructure configuration.

eventSelector

A string containing an optional event selector that further refines the query. The query returns only events that match both the specified event group and the additional event selector. An event selector is specified in the form of an XPath expression (for more information, see *“Writing event selectors”* on page 275). If you do not want to specify an additional event selector, this parameter can be null.

ascendingOrder

A boolean value specifying whether the returned events are to be sorted in ascending or descending order according to the value of the `creationTime` property. If this parameter is *true*, the events are sorted in ascending (chronological) order; if it is *false*, they are sorted in descending (reverse chronological) order.

maxEvents

An integer specifying the maximum number of events you want returned. The returned object is an array containing the events from the specified event group.

Note: If the number of matching events exceeds the query threshold defined in the data store profile, a `QueryThresholdExceededException` exception is thrown. The default query threshold is 100 000.

The following code fragment returns all events that belong to an event group called *critical_hosts* and whose severity is greater than 30 (warning), but specifies that no more than 5000 matching events should be returned:

```
CommonBaseEvent[] events =
    eventAccess.queryByEventGroup("critical_hosts",
                                "CommonBaseEvent[@severity > 30]",
                                true,
                                5000);
```

Querying all events from an event group:

To query all events from an event group, use the `queryEventsByEventGroup(String, String, boolean)` method of the `EventAccess` bean.

1. If you need to create an event access bean, see “*Creating an event access bean*” on page 270.
2. Call the `EventAccess.queryEventsByEventGroup(String, String, boolean)` method.

```
CommonBaseEvent[] events = eventAccess.queryEventsByEventGroup(eventGroup,
                                                                eventSelector,
                                                                ascendingOrder);
```

The parameters of this method are as follows:

eventGroup

A string containing the name of the event group you want to query events from. This must be the name of an existing event group defined in the event infrastructure configuration.

eventSelector

A string containing an optional event selector that further refines the query. The query returns only events that match both the specified event group and the additional event selector. An event selector is specified in the form of an XPath expression (for more information, see “*Writing event selectors*” on page 275). If you do not want to specify an additional event selector, this parameter can be null.

ascendingOrder

A boolean value specifying whether the returned events are to be sorted in ascending or descending order according to the value of the `creationTime` property. If this parameter is *true*, the events are sorted in ascending (chronological) order; if it is *false*, they are sorted in descending (reverse chronological) order.

The returned object is an array containing the events from the specified event group.

Note: If the number of matching events exceeds the query threshold defined in the data store profile, a `QueryThresholdExceededException` exception is thrown. The default query threshold is 100 000.

The following code fragment returns all events that belong to an event group called *critical_hosts* and whose severity is greater than 30 (warning):

```
CommonBaseEvent[] events =
    eventAccess.queryByEventGroup("critical_hosts",
                                "CommonBaseEvent[@severity > 30]",
                                true);
```

Querying the existence of events in an event group:

You can determine the existence of events without retrieving them.

In some situations, you might want to find out whether any events exist in a particular event group without actually retrieving the events. To do this, use the `eventExists()` method of the event access bean.

1. If you need to create an event access bean, see “*Creating an event access bean*” on page 270
2. Call the `eventExists(String, String)` method of the `EventAccess` bean.

```
boolean hasEvents = eventAccess.eventExists(eventGroup,
                                             eventSelector);
```

The parameters of this method are as follows:

eventGroup

A string containing the name of the event group you want to check for events. This must be the name of an existing event group defined in the event infrastructure configuration.

eventSelector

A string containing an optional event selector that further refines the query. The query only checks for events that match both the specified event group and the additional event selector. An event selector is specified in the form of an XPath expression (for more information, see “*Writing event selectors*” on page 275). If you do not want to specify an additional event selector, this parameter can be null.

The returned boolean object equals `true` if any events exist that match the specified event group and event selector, `false` if none exist.

The following code fragment checks for the existence of any events in an event group called *critical_hosts* and retrieves any that exist.

```
if (eventAccess.eventExists("critical_hosts",null)) {
    CommonBaseEvent[] events =
        eventAccess.queryByEventGroup("critical_hosts",
                                    null,
                                    true);
}
```

Querying events by association type:

Events can be queried by association types.

The Common Base Event specification defines properties that establish relationships between events. The `associatedEvents` property is a complex element that contains one or more subelements of the `AssociatedEvent` type, each representing an associated event. Each `AssociatedEvent` element, contains subelements that identify the type of association and the application that established the association. Examples of association types might include `CausedBy` or `Correlated`.

By specifying the global instance identifier of a known event and a type of association, you can retrieve events that satisfy the specified association. To query events by association type, use the `EventAccess.queryEventsByAssociation(String, String)` method.

1. **Optional:** Create an event access bean.
2. Call the `EventAccess.queryEventsByAssociation(String, String)` method.

```
CommonBaseEvent[] events = eventAccess.queryEventsByAssociation(associationType,
                                                                eventId);
```

The parameters of this method are as follows:

associationType

The type of association. This is the name of an association type specified by the `associationEngineInfo` property.

eventId

The global instance identifier of a known event.

The returned object is an array that contains the events that satisfy the specified type of association with the known event. Only events that are still in the event database at the time of the query are returned (an associated event might be purged from the database).

The following code fragment returns all of the events from the event database that have a `CausedBy` association with a known event:

```
String eventId = causeEvent.getGlobalInstanceId();
CommonBaseEvent[] resultEvents = eventAccess.queryEventsByAssociation("CausedBy",
                                                                    eventId);
```

Deleting events from the data store:

An event consumer or administrative tool can delete events from the data store using the event access interface.

You can delete all events from the data store, or you can limit the deleted events by specifying event groups, event selectors, or both.

Note: If WebSphere security is enabled, the application user ID must be mapped to the `eventAdministrator` role to delete events.

To delete events from the data store, use the `purgeEvents()` method of the event access bean.

```
int purged = eventAccess.purgeEvents(eventGroup,
                                    eventSelector,
                                    transactionSize);
```

The parameters are as follows:

eventGroup

A string containing the name of the event group that includes the events you want to delete. This must be the name of an existing event group defined in the event infrastructure configuration. If you do not want to specify an event group, this parameter can be null.

eventSelector

A string containing an optional event selector that identifies the events to delete. An event selector is specified in the form of an XPath expression (for more information, see *Writing event selectors* on page 275). If you do not want to specify an event selector, this parameter can be null.

transactionSize

A nonzero integer specifying the number of events you want deleted in a single database transaction. In most cases, you can use the constant `DEFAULT_PURGE_TRANSACTION_SIZE`, which is defined by the `EventAccess` interface.

The `purgeEvents()` method deletes all of the events that match all of the criteria you specify. If the *eventGroup* and *eventSelector* parameters are both null, all events in the data store are deleted. Events that arrive after the delete operation starts are not purged. The returned value is an integer specifying how many events were deleted.

Note: If the value of the *transactionSize* parameter exceeds the maximum purge transaction size defined in the data store profile, a `PurgeThresholdExceededException` exception is thrown and no events are deleted. The default maximum purge transaction size is 100 000.

Writing event selectors:

An event selector defines a set of events.

An event selector is a regular expression that defines a set of events based on their property data (attributes or sub-elements). For example, an event selector might specify all events from a particular host whose severity is greater than 30 (warning). Event selectors are used to define event groups, specify filter criteria, and query the event server.

Because the Common Base Event specification is based on XML, event selectors are written using a subset of XPath syntax. The specific syntax you can use for an event selector depends on how the event selector is to be used, as summarized by the following table.

Event selector purpose	Syntax
Event group definition	Limited to XPath subset supported by default data store plug-in
Event query and purge through event access interface	Limited to XPath subset supported by default data store plug-in
Emitter filter configuration	Any valid XPath
Subscription through Notification Helper interface	Any valid XPath

Note: The default data store plug-in uses a subset of XPath syntax. However, if you are using a different data store plug-in, it might support a different subset of XPath. The event selectors you write for event group definitions and for the event access interface must use the syntax that is supported by your data store plug-in.

Writing XPath event selectors

XPath is a standard language used to identify parts of an XML document; for more information, see the XPath specification at <http://www.w3.org/TR/xpath>.

A simple XPath event selector that specifies an attribute value takes the following form:

CommonBaseEvent[@attribute = value]

The *value* can be either a numeric value or a string enclosed in single or double quotation marks.

You can also specify an attribute of a subelement:

CommonBaseEvent[/subelement/@attribute = value]

When using XPath operators, keep the following general rules in mind:

- When used to compare XML dateTime values, the comparison operators perform logical comparisons that recognize time zone differences.
- Logical operators and function names must be specified using all lowercase letters (for example, and rather than AND).
- Operators must be separated with white space from the surrounding attribute names and values (@severity > 30 rather than @severity>30).
- Parentheses can be used to change operator precedence.

The following examples are valid XPath event selectors.

XPath event selectors	Selector definition
CommonBaseEvent[@extensionName = 'ApplicationStarted']	All events with the <i>extensionName</i> attribute ApplicationStarted
CommonBaseEvent[sourceComponentId/@location = "server1"]	All events containing a sourceComponentId element with the <i>location</i> attribute server1
CommonBaseEvent[@severity]	All events with a <i>severity</i> attribute, regardless of its value
CommonBaseEvent[@creationTime < '2003-12-10T12:00:00-05:00' and @severity > 30]	All events created before noon EST on 10 December 2003 and with severity greater than 30 (warning):
CommonBaseEvent[contains(@msg, 'disk full')]	All events with the phrase disk full occurring within the msg attribute
CommonBaseEvent[(@severity = 30 or @severity = 50) and @priority = 100]	All events whose <i>severity</i> attribute is equal to 30 or 50, and whose <i>priority</i> is equal to 100.

Writing event selectors for the default data store plug-in

If your event selector might be used to define an event group or to query the persistent data store, it is subject to the restrictions of the default data store plug-in. These restrictions are as follows:

- An event property can be specified only on the left side of an operator or XPath function. The value on the right side of an operator must be a literal value. The following example is not a valid event selector:

```
CommonBaseEvent[30 < @priority and  
contains('this message', @msg)]
```

Instead, this could be rewritten as follows:

```
CommonBaseEvent[@priority > 30 and  
contains(@msg, 'this message')]
```

- Only the following XPath functions are supported:
 - contains
 - starts-with

- false
- true
- not
- The union operator (|) is not supported.
- An event selector must take the following form:

```
CommonBaseEvent[predicate_expression]
```

Only a single predicate expression can be associated with the CommonBaseEvent element. Stacked predicates are not supported (for example, CommonBaseEvent[@extensionName = "server_down"][@severity = 10]).

- A predicate can be only be associated with the last step of a location path. The following example is not a valid event selector:

```
CommonBaseEvent[contextDataElement[@contextValue = "myContextValue"]
                /@contextId = "myContextId"]
```

Instead, this could be rewritten as follows:

```
CommonBaseEvent[contextDataElement[@contextValue = "myContextValue"
                and @contextId = "myContextId"]]
```

- If an event selector refers to properties of extended data elements that are at different levels of the XML containment hierarchy, these elements must be grouped together by level. The following example is not a valid event selector, because the references to the *type* and *value* attributes (both top-level) of *extendedDataElements* are separated:

```
CommonBaseEvent[extendedDataElements[@type = 'int' and
                children/@type = 'intArray' and
                children/@name = 'myName' and
                @value = 10]]
```

Instead, this could be rewritten as follows, grouping the top-level and second-level attributes together:

```
CommonBaseEvent[extendedDataElements[@type = 'int' and
                @value = 10 and
                children/@type = 'intArray' and
                children/@name = 'myName']]
```

- Node indexes are not supported (for example, CommonBaseEvent[extendedDataElements[1]]).
- Wildcard characters are not supported (for example, CommonBaseEvent[extendedDataElements/*/children/values = "text"]).
- When referring to the *values* property of an extended data element, you must specify not only the value but also the type of the property:

```
CommonBaseEvent[extendedDataElements[values = "myVal"
                and @type = "string"]]
```

You can specify the type for multiple comparisons within a compound expression by grouping them with parentheses:

```
CommonBaseEvent[extendedDataElements[(values = "myVal" or
                values = "yourVal") and
                @type = "string"]]
```

In this example, the *type* expression applies to both parts of the compound expression in parentheses. You cannot override this by specifying a different *type* expression inside the parentheses.

You can also group together multiple related types by using the starts-with or contains functions. For example, the following expression would match a property with either the string or stringArray type:

```
CommonBaseEvent[extendedDataElements[values = "myVal" and
                                         starts-with(@type, 'string')]]
```

Creating an event catalog application

The *event catalog* is a repository of event metadata.

The *event catalog* metadata consists of event definitions, which describe classes of events and their allowed content. (This is distinct from the event instance metadata you can access using the Eclipse Modeling Framework interfaces described in “*Accessing event instance metadata*” on page 253.) Applications can use the event catalog to manage their enterprise-specific event definitions, but must implement their own validation logic to ensure that events conform to these definitions.

Events defined according to the Common Base Event specification can be categorized into event classes based upon extension name (the value of the *extensionName* attribute). Using the event catalog, you can define the permitted content of a particular class of event by specifying what extended data elements events of that class can contain, as well as the permitted values for other Common Base Event properties. An event definition defines constraints on event content above and beyond those of the Common Base Event specification.

Event definitions are defined hierarchically and inherit the definitions of their parents. A single root event definition, *event*, defines the basic requirements of any event that conforms to the Common Base Event specification. All other event definitions inherit from this root definition. By default, this root event definition is automatically installed in the Event Catalog, along with event definitions for Event Catalog notification events (for more information, see “*Change notification*” on page 283).

Note: Currently, event definitions do not support all of the forms of constraints required to fully describe the Common Base Event specification (for example, the requirement that the *globalInstanceId* property must begin with an alphabetic character). Therefore, it is possible that an event might conform to the event definition and still not pass validation by the event emitter.

By using the event catalog interfaces, you can create, delete, and query event definitions. (Once created, an event definition cannot be modified.) You can also list existing event definitions in a readable format, as well as importing and exporting event definitions in XML format.

Event definitions:

Event definitions are comprised of several types of information.

An *event definition* contains several kinds of information:

Name The name of the event definition, which is the same as the extension name of the events described by the definition. All events with a particular extension name share the same event definition.

Parent The name of the parent event definition. Any event definition (with the exception of the root definition event) has a parent event definition from which it inherits property descriptions and extended data element descriptions (although some aspects of the inherited data can be overridden). The parent can be any valid event definition that exists in the event catalog.

Property descriptions

Descriptions of the permitted Common Base Event properties for the event definition. A property description can describe any property defined in the Common Base Event specification as a simple type, including properties of complex subelements.

Extended data element descriptions

Descriptions of the permitted extended data elements for the event definition. An extended data element description defines the name and type of the extended data element; it can also define default values, how many of the extended data element are allowed, and descriptions of child extended data elements.

Represented as an XML document, an event definition takes the following general form:

```
<eventDefinition name="eventDefinitionName"
                 parent="parentEventDefinitionName">
  <property name="propertyName" ... />
  <extendedDataElement name="extendedDataElementName"
                      type="type" ... />
</eventDefinition>
```

Property descriptions:

Property descriptions are defined by the Common Base Event specification.

A property description describes a property that an event can contain. This can be any property defined by the Common Base Event specification as a simple type. A property description cannot describe a complex property such as *msgDataElement*, but it can describe a simple property that is a child of a complex property. An event definition can contain any number of property descriptions (including none).

A property description includes the following fields:

name The name of the property. This must be the name of an attribute of the *CommonBaseEvent* element, or an attribute of a complex subelement of *CommonBaseEvent*. Some examples are *severity*, *priority*, and *globalInstanceId*.

path An XPath location path specifying the path to the property, if the property is not an attribute of *CommonBaseEvent*. The path identifies the parent property of the property being described. These are examples:

- To describe a property of *CommonBaseEvent* such as *severity*, do not specify a path. A null path specifies a top-level property.
- To describe a property of *msgDataElement*, which is a complex property of *CommonBaseEvent*, you specify the path *msgDataElement*.
- To describe a property of *msgHelp*, which is itself a complex property of *msgDataElement*, specify the path *msgDataElement/msgHelp*.

The path can also describe a specific instance of a repeated property. For example, if an event definition describes several *contextDataElements* properties, you might specify one called *businessContext*, you would use the path *contextDataElements[@name='businessContext']*.

defaultValue

The default value of the property. The default value represents the value that should be used during content completion for an event that is missing

a required property. (Therefore, it is meaningful for a property description to be required and to define a default value.) This field is optional.

required

A boolean value specifying whether the property is required or optional. If this field is equal to true, the property is required. This field is optional; if it is not specified, the property is assumed to be optional.

permittedValue

A permitted value for the property. If an event definition allows only certain values for a property, each one is represented by a *permittedValue* field in the property description. A property description can include any number of permitted values. This field is optional and must not be specified if the *minValue* or *maxValue* fields are specified.

minValue

maxValue

The minimum and maximum permitted values for the property. If an event definition allows a range of values for a property, these fields defines the lower and upper bounds of that range. If you specify only *minValue*, the permitted range has no upper bound; similarly, if you specify only *maxValue*, the permitted range has no lower bound. These fields are optional and must not be specified if *permittedValue* fields are specified.

Extended data element descriptions:

Extended data elements are one type of information an event definition may contain.

An extended data element description describes an extended data element that an event of a particular event class can contain. An event definition can contain any number of extended data element descriptions (including none).

An extended data element description includes the following fields:

name The name of the extended data element. This defines the value of the *name* attribute of the element.

type The data type of the extended data element. This defines the value of the *type* attribute of the element. This must be one of the following supported data types:

- noValue
- byte
- short
- int
- long
- float
- double
- string
- dateTime
- boolean
- byteArray
- shortArray
- intArray
- longArray

- floatArray
- doubleArray
- stringArray
- dateTimeArray
- booleanArray
- hexBinary

defaultValue

The default value of the extended data element, or multiple default values if the type is an array. The default value represents the value that should be used during content completion for an event that is missing a required extended data element. This field is optional.

minOccurs

The minimum number of instances of the extended data element that must appear. This field is optional; the default value is 1.

maxOccurs

The maximum number of instances of the extended data element that can appear. This field is optional; the default value is 1.

Note: The current Common Base Event specification allows only one extended data element with a given name at each level of the event containment hierarchy, but this restriction will not be included in future versions of the specification and is not enforced by the Common Event Infrastructure.

Event catalog inheritance:

Event definitions inherit the properties of their parents.

By default, an event definition inherits the property descriptions and extended data element descriptions of its parent. However, a child event definition can override these inherited descriptions, subject to certain restrictions. When you add an event definition to the event catalog, the catalog verifies that the new event definition does not violate the rules governing inheritance; if it does, an `InheritanceNotValidException` exception is thrown. Similarly, if you replace an existing event definition that has descendants, the event catalog verifies the validity of the existing inheritance relationships and throws an `InheritanceNotValidException` exception if any of them are no longer valid. In either case, the new event definition is not added to the catalog unless all inheritance relationships are valid.

An event definition can exist in either of two forms, *unresolved* and *resolved*:

- An unresolved event definition includes only those property definitions and extended data element descriptions that are defined within the event definition itself.
- A resolved event definition includes the data in the unresolved event definition in addition to the property definitions and extended data element descriptions it inherits.

Overriding inherited property descriptions

A child event definition inherits each property description from its parent without change unless it already has a locally defined property description of the same name and path (note that case is significant). If the child has a property description

of the same name and path, the fields of the child description can override the fields of the parent description as follows:

Default value

The child can override the default value specified by the parent property description. If the child does not specify a default value, it inherits the value from the parent.

Required or optional

The child always overrides the parent. However, if the parent defines a property as required, the child must also specify that it is required. An inherited required property cannot be redefined as optional.

Permitted values or minimum and maximum values

If the parent defines permitted values or minimum and maximum values, the child can override these by specifying either permitted values or minimum and maximum values. Note that an event definition can contain only permitted values or minimum and maximum values, not both:

- If the parent defines minimum and maximum values, but the child defines permitted values, the minimum and maximum values defined by the parent are ignored.
- If the parent defines permitted values, but the child defines minimum and maximum values, the permitted values defined by the parent are ignored.
- If the parent defines only a maximum value, but the child defines only a minimum value, the child inherits the maximum value defined by the parent.
- If the child does not specify permitted values or minimum and maximum values, the values specified by the parent are inherited.

Overriding inherited extended data element descriptions

A child event definition inherits each extended data element description from its parent without change unless it already has a locally defined extended data element description of the same name. If the child does have an extended data element description of the same name, the fields of the child description can override the fields of the parent description as follows:

Type The child must specify the same type as the parent.

Minimum occurrence

The child always overrides the parent.

Maximum occurrence

The child always overrides the parent.

Default values

The child can override the default values specified by the parent extended data element description. If the child does not specify default values, it inherits the values from the parent.

Default hexadecimal value

The child can override the default hexadecimal value specified by the parent extended data element description. If the child does not specify a default hexadecimal value, it inherits the value from the parent.

Nested extended data element description

The child can override a nested extended data element description by defining a nested description of the same name. If the child overrides an

inherited nested description, the same rules apply to overriding the individual fields. If the child does not specify a nested extended data element description of the same name, it inherits the nested description from the parent.

Change notification:

Each time an event definition is added, removed, or replaced, the event catalog sends an event to the event server indicating that this has happened.

An event consumer can subscribe to these events to receive notification of changes in the event catalog. By default, the event catalog uses the default emitter factory to obtain an emitter for sending these events; however, this can be changed in the Event Catalog configuration.

The event catalog can send three classes of notification events, using the following extension names:

- `cei_event_definition_added`
- `cei_event_definition_replaced`
- `cei_event_definition_removed`

These three event classes inherit property descriptions from a common parent class, `cei_event_definition`. Event definitions for all four event classes are automatically loaded into the event catalog during installation, along with the default root event definition.

Note: When an event definition is removed from the event catalog, any children or other descendants of that event definition are also removed. The event catalog sends a separate change notification event for each event definition that is removed.

Each change notification event contains the following properties:

Property	Value
<i>version</i>	1.0.1
<i>globalInstanceId</i>	A globally unique identifier for the event
<i>creationTime</i>	Current date and time when the event is generated
<i>severity</i>	10 (information)
<i>priority</i>	10 (low)
<i>sourceComponentId</i>	Identification of the Event Catalog component and event server host machine
<i>situation</i>	Situation data, including one of the following values for situation category: <ul style="list-style-type: none"> • <code>CreateSituation</code> (event definition added) • <code>ConfigureSituation</code> (event definition replaced) • <code>DestroySituation</code> (event definition removed)
<i>extensionName</i>	One of the following values: <ul style="list-style-type: none"> • <code>cei_event_definition_added</code> • <code>cei_event_definition_replaced</code> • <code>cei_event_definition_removed</code>

Property	Value
<i>extendedDataElements</i>	A single extended data element with one attribute, <i>eventDefinitionName</i> . This attribute is a string specifying the name of the event definition that has been added, replaced, or removed.

Creating an event definition:

Event definitions are instances of the EventDefinition class.

An event definition is an instance of the class EventDefinition. To create an event definition, first create a new instance of this class and then populate it with property descriptions and extended data element descriptions. After you have created an event definition, you can add it to the event catalog; for more information, see *"Adding an event definition to the event catalog"* on page 287.

To create a new, empty event definition, create an instance of EventDefinition:

```
EventDefinition definition = new EventDefinition(name, parent);
```

The parameters of this constructor are as follows:

name

The name of the event definition. This is the value of the *extensionName* attribute for the events you are describing.

parent

The name of the parent event definition. If you do not want your event definition to inherit any property descriptions or extended data element descriptions other than those required by the Common Base Event specification, this parameter should be event. If this parameter is null, the new event definition is defined as a root event definition; a root event definition can only be added to the catalog if it is empty, or if you intend to replace the current root event definition.

The returned object is a new unresolved event definition containing no property descriptions or extended data element descriptions.

The following code fragment creates a new event definition called *insurance_claim_start_auto*, which is a child of the event definition *insurance_claim_start*:

```
EventDefinition definition = new EventDefinition("insurance_claim_start_auto",
                                                "insurance_claim_start");
```

You can now populate the event definition with property descriptions and extended data element descriptions.

Adding property descriptions to an event definition:

A property description is an instance of the class PropertyDescription.

To add a property description to an event definition, you must first create a new property description and then set the values of its fields. You can then add the property description to the event definition.

1. To create a new property description, create an instance of PropertyDescription, specifying the name and path of the property.

```
PropertyDescription propDesc = new PropertyDescription(name, path);
```

The parameters of this constructor are as follows:

name

The name of the property. This must be the name of a simple property either of the `CommonBaseEvent` element or one of its children.

path

An XPath location path specifying the path to the property. For a top-level property of `CommonBaseEvent` (such as *severity* or *priority*), *path* should be null.

The returned object is a new `PropertyDescription` object.

2. Populate the fields of the property description. The `PropertyDescription` class provides a setter method for each of the fields in a property description. Refer to the Javadoc API documentation for complete information about these methods. For example, to specify that a property is required, you would set the *required* property to true using the `setRequired(boolean)` method:

```
propDesc.setRequired(true);
```
3. Add the property description to the event definition using the `EventDefinition.addPropertyDescription()` method.

```
definition.addPropertyDescription(propDesc);
```

If the event definition already includes another property description with the same name and path, a `DescriptionExistsException` exception is thrown.

The following code fragment creates a new property description, populates it with data, and adds it to an event definition.

```
PropertyDescription propDesc = new PropertyDescription("severity",null);
propDesc.setRequired(true);
propDesc.setMinValue('30');

// definition is a valid event definition
definition.addPropertyDescription(propDesc);
```

Adding extended data element descriptions to an event definition:

An extended data element description is an instance of the `ExtendedDataElementDescription` class.

To add an extended data element description to an event definition, you must first create a new extended data element description and then set the values of its fields. You can also add nested (child) extended data element descriptions, which describe nested extended data elements. You can then add the extended data element description to the event definition.

1. To create a new extended data element description, create an instance of `ExtendedDataElementDescription`, specifying the name and type of the extended data element.

```
ExtendedDataElementDescription edeDesc =
    new ExtendedDataElementDescription(name, type);
```

The parameters of this constructor are as follows:

name

The name of the extended data element. This must be the value of the *name* property of the extended data element you want to describe.

type

The data type of the extended data element. This must be one of the following integer constants defined by the `org.eclipse.hyades.logging.events.cbe.ExtendedDataElement` class:

- `TYPE_BOOLEAN_ARRAY_VALUE`
- `TYPE_BOOLEAN_VALUE`
- `TYPE_BYTE_ARRAY_VALUE`
- `TYPE_BYTE_ARRAY`
- `TYPE_DATE_TIME_ARRAY_VALUE`
- `TYPE_DATE_TIME_VALUE`
- `TYPE_DOUBLE_ARRAY_VALUE`
- `TYPE_DOUBLE_VALUE`
- `TYPE_FLOAT_ARRAY_VALUE`
- `TYPE_FLOAT_VALUE`
- `TYPE_HEX_BINARY_VALUE`
- `TYPE_INT_ARRAY_VALUE`
- `TYPE_INT_VALUE`
- `TYPE_LONG_ARRAY_VALUE`
- `TYPE_LONG_VALUE`
- `TYPE_NO_VALUE_VALUE`
- `TYPE_SHORT_ARRAY_VALUE`
- `TYPE_SHORT_VALUE`
- `TYPE_STRING_ARRAY_VALUE`
- `TYPE_STRING_VALUE`

The returned object is a new `ExtendedDataElementDescription` object.

2. Populate the fields of the extended data element description. The `ExtendedDataElementDescription` class provides a setter method for each of the fields in an extended data element description. Refer to the Javadoc API documentation for complete information about these methods. For example, to specify that an extended data element must occur at least once, you would set the `maxOccurs` property to 4 using the `setMaxOccurs(int)` method:

```
edeDesc.setMaxOccurs(4);
```

3. **Optional:** To add a child extended data element description, use the `ExtendedDataElementDescription.addChild()` method.

```
edeDesc.addChild(childEdeDesc);
```

The `childEdeDesc` parameter must be a valid extended data element description.

4. Add the extended data element description to the event definition using the `EventDefinition.addExtendedDataElementDescription()` method.

```
definition.addExtendedDataElementDescription(edeDesc);
```

If the event definition already includes another extended data element description with the same name and path, a `DescriptionExistsException` exception is thrown.

The following code fragment creates a new extended data element description, populates it with data, and adds it to an event definition.


```

ExtendedDataElementDescription edeDesc =
    new ExtendedDataElementDescription("age", TYPE_SHORT_VALUE);
edeDesc.setMinOccurs(1);
edeDesc.setMaxOccurs(1);

// definition is a valid event definition
definition.addExtendedDataElementDescription(edeDesc);

```

Creating an event catalog bean:

An event catalog bean is used to access the event catalog.

The event catalog is implemented as a stateless session bean using the Enterprise JavaBean architecture. To access the event catalog, an event catalog application must first create an instance of the event catalog session bean.

Use the home interface to create an instance of the event catalog session bean.

```

//use home interface to create event catalog bean
InitialContext context = new InitialContext();
Object eventCatalogHomeObj =
    context.lookup("ejb/com/ibm/events/catalog/EventCatalog");
EventCatalogHome eventCatalogHome = (EventCatalogHome)
    PortableRemoteObject.narrow(eventCatalogHomeObj,
        EventCatalogHome.class);
eventCatalog = (EventCatalog) eventCatalogHome.create();

```

Adding an event definition to the event catalog:

Newly created events can be added to the event catalog.

After you have created a new event definition and populated it with property descriptions and extended data element descriptions, you can add it to the event catalog. Once added to the event catalog, an event definition cannot be modified, but it can be replaced.

Note: If WebSphere security is enabled, the application user ID must be mapped to the catalogAdministrator role to add event definitions to the event catalog.

To add an event definition to the event catalog, use the addEventDefinition method.

```
boolean result = eventCatalog.addEventDefinition(definition, replace)
```

The parameters of this method are as follows:

definition

The event definition you want to add. This must be a valid instance of EventDefinition.

replace

A Boolean value indicating whether the specified event definition replaces an existing definition that has the same name.

If the *replace* parameter is false, the name of the specified event definition must not match that of any existing event definition in the catalog. If it does, an EventDefinitionExistsException exception is thrown.

If the *replace* parameter is true, the new event definition replaces any existing event definition with the same name that is already in the catalog. However, to

preserve the inheritance hierarchy, the new event definition must name the same parent as the old event definition; otherwise, a `ParentNotValidException` exception is thrown.

The returned Boolean indicates whether an existing event definition was replaced. This is equal to true only if *replace* is equal to true and an event definition with the same name was replaced by the new definition.

When an event definition is added to the event catalog, the event catalog sends an event to the event server notifying event consumers that this change occurred. See “*Change notification*” on page 283.

Note: If you attempt to add an event definition that violates inheritance rules, an `InheritanceNotValidException` exception is thrown and the event definition is not added to the catalog. This can happen if a new event definition overrides inherited property or extended data element descriptions in ways that are not valid, or if replacing an existing event definition would cause descendants to override inherited descriptions in ways that are not valid. For more information, see “*Event catalog inheritance*” on page 281.

Removing an event definition from the catalog:

Event definitions no longer needed can be removed from the event catalog.

If an event definition is no longer needed, you can remove it from the event catalog.

Note: If WebSphere security is enabled, the application user ID must be mapped to the `catalogAdministrator` role to remove event definitions from the event catalog.

To remove an event definition from the event catalog, use the `removeEventDefinition` method.

```
eventCatalog.removeEventDefinition(name)
```

The *name* parameter is the name of the event definition you want to remove from the event catalog. If no matching event definition exists in the event catalog, an `EventDefinitionNotFoundException` exception is thrown.

When an event definition is removed from the event catalog, its children and all other descendants are also removed. For each event definition that is removed, the event catalog sends an event to the event server notifying event consumers that this change has taken place. See “*Change notification*” on page 283 for more information.

Note: When an event definition is removed, the event catalog does not check the event server to determine whether any existing events in the event data store are described by that event definition. Therefore, you should make certain that an event definition is no longer needed before removing it from the event catalog.

Command reference

Command-line scripts are available to access some functions of the Common Event Infrastructure.

These scripts are implemented as Jacl scripts, which must be run using the WebSphere wsadmin tool (located in the *profile_root/bin* directory). For more information about the wsadmin tool, see the WebSphere Application Server documentation.

Use the following syntax to run the scripts:

```
wsadmin -f scriptname.jacl
```

You can shorten parameter names, as long as you provide enough of the name to distinguish it from other parameters. For example, when you use the *eventquery.jacl* script, you can type *-ex* instead of *-extensionname*. However, *-e* is not valid, because it can represent either *-extensionname* or *-end*.

To get help with the syntax and usage for a command, type the command followed by the word *help*:

```
wsadmin -f scriptname.jacl help
```

If you are using the wsadmin tool with the Simple Objects Access Protocol (SOAP) protocol, a command might time out before the operation can complete. For example, this might happen if you query or purge a large number of events from the event server. If this happens, the wsadmin tool displays an error message indicating a failed SOAP RPC call:

```
Failed to make the SOAP RPC call: invoke
```

If you get this error message, try the command again, specifying RMI as the connection type and 2809 as the destination port. For example, the following command purges events from the event server using an RMI connection:

```
wsadmin -conntype rmi -port 2809 eventpurge.jacl -seconds 0
```

For more information about the *-conntype* parameter of the wsadmin tool, refer to the WebSphere Application Server documentation.

The eventcatalog.jacl script:

You can use a command line interface to access the event catalog.

Purpose

Lists event definitions or source categories in the event catalog and imports and exports event definitions.

```
wsadmin -f eventcatalog.jacl [-serverName server_name] [-listdefinitions |  
-listcategories | -exportdefinitions | -importdefinitions] [-file filename] [-name  
event_def_name] [-pattern] [-resolve] [-replace]
```

Description

The *eventcatalog.jacl* script provides command-line access to the contents of the event catalog. It also provides support for importing and exporting event definitions.

Parameters

-serverName *server_name*

The name of the application server where the EventServer application is deployed. This argument is optional but should be specified when there are

multiple application servers running on the WebSphere node where the EventServer application is deployed. Otherwise, the commands may fail because they cannot locate the Common Event Infrastructure enterprise beans.

-listdefinitions

Lists the specified event definitions in a readable format, sorted by name in ascending order. The listing is written to the file specified by the **-file** parameter. If this parameter is not specified, the listing is written to the standard output.

-listcategories

Lists all of the defined event source categories and the event classes they contain, sorted by source category in ascending order. The listing is written to the file specified by the **-file** parameter. If this parameter is not specified, the listing is written to the standard output.

-exportdefinitions

Lists the specified event definitions in a format that is suitable for importing. The listing is written as an XML document conforming to the `eventdefinition5_0_1.xsd` XSD schema, which is packaged in the `events-client.jar` file. The listing is written to the file specified by the **-file** parameter. If this parameter is not specified, the listing is written to the standard output.

-importdefinitions

Reads a listing of event definitions from a file and adds the event definitions to the event catalog. The listing of event definitions to import must be written as an XML document that conforms to the `eventdefinition.xsd` XSD schema.

-file *filename*

For a list or export operation, the name of the file to which the output is written. For an import operation, the file that contains the event definitions to be imported. This parameter is required for import operations and optional for list and export operations. If this parameter is not specified for a list or export operation, the output is written to the standard output.

-name *event_def_name*

A name that identifies the event definitions to be listed or exported. If the **-pattern** parameter is not specified, the **-name** parameter identifies a single specific event definition. If **-pattern** is specified, **-name** specifies a pattern against which event definition names are compared. In this pattern, a percent character (%) matches any sequence of zero or more characters, and an underscore (_) matches any single character. All other characters are treated literally.

This parameter is valid only with the **-listdefinitions** and **-exportdefinitions** options. It is not valid with the **-listcategories** or **-importdefinitions** options.

-pattern

Specifies that the value specified with the **-name** parameter is to be treated as a pattern. This parameter is valid only with the **-listdefinitions** and **-exportdefinitions** options. It is not valid with the **-listcategories** or **-importdefinitions** options.

-resolve

Specifies that the event definitions to be listed or exported are resolved. A resolved event definition includes the property and extended data element descriptions that are inherited from its ancestors in the inheritance hierarchy. If this parameter is not specified, the event definition listing contains only the raw event definitions.

This parameter is valid only with the **-listdefinitions** and **-exportdefinitions** options. It is not valid with the **-listcategories** or **-importdefinitions** options.

-replace

Specifies that the event definitions to be imported replace existing event definitions with the same names. If this parameter is not specified, a name collision between an existing event definition and an imported event definition results in an error, and no event definitions are imported.

This parameter is valid only with the **-importdefinitions** option. It is not valid with the **-listdefinitions**, **-listcategories**, or **-exportdefinitions** options.

Examples

This example displays the contents of a single, resolved event definition named `insurance_claim_start` and writes the result to standard output:

```
wsadmin -f eventcatalog.jacl -listdefinitions -name insurance_claim_start -resolve
```

This example exports a set of event definitions, the names of which begin with the string `insurance_claim_start` and writes the result to an XML file:

```
wsadmin -f eventcatalog.jacl -exportdefinitions -file d:\myexport.xml  
-name insurance_claim_start% -pattern
```

This example imports a set of event definitions from the file `myimport.xml` and replaces existing definitions with the same names:

```
wsadmin -f eventcatalog.jacl -importdefinitions -file d:\myimport.xml -replace
```

This example displays a listing of all defined event source categories and the events they contain. The result is written to standard output:

```
wsadmin -f eventcatalog.jacl -listcategories
```

The emitevent.jacl script:

You can use a command line interface to submit events to the event server.

Purpose

Sends an event to the event server.

```
wsadmin -f emitevent.jacl [-xml url] [-serverName server_name] [-msg message]  
[-severity severity] [-extensionname extension_name] [-emitter profile_name]  
[-synchronous | -asynchronous]
```

Description

The `emitevent.jacl` script provides a command-line interface for submitting events to the event server. You can provide the event content by providing a source XML file or by specifying property values on the command line.

Events generated by this script have the following default content:

```
<CommonBaseEvent creationTime=current_system_time  
version="1.0.1"> <sourceComponentId component="emitevent.jacl"  
componentIdType="Application" location=local_hostname  
locationType="Hostname"  
subComponent="com.ibm.events.cli.util.EmitEventCliHelper"  
componentType="http://www.ibm.com/namespaces/autonomic/Tivoli  
/EventInfrastructure"/> <situation categoryName="ReportSituation">
```

```

    <situationType xsi:type="ReportSituation" reasoningScope="EXTERNAL"
      reportCategory="CLI"/>
  </situation>
</CommonBaseEvent>

```

The `current_system_time` parameter is the system time at which the event is generated, specified as an XML `dateTime` string.

Parameters

-xml *url*

A uniform resource locator (URL) that specifies the location of an XML document that contains the event to be submitted. This XML document must conform to the Common Base Event version 1.0.1 XSD schema. If no URL scheme (such as `http://`) is specified, a local file is assumed. This parameter is optional.

Two sample XML files, `eventsample1.xml` and `eventsample2.xml`, are available in the `install_root/events/samples` directory.

-serverName *server_name*

The name of the application server where the EventServer application is deployed. This argument is optional but should be specified when there are multiple application servers running on the WebSphere node where the EventServer application is deployed. Otherwise, the commands may fail because they cannot locate the Common Event Infrastructure enterprise beans.

-msg *message*

The value to use for the message property of the event. If the message contains spaces, enclose this value in quotation marks. This parameter is optional. If you specify this parameter in addition to an XML file, the value of the **-msg** parameter overrides any value specified in the XML file for the `msg` property.

-severity *severity*

The value to use for the severity property of the event. This parameter is optional. If you specify this parameter in addition to an XML file, the value of the **-severity** parameter overrides any value specified in the XML file for the severity property.

-extensionname *extension_name*

The value to use for the `extensionName` property of the event. If the extension name contains spaces, enclose this value in quotation marks. This parameter is optional. If you specify this parameter in addition to an XML file, the value of the **-extensionname** parameter overrides any value specified in the XML file for the `extensionName` property.

-emitter *profile_name*

The Java Naming and Directory Interface (JNDI) name of the emitter factory profile to use when obtaining an emitter. This parameter is optional. If this parameter is not specified, the default emitter factory profile (`/com/ibm/events/configuration/emitter/Default`) is used.

-synchronous | **-asynchronous**

The synchronization mode to use for event transmission. This parameter is optional. If it is not specified, the preferred synchronization mode configured for the emitter is used.

Examples

The following example sends an event to the event server with a severity of 30 and the extension name `test_event` (all other properties have the default values):

```
wsadmin -f emitevent.jacl -severity 30 -extensionname test_event
```

The following example sends an event using the properties specified in `eventsample1.xml`:

```
wsadmin -f emitevent.jacl -xml ../samples/eventsample1.xml
```

The eventquery.jacl script:

You can use a command line interface to query the event database.

Purpose

Generates a report listing events in the event database.

```
wsadmin -f eventquery.jacl [-serverName server_name] [-globalinstanceid global_instance_id | -group event_group] [-severity severity] [-extensionname extension_name] [-start start_time] [-end end_time] [-number number] [-ascending | -descending]
```

Description

The `eventquery.jacl` script queries the event database and generates a report that lists the result. You can query events based on the event group, the severity, or the extension name. You can also query events that were created during a specified period of time.

Parameters

-serverName *server_name*

The name of the application server where the EventServer application is deployed. This argument is optional but should be specified when there are multiple application servers running on the WebSphere node where the EventServer application is deployed. Otherwise, the commands may fail because they cannot locate the Common Event Infrastructure enterprise beans.

-globalinstanceid *global_instance_id*

The global instance identifier of the event to query. Either this parameter or **-group** (but not both) is required.

-group *event_group*

The event group from which to query events. The *event_group* value must be the name of an event group defined in the Common Event Infrastructure configuration. Either this parameter or **-globalinstanceid** (but not both) is required.

-severity *severity*

The severity of the events that you want to include in the report. The *severity* value must be an integer. Only events with a severity equal to the value that you specify are included in the report. This parameter is optional.

-extensionname *extension_name*

The extension name of events that you want include in the report. Use this parameter to restrict the query to events of a specific type. Only events with the `extensionName` property equal to *extension_name* are included in the report. This parameter is optional.

-start *start_time*

The earliest time of the events that you want to include in the report. Use this parameter to restrict the query to events that were generated after a specified

date and time. This parameter must be a date and time that is specified according to the XML dateTime data type. The basic format is CCYY-MM-DDThh:mm:ss, optionally followed by a time zone indicator. For example, noon on 1 January 2004 in Eastern Standard Time is 2004-01-01T12:00:00-05:00. For more information about the dateTime data type, refer to the *XML schema at www.w3.org*.

-end *end_time*

The latest time of the events that you want to include in the report. Use this parameter to restrict the query to events that were generated before a specified date and time. This parameter must be a date and time that is specified according to the XML dateTime data type. For more information, see the description of the **-start** parameter.

-number *number*

The maximum number of events that you want to include in the report. This parameter must be an integer. If the number of matching events in the database exceeds the specified value, the report is truncated. If the report is sorted in ascending order, this means that the most recent matching events are omitted. If the report is sorted in descending order, the oldest matching events are omitted.

-ascending | -descending

The chronological order in which the events in the report are sorted. This must be one of the following values:

ascending

Ascending (chronological) order, with the oldest events first. This is the default value.

descending

Descending (reverse chronological) order, with the most recent events first.

Example

The following example lists all of the events from the database that belong to the **All events** event group and were generated on 17 February 2004. The report is sorted in reverse chronological order:

```
eventquery.jacl -group "All events" -start "2004-02-17T00:00:00-05:00"  
-end "2004-02-17T23:59:59-05:00" -order DESC
```

The eventbucket.jacl script:

You can use a command line interface to change the event database bucket configuration.

Purpose

Displays or changes the event database bucket configuration.

```
wsadmin -f eventbucket.jacl [-status] [-change] [-serverName server_name]
```

Description

The **eventbucket.jacl** script displays or changes the event database bucket configuration. Buckets are used by the rapid purge utility to purge old event data from the event database. By running this command, you can determine the current bucket configuration, or you can swap the active and inactive buckets.

Note: If WebSphere security is enabled, your user ID must be mapped to the eventAdministrator role to view or change the event database bucket configuration.

Parameters

-status

Displays information about the current bucket configuration, including the active bucket setting and the bucket check interval (the frequency with which the data store plug-in checks to determine which bucket is active).

-change

Swaps the buckets so the active bucket becomes inactive and the inactive bucket becomes active. The inactive bucket must be empty before you can use this option.

-serverName *server_name*

The name of the application server where the EventServer application is deployed. This argument is optional but should be specified when there are multiple application servers running on the WebSphere node where the EventServer application is deployed. Otherwise, the commands may fail because they cannot locate the Common Event Infrastructure enterprise beans.

Examples

This example displays the current bucket configuration:

```
wsadmin -f eventbucket.jacl -status
```

This example swaps the active and inactive buckets:

```
wsadmin -f eventbucket.jacl -change
```

The eventpurge.jacl script:

You can use a command line interface to purge events from the event database.

Purpose

Purges events from the event database.

```
wsadmin -f eventpurge.jacl [-serverName server_name] [-seconds seconds | -end  
end_time] [-group event_group] [-severity severity] [-extensionname extension_name]  
[-start start_time] [-size size]
```

Description

The **eventpurge.jacl** script purges events from the event database. You can purge all events from the event database, or you can limit the purge to events meeting certain criteria.

Note: If WebSphere security is enabled, your user ID must be mapped to the eventAdministrator role to delete events.

Parameters

-serverName *server_name*

The name of the application server where the EventServer application is deployed. This argument is optional but should be specified when there are multiple application servers running on the WebSphere node where the

EventServer application is deployed. Otherwise, the commands may fail because they cannot locate the Common Event Infrastructure enterprise beans.

-seconds *seconds*

The minimum age of events you want purged. The *seconds* value must be an integer. Only events older than the specified number of seconds are purged. This parameter is required if you do not specify the **-end** parameter.

-end *end_time*

The end time of the group of events you want to delete. Only events generated before the specified time are deleted. The *end_time* value must be specified in the XML dateTime format (CCYY-MM-DDThh:mm:ss). For example, noon on 1 January 2006 in Eastern Standard Time would be 2006-01-01T12:00:00-05:00. For more information about the dateTime data type, refer to the XML schema at www.w3.org.

This parameter is required if you do not specify the **-seconds** parameter.

-group *eventGroup*

The event group from which to purge events. The *event_group* value must be the name of an event group defined in the Common Event Infrastructure configuration. This parameter is optional.

-severity *severity*

The severity of events you want purged. The *severity* value must be an integer; only events whose severity is equal to the value you specify are purged. This parameter is optional.

-extensionname *extension_name*

The extension name of events you want included in the purge. Use this parameter to restrict the purge to events of a specific type. Only events whose *extensionName* property is equal to *extension_name* are purged. This parameter is optional.

-start *start_time*

The beginning time of the group of events you want to delete. Only events generated after the specified time are deleted. The *start_time* value must be specified in the XML dateTime format (CCYY-MM-DDThh:mm:ss). This parameter is optional.

-size *size*

The number of events to purge in a single transaction. The *size* value must be an integer. After this number of events have been purged, the command commits the transaction before continuing in a new transaction. This parameter is optional.

Example

The following example purges all events from the database whose severity is 20 (harmless) and were generated earlier than 10 minutes ago.

```
eventpurge.jacl -group "All events" -severity 20 -seconds 600
```

Troubleshooting WebSphere Process Server administration

Troubleshooting is the process of finding and eliminating the cause of a problem. This group of topics helps you identify and resolve problems that can occur during typical administration tasks.

Troubleshooting the failed event manager

This topic discusses problems that you can encounter while using the failed event manager.

Note: This topic does not discuss how to use the failed event manager to find, modify, resubmit, or delete failed events on the system. For information about managing failed events, see *Managing WebSphere Process Server failed events* in the information center.

Select the problem you are experiencing from the table below:

Problem	Refer to the following
I am having trouble with reduced performance during an advanced search	"Advanced search feature not optimized"
I am having trouble entering values in the Search page's By Date tab	"Values in the By Date tab automatically change to default if entered incorrectly"
I am having trouble deleting expired events	"Executing the Delete Expired Events function appears to suspend the failed event manager" on page 298

Advanced search feature not optimized

The failed event manager's advanced search feature is not optimized. Therefore, you may experience reduced performance when using the Advanced search tab with a large set of failed events.

Values in the By Date tab automatically change to default if entered incorrectly

The Search page's **By Date** tab contains two fields: **From Date** and **By Date**. Both fields are required. The values are locale-dependent, and they must be formatted exactly as shown in the example above the field. Any inconsistency in the value's format (for example, including four digits in the year instead of 2, or omitting the time) will cause the failed event manager to issue the following warning and substitute a default value in the field:

```
CWMAN0017E: The date entered could not be parsed correctly:  
your_incorrectly_formatted_date. Date: default_date is being used.
```

The default value of the **From Date** field is defined as January 1, 1970, 00:00:00 GMT.

Important: The actual default value shown in your failed event manager implementation will vary depending on your locale and time zone. For example, the From Date field defaults to 12/31/69 7:00 PM for a machine with an en_US locale in the Eastern Standard Time (EST) time zone.

The default value for the **By Date** field is always the current date and time, formatted for your locale and time zone.

To avoid this problem, always enter your dates and times carefully, following the example provided above each field.

Executing the Delete Expired Events function appears to suspend the failed event manager

If you use the Delete Expired Events button in situations where there are many failed events in the current search results, or where those events contain a large amount of business data, the failed event manager can appear to be suspended indefinitely.

In this situation, the failed event manager is not actually suspended; it is working through the large data set, and will refresh the results set as soon as the command completes.

Troubleshooting the Business Process Choreographer configuration

Use this topic to solve problems relating to the configuration of the business process container, or the human task container.

The purpose of this section is to aid you in understanding why the configuration of your business process container or human task container is not working as expected and to help you resolve the problem. The following tasks focus on problem determination and finding solutions to problems that might occur during the configuration of the business process container or the human task container.

Business Process Choreographer log files

This describes where to find the log files for your Business Process Choreographer configuration.

Profile creation

The profile actions for Business Process Choreographer write to the bpecaugment.log file in the logs directory.

If you select the sample configuration option in the profile wizard, it invokes the bpeconfig.jacl script, and actions are logged in the bpeconfig.log file in the logs directory.

Administrative scripts

All of the Business Process Choreographer scripts that are run using wsadmin are logged in the wsadmin.traceout file. However, because this file is overwritten each time that wsadmin is invoked, make sure that you save this log file before invoking wsadmin again.

Configuration-related scripts

The script files bpeconfig.jacl, taskconfig.jacl, clientconfig.jacl, and bpeunconfig.jacl write their log files in the logs directory with the names bpeconfig.log, taskconfig.log, clientconfig.log, and bpeunconfig.log. Also check the wsadmin.traceout file.

Administrative utility scripts

The administrative scripts in the util subdirectory of the ProcessChoreographer directory do not write their own log files. Check the wsadmin.traceout file and the application server log files.

Configuration checker

The bpecheck.jacl script file, found in the ProcessChoreographer directory can be used to check for common configuration problems. The results are written to the bpecheck.log file in the logs directory.

Enabling tracing for Business Process Choreographer

This describes what to do before contacting support.

Enabling tracing

Business Process Choreographer tracing uses the standard WebSphere Process Server tracing mechanism. This must be enabled in the normal way.

The trace specification is as follows:

```
com.ibm.bpe.*=all=enabled;com.ibm.ws.staffsupport.*=all=enabled
```

where com.ibm.bpe traces business processes and most aspects of human tasks. The remaining aspects of human tasks, the staff plug-ins, are traced by com.ibm.ws.staffsupport.

What to send support

After enabling tracing, recreate your problem scenario then provide the following files:

- SystemOut.log
- SystemErr.log
- trace.log

These files are located in *install_rootprofiles/profile_name/logs/*

The task container application fails to start

Startup bean named ejb/htm/TaskContainerStartUpBean forced the application to stop.

Symptom

The following errors are written to the SystemOut.log file:

```
WSVR0037I: Starting EJB jar: taskejb.jar
NMSV0605W: A Reference object looked up from the context "java:"
with the name "comp/env/scheduler/DefaultUserCalendarHome"
was sent to the JNDI Naming Manager and an exception resulted.
Reference data follows:
Reference Factory Class Name: com.ibm.ws.naming.util.IndirectJndiLookupObjectFactory
Reference Factory Class Location URLs:
Reference Class Name: java.lang.Object
Type: JndiLookupInfo
Content: JndiLookupInfo:
jndiName="com/ibm/websphere/scheduler/calendar/DefaultUserCalendarHome";
providerURL=""; initialContextFactory=""
:
StartBeanInfo E STUP0005E: Startup bean named ejb/htm/TaskContainerStartUpBean
forced application to stop.
ApplicationMg W WSVR0101W: An error occurred starting, TaskContainer_utxt1b10Node01_server1
ApplicationMg A WSVR0217I: Stopping application: TaskContainer_utxt1b10Node01_server1
EJBContainerI I WSVR0041I: Stopping EJB jar: taskejb.jar
```

Reason

You get this error if the SchedulerCalendars application is not available when the TaskContainer application starts.

Resolution

Either install the SchedulerCalendars application manually, or if it is already installed, add a new target mapping for it.

In a default profile, the SchedulerCalendars application is available automatically as a WebSphere system application. However, in a custom profile it is not available automatically.

The bpeconfig.jacl script tries to install the SchedulerCalendars application, but this is not always possible.

If you use the administrative console install wizard to configure Business Process Choreographer in an ND environment, you must install the SchedulerCalendars application manually.

Troubleshooting the Business Process Choreographer database and data source

Use this task to solve problems with the Business Process Choreographer database and data source.

Both the business process container and the human-task container need a database. Without the database, enterprise applications that contain business processes and human tasks will not work.

- If you are using DB2[®]:
 - If you use the DB2 Universal JDBC driver type 4 and get DB2 internal errors such as "com.ibm.db2.jcc.a.re: XAER_RMERR : The DDM parameter value is not supported. DDM parameter code point having unsupported value : 0x113f DB2ConnectionCorrelator: NF000001.PA0C.051117223022" when you test the connection on the Business Process Choreographer data source or when the server starts up, perform the following actions:
 1. Check the class path settings for the data source. In a default setup the WebSphere variable `{DB2UNIVERSAL_JDBC_DRIVER_PATH}` can point to the WebSphere Process Server embedded DB2 Universal JDBC driver which is found in the `universalDriver_wbi` directory.
 2. The version of the driver might not be compatible with your DB2 server version. Make sure that you use the original `db2jcc.jar` files from your database installation, and not the WebSphere Process Server embedded DB2 Universal JDBC driver. If required, changed the value of the WebSphere variable `{DB2UNIVERSAL_JDBC_DRIVER_PATH}` to point to your original `db2jcc.jar` file.
 3. Restart the server.
 - If the `db2diag.log` file of your DB2 instance contains messages like `ADM5503E` as illustrated below:

```
2004-06-25-15.53.42.078000 Instance:DB2 Node:000
PID:2352(db2syscs.exe) TID:4360 Appid:*LOCAL.DB2.027785142343
data management sqlEscalateLocks Probe:4 Database:BPEDB
```

```
ADM5503E The escalation of "10" locks on table "GRAALFS .ACTIVITY_INSTANCE_T"
to lock intent "X" has failed. The SQLCODE is "-911"
```


Increase the LOCKLIST value. For example to set the value to 500, enter the following DB2 command:

```
db2 UPDATE DB CFG FOR BPEDB USING LOCKLIST 500
```

This can improve performance significantly.

- To avoid deadlocks, make sure your database system is configured to use sufficient memory, especially for the bufferpool. For DB2, use the DB2 Configuration Advisor to determine reasonable values for your configuration.
- If you get errors mentioning the data source implementation class `COM.ibm.db2.jdbc.DB2XADataSource`:
 - Check that all WebSphere environment variables that are used in the `server.policy` file, have been set correctly. For example, `DB2_INSTALL_ROOT` and `DB2_JDBC_DRIVER_PATH`.
 - Check that the class path definition for your JDBC provider is correct, and that it does not have two entries.
 - Check that the component-managed authentication alias is set to `cellName/BPEAuthDataAliasDbType_Scope`. Where, `cellName` is the name of the cell, `DbType` is the database type, and `Scope` is the scope of the definition.
- If you are using a remote DB2 for z/OS® database, and you get SQL code 30090N in the `SystemOut.log` file when the application server attempts to start the first XA transaction with the remote database, perform the following:
 - Make sure that the instance configuration variable `SPM_NAME` points to the local server with a host name not longer than eight characters. If the host name is longer than eight characters, define a short alias in the `etc/hosts` file.
 - Otherwise, you might have invalid syncpoint manager log entries in the `sql1lib/spmlog` directory. Try clearing the entries in the `sql1lib/spmlog` directory and restart.
 - Consider increasing the value of `SPM_LOG_FILE_SZ`.
- If you are using Cloudscape:
 - If you get a "Too many open files" error on Linux or UNIX systems, increase the number of file handles available, for example, to 4000 or more. For more information about how to increase the number of available file handles, refer to the documentation for your operating system.
 - If you get a "Java class not found" exception when trying to invoke Cloudscape tools, make sure that you have set up the Java environment, and that your `classpath` environment variable includes the following JAR files:
 - `db2j.jar`
 - `db2jtools.jar`
 - `db2jcc.jar`
 - `db2jcvview.jar`
 - If you cannot connect to your Cloudscape database using the Cloudscape tools (like `ij` or `cvview`) and you get the following exception:

```
ERROR XJ040: Failed to start database 'c:\WebSphere\AppServer\profiles\profile_name\databases\BPEDB',
see the next exception for details.
ERROR XSDB6: Another instance of Cloudscape may have already booted the database
c:\WebSphere\AppServer\profiles\profile_name\databases\BPEDB.
```

you must stop your WebSphere Application Server before using these tools because only one application can access the Cloudscape database at a time.

- If you get a database error when installing an enterprise application that contains a business process or human task. When an enterprise application is

installed, any process templates and task templates are written into the Business Process Choreographer database. Make sure that the database system used by the business process container is running and accessible.

- If you have problems using national characters. Make sure that your database was created with support for Unicode character sets.
- If tables or views cannot be found in the database. When configuring the authentication alias for the data source, you must specify the same user ID that was used to create the database tables (or to run the scripts to create them).

Troubleshooting the Business Process Choreographer queue manager and JMS provider

Use this to solve problems with Business Process Choreographer relating to queues, the queue manager, and the Java Message Service (JMS) provider.

Business Process Choreographer uses reliable messaging. The messaging service can either be the JMS provider embedded in WebSphere

, or the separately installed product WebSphere

MQ. Here are some solutions to possible problems:

Troubleshooting business process and human tasks

Use this topic to solve problems relating to business processes and human tasks.

The following tasks focus on troubleshooting problems that can happen during the execution of a business process or task.

Troubleshooting the installation of business process and human task applications

When installing an application containing business processes, human tasks, or both in an ND environment, you get an exception in the deployment manager SystemErr.log file

Symptom

When installing an application containing business processes, human tasks, or both in an ND environment, you find the following exception in the deployment manager SystemErr.log file:

```
SystemErr R com.ibm.ws.management.commands.sib.SIBAdminCommandException:
CWSJA0012E: Messaging engine not found.
at com.ibm.ws.management.commands.sib.SIBAdminCommandHelper.createDestination
(SIBAdminCommandHelper.java:787)
at com.ibm.ws.management.commands.sib.CreateSIBDestinationCommand.afterStepsExecuted
(CreateSIBDestinationCommand.java:459)
at com.ibm.websphere.management.cmdframework.provider.AbstractTaskCommand.execute
(AbstractTaskCommand.java:547)
at com.ibm.ws.sca.internal.deployment.sib.SIBAdminHelper.call(SIBAdminHelper.java:136)
at com.ibm.ws.sca.internal.deployment.sib.SIBAdminHelper.createSIBDestination
(SIBAdminHelper.java:112)
at com.ibm.ws.sca.internal.deployment.sib.SIBAdmin.createDestination(SIBAdmin.java:327)
at com.ibm.ws.sca.internal.deployment.sib.SIBDestinationTask.createDestination
(SIBDestinationTask.java:263)
at com.ibm.ws.sca.internal.deployment.sib.SIBDestinationTask.preInstallModule
(SIBDestinationTask.java:71)
at com.ibm.ws.sca.internal.deployment.SCATaskBase.installModule(SCATaskBase.java:57)
at com.ibm.ws.sca.internal.deployment.sib.SIBDestinationTask.processArtifacts
(SIBDestinationTask.java:228)
at com.ibm.ws.sca.internal.deployment.sib.SIBDestinationTask.install
```

```
(SIBDestinationTask.java:287)
at com.ibm.ws.sca.internal.deployment.SCAInstallTask.performInstallTasks
(SCAInstallTask.java:116)
at com.ibm.ws.sca.internal.deployment.SCAInstallTask.performTask
(SCAInstallTask.java:61)
at com.ibm.ws.management.application.SchedulerImpl.run(SchedulerImpl.java:253)
at java.lang.Thread.run(Thread.java:568)
```

Reason

The bus member for the "SCA.SYSTEM.cellName.Bus" bus is missing.

Resolution

In the administrative console, click **Service Integration** → **Buses** → **SCA.SYSTEM.cellName.Bus**. In the Topology section, click **Bus members**. Add the server or cluster where you want to install the business process or human task application as a bus member, then restart the affected server or cluster and try installing the application again.

Troubleshooting the execution of business processes

This describes the solutions to common problems with business process execution.

In Business Process Choreographer Explorer, you can search for error message codes on the IBM[®]

technical support pages.

1. On the error page, click the **Search for more information** link. This starts a search for the error code on the IBM technical support site. This site only provides information in English.
2. Copy the error message code that is shown on the error page to the clipboard. The error code has the format CWWBcnnnc, where each c is a character and nnnn is a 4-digit number. Go to the WebSphere Process Server technical support page.
3. Paste the error code into the **Additional search terms** field and click **Go**.

Solutions to specific problems are in the following topics.

ClassCastException when stopping an application containing a microflow:

The SystemOut.log file contains ClassCastException exceptions around the time when an application containing a microflow had been stopped.

Reason

When an application is stopped, the classes contained in the EAR file are removed from the class path. However, microflow instances may still be executing that need these classes.

Resolution

Perform the following actions:

1. Stop the microflow process template first. From now on, it is not possible to start new microflow instances from that template.
2. Wait for at least the maximum duration of the microflow execution so that any running instances can complete.

3. Stop the application.

XPath query returns an unexpected value from an array:

Using an XPath query to access a member in an array returns an unexpected value.

Reason

A common cause for this problem is assuming that the first element in the array has an index value of zero. In XPath queries in arrays, the first element has the index value one.

Resolution

Check that your use of index values into arrays start with element one.

An activity has stopped because of an unhandled fault (Message: CWWBE0057I):

The system log contains a CWWBE0057I message, the process is in the state "running", but it does not proceed its navigation on the current path.

Reason

Invoke activities, inline human tasks, and Java snippets are put in a stopped state, if all of the following happen:

- A fault is raised by the activity
- The fault is not handled on the enclosing scope
- The continueOnError attribute of the activity is set to false

Resolution

The solution to this problem requires actions at two levels:

1. An administrator must repair the stopped activity instance manually. For example, to force complete or force retry the stopped activity instance.
2. The reason for the failure must be investigated. In some cases the failure is caused by a modeling error that must be corrected in the model.

For example, if you use the WebSphere Scheduler default calendar, and have an expiration time with 'Timeout' defined for your activity, make sure that the definition of the time period is in the correct format, in particular make sure that there is no blank between the number and the unit of time. Examples of correctly specified timeout periods:

- 1minute
- 2hours 4minutes 1second
- 1day 1hour

A microflow is not compensated:

A microflow has called a service, and the process fails, but the undo service is not called.

Resolution

There are various conditions that must be met to trigger the compensation of a microflow. Check the following:

1. Log on to the Business Process Choreographer Explorer and click **Failed Compensations** to check whether the compensation service has failed and needs to be repaired.
2. The compensation of a microflow is only triggered when the transaction for the microflow is rolled back. Check whether this is the case.
3. The compensationSphere attribute of the microflow must be set to required.
4. A compensation service is only run, if the corresponding forward service has not participated in the microflow's transaction. Ensure that the forward service does not participate in the navigation transaction, for example, on the reference of the process component, set the Service Component Architecture (SCA) qualifier suspendTransaction to True.

A long-running process appears to have stopped:

A long-running process is in the state running, but it appears that it is doing nothing.

Reason

There are various possible reasons for such behavior:

1. A navigation message has been retried too many times and has been moved to the retention or hold queue.
2. A reply message from the Service Component Architecture (SCA) infrastructure failed repeatedly.
3. The process is waiting for an event, timeout, or for a long-running invocation or task to return.
4. An activity in the process is in the stopped state.

Resolution

Each of the above reasons requires different corrective actions:

1. Check if there are any messages in the retention or hold queue, as described in the PDF for administering.
2. Check if there are any in the failed event management view of the administrative console.
 - If there are any failed events from Service Component Architecture (SCA) reply messages, reactivate the messages.
 - Otherwise, either force complete or force retry the long-running activity.
3. Check if there are activities in the stopped state, and repair these activities. If your system log contains a CWWBE0057I message you might also need to correct your model as described in Message: CWWBE0057I.

Invoking a synchronous subprocess in another EAR file fails:

When a long-running process calls another process synchronously, and the subprocess is located in another enterprise archive (EAR) file, the subprocess invocation fails.

Example of the resulting exception:

```
com.ibm.ws.sca.internal.ejb.util.EJBStubAdapter com.ibm.ws.sca.internal.ejb.util.EJBStubAdapter#003
Exception:
java.rmi.AccessException: CORBA NO_PERMISSION 0x49424307 No; nested exception is:
org.omg.CORBA.NO_PERMISSION: The WSCredential does not contain a forwardable token.
Please enable Identity Assertion for this scenario.
vmcid: 0x49424000 minor code: 307 completed: No
at com.ibm.CORBA.iiop.UtilDelegateImpl.mapSystemException(UtilDelegateImpl.java:202)
at javax.rmi.CORBA.Util.mapSystemException(Util.java:84)
```

Reason

Common Secure Interoperability Version 2 (CSIv2) identity assertion must be enabled when calling a synchronous subprocess in another EAR file.

Resolution

Configure CSIv2 inbound authentication and CSIv2 outbound authentication.

Unexpected exception during execution (Message: CWWBA0010E):

Either the queue manager is not running or the Business Process Choreographer configuration contains the wrong database password.

Resolution

Check the following:

1. If the `systemout.log` file contains "javax.jms.JMSEException: MQJMS2005: failed to create MQQueueManager", start the queue manager.
2. Make sure that the database administrator password stored in the Business Process Choreographer configuration matches the one set in the database.

Event unknown (Message: CWWBE0037E):

An attempt to send an event to a process instance or to start a new process instance results in a "CWWBE0037E: Event unknown." exception.

Reason

A common reason for this error is that a message is sent to a process but the receive or pick activity has already been navigated, so the message cannot be consumed by this process instance again.

Resolution

To correct this problem:

- If the event is supposed to be consumed by an existing process instance, you must pass correlation set values that match an existing process instance which has not yet navigated the corresponding receive or pick activity.
- If the event is supposed to start a new process instance, the correlation set values must not match an existing process instance.

For more information about using correlation sets in business processes, see technote 1171649.

Cannot find nor create a process instance (Message: CWWBA0140E):

An attempt to send an event to a process instance results in a 'CreateRejectedException' message.

Reason

A common reason for this error is that a message is sent to a receive or pick activity that cannot instantiate a new process instance because its createInstance attribute is set to no and the values that are passed with the message for the correlation set which is used by this activity do not match any existing process instances.

Resolution

To correct this problem you must pass a correlation set value that matches an existing process instance.

For more information about using correlation sets in business processes, see [Correlation sets in BPEL processes](#) .

Uninitialized variable or NullPointerException in a Java snippet:

Using an uninitialized variable in a business process can result in diverse exceptions.

Symptoms

Exceptions such as:

- During the execution of a Java snippet or Java expression, that reads or manipulate the contents of variables, a NullPointerException is thrown.
- During the execution of an assign, invoke, reply or throw activity, the BPEL standard fault "uninitializedVariable" (message CWWBE0068E) is thrown.

Reason

All variables in a business process have the value null when a process is started, the variables are not pre-initialized. Using an uninitialized variable inside a Java snippet or Java expression leads to a NullPointerException.

Resolution

The variable must be initialized before it is used. This can be done by an assign activity, for example, the variable needs to occur on the to-spec of an assign, or the variable can be initialized inside a Java snippet.

Missing reply exception (message: CWWBE0071E):

The execution of a microflow or long-running process results in a MissingReplyException (message: CWWBE0071E), or this exception is found in the system log or SystemOut.log file.

Reason

A two-way operation must send a reply. This error is generated if the process ends without navigating the reply activity. This can happen in any of the following circumstances:

- The reply activity is skipped.
- A fault occurs and corresponding fault handler does not contain a reply activity.
- A fault occurs and there is no corresponding fault handler.

Resolution

Correct the model to ensure that a reply activity is always performed before the process ends.

Parallel paths are sequentialized:

There are two or more parallel invoke activities inside a flow activity, but the invoke activities are run sequentially.

Resolution

- To achieve real parallelism, each path must be in a separate transaction. Set the 'transactional behavior' attribute of all the parallel invoke activities to 'commit before' or 'requires own'.
- If you are using Cloudscape as the database system, the process engine will serialize the execution of parallel paths. You cannot change this behavior.

Copying a nested data object to another data object destroys the reference on the source object:

A data object, Father, contains another data object, Child. Inside a Java snippet, the object containing Child is fetched and set on a substructure of data object, Mother. The reference to Child in data object Father disappears.

Reason

The reference to Child is moved from Father to Mother.

Resolution

When such a data transformation is performed in a Java snippet, copy the data object before it is assigned to another object. The following code snippet illustrates how to do this:

```
BOCopy copyService = (BOCopy)ServiceManager.INSTANCE.locateService
    ("com/ibm/websphere/bo/BOCopy");
DataObject Child = Father.get("Child");
DataObject BCopy = copyService.copy(Child);
Mother.set("Child", BCopy);
```

CScope is not available:

Starting a microflow or running a navigation step in a long-running process fails with an assertion, saying: 'postcondition violation !(cscope != null)'.

Reason

In certain situations, the process engine uses the compensation service, but it was not enabled.

Resolution

Enable the compensation service as described in the PDF for administration.

Working with process-related or task-related messages

Describes how to get more information about Business Process Choreographer messages that are written to the display or a log file.

Messages that belong to Business Process Choreographer are prefixed with either *CWWB* for process-related messages, or *CWTK* for task-related messages. The format of these messages is *PrefixComponentNumberTypeCode*. The type code can be:

- I** Information message
- W** Warning message
- E** Error message

When processes and tasks run, messages are either displayed in Business Process Choreographer Explorer, or they are added to the `SystemOut.log` file and traces. If the message text provided in these files is not enough to help you solve your problem, you can use the WebSphere Application Server symptom database to find more information. To view Business Process Choreographer messages, check the `activity.log` file by using the WebSphere log analyzer.

1. Start the WebSphere log analyzer.
Run the following script: `install_root/bin/waslogbr.sh`
2. **Optional:** Click **File > Update database > WebSphere Application Server Symptom Database** to check for the newest version of the symptom database.
3. **Optional:** Load the activity log.
 - a. Select the activity log file
 - `install_root/profiles/profile_name/logs/activity.log` file
 - b. click **Open**.

Troubleshooting Business Process Choreographer Explorer

Use this to solve problems relating to the Business Process Choreographer Explorer.

Use the following information to solve problems relating to Business Process Choreographer Explorer.

- If you try to access Business Process Choreographer Explorer with a browser, but get the error message 'HTTP 404 - File not found', try the following:
 - Use the administrative console to make sure that the Web client application `BPCEXplorer_node_name_server_name` is actually deployed and running on the server.
 - In the administrative console, on the page for the application, under "View Deployment Descriptor", verify that the context root is `/bpc`.
- If you get an error message when using Business Process Choreographer Explorer, click the **Search for more information** link on the error page. This starts a search for the error code on the IBM technical support site. This site only provides information in English. Copy the error message code that is shown on the Business Process Choreographer Explorer Error page to the clipboard. The error code has the format `CWWBcnnnnc`, where each `c` is a character and `nnnn` is a 4-digit number. Go to the WebSphere Process Server technical support page. Paste the error code into the **Additional search terms** field and click **Go**.
- If you get an `EngineMissingReplyException` message, this is a symptom of a problem with your process model. For more information about solving this, see "Troubleshooting the administration of business processes and human tasks" on page 310.

- If you can log onto Business Process Choreographer Explorer, but some items are not displayed, or if certain buttons are not enabled, this indicates a problem with your authorization.

Possible solutions to this problem include:

- Use the administrative console to turn security on.
- Check that you are logged onto Business Process Choreographer Explorer using the correct identity. If you log on with a user ID that is not a process administrator, all administrative views and options will be invisible or not enabled.
- Use WebSphere Integration Developer to check or modify the authorization settings defined in the business process.
- Error message WWBU0024E Could not establish a connection to local business process EJB with a reason: "Naming Exception". This error can indicate that the business process container has been stopped. Verify that the application BPEContainer_*InstallScope* is running, where *InstallScope* is either the *cluster_name* or *hostname_servername*.

Related tasks

"Troubleshooting the execution of business processes" on page 303

This describes the solutions to common problems with business process execution.

Troubleshooting the administration of business processes and human tasks

This article describes how to solve some common problems with business processes.

The following information can help you to debug problems with your business processes.

The administrative console stops responding if you try to stop a business process application while it still has process instances. Before you try to stop the application, you must stop the business processes so that no new instances are created, and do one of the following:

- Wait for all of the existing process instances to end in an orderly way.
- Terminate and delete all of the process instances.

Only then, can you stop the process application. For more information about preventing this problem, refer to technote 1166009.

Using process-related and task-related audit trail information

Explains the event types and database structures for business processes and human tasks.

Logging must be enabled for the business process container, the task container, or both.

If logging is enabled, whenever a significant step during the running of a business process or a human task occurs, information is written to the audit log or Common Event Infrastructure (CEI) log. For more information about CEI, refer to the PDF for monitoring. The following topics describe the event types and database structures for business processes and human tasks.

Audit event types for business processes:

This describes the types of events that can be written to the audit log during the processing of business processes.

For an event to be logged, the following conditions must be met:

- The corresponding audit logging type is enabled for the business process container
- The event must be enabled for the corresponding entity in the process model

The following tables list the codes for audit events that can occur while business processes are running.

Table 13. Process instance events

Audit event	Event code
PROCESS_STARTED	21000
PROCESS_SUSPENDED	21001
PROCESS_RESUMED	21002
PROCESS_COMPLETED	21004
PROCESS_TERMINATED	21005
PROCESS_RESTARTED	21019
PROCESS_DELETED	21020
PROCESS_FAILED	42001
PROCESS_COMPENSATING	42003
PROCESS_COMPENSATED	42004
PROCESS_TERMINATING	42009
PROCESS_FAILING	42010
PROCESS_CORRELATION_SET_INITIALIZED	42027
PROCESS_COMPENSATION_INDOUBT	42030
PROCESS_WORKITEM_DELETED	42041
PROCESS_WORKITEM_CREATED	42042
PROCESS_COMPENSATION_FAILED	42046
PROCESS_EVENT_RECEIVED	42047
PROCESS_EVENT_ESCALATED	42049
PROCESS_WORKITEM_TRANSFERRED	42056

Table 14. Activity events

Audit event	Event code
ACTIVITY_READY	21006
ACTIVITY_STARTED	21007
ACTIVITY_COMPLETED	21011
ACTIVITY_CLAIM_CANCELED	21021
ACTIVITY_CLAIMED	21022
ACTIVITY_TERMINATED	21027
ACTIVITY_FAILED	21080
ACTIVITY_EXPIRED	21081
ACTIVITY_LOOPED	42002

Table 14. Activity events (continued)

Audit event	Event code
ACTIVITY_SKIPPED	42005
ACTIVITY_TERMINATING	42008
ACTIVITY_FAILING	42011
ACTIVITY_OUTPUT_MESSAGE_SET	42012
ACTIVITY_FAULT_MESSAGE_SET	42013
ACTIVITY_STOPPED	42015
ACTIVITY_FORCE_RETRIED	42031
ACTIVITY_FORCE_COMPLETED	42032
ACTIVITY_UNDO_STARTED	42033
ACTIVITY_UNDO_SKIPPED	42034
ACTIVITY_UNDO_COMPLETED	42035
ACTIVITY_MESSAGE_RECEIVED	42036
ACTIVITY_LOOP_CONDITION_TRUE	42037
ACTIVITY_LOOP_CONDITION_FALSE	42038
ACTIVITY_WORKITEM_DELETED	42039
ACTIVITY_WORKITEM_CREATED	42040
ACTIVITY_ESCALATED	42050
ACTIVITY_WORKITEM_REFRESHED	42054
ACTIVITY_WORKITEM_TRANSFERRED	42055

Table 15. Events related to variables

Audit event	Event code
VARIABLE_UPDATED	21090

Table 16. Control link events

Audit event	Event code
LINK_EVALUATED_TO_TRUE	21034
LINK_EVALUATED_TO_FALSE	42000

Table 17. Process template events

Audit event	Event code
PROCESS_INSTALLED	42006
PROCESS_UNINSTALLED	42007

Table 18. Scope instance events

Audit event	Event code
SCOPE_STARTED	42020
SCOPE_SKIPPED	42021
SCOPE_FAILED	42022
SCOPE_FAILING	42023

Table 18. Scope instance events (continued)

Audit event	Event code
SCOPE_TERMINATED	42024
SCOPE_COMPLETED	42026
SCOPE_COMPENSATING	42043
SCOPE_COMPENSATED	42044
SCOPE_COMPENSATION_FAILED	42045
SCOPE_EVENT_RECEIVED	42048
SCOPE_EVENT_ESCALATED	42051

Audit event types for human tasks:

This describes the types of events that can be written to the audit log during the processing of human tasks.

For an event to be logged, the following conditions must be met:

- The corresponding audit logging type is enabled for the human task container
- The event must be enabled for the corresponding entity in the task model

The following tables list the codes for audit events that can occur while human tasks are running.

Table 19. Task instance events

Audit event	Event code
TASK_CREATED	51001
TASK_DELETED	51002
TASK_STARTED	51003
TASK_COMPLETED	51004
TASK_CLAIM_CANCELLED	51005
TASK_CLAIMED	51006
TASK_TERMINATED	51007
TASK_FAILED	51008
TASK_EXPIRED	51009
TASK_WAITING_FOR_SUBTASK	51010
TASK_SUBTASKS_COMPLETED	51011
TASK_RESTARTED	51012
TASK_SUSPENDED	51013
TASK_RESUMED	51014
TASK_COMPLETED_WITH_FOLLOW_ON	51015
TASK_UPDATED	51101
TASK_OUTPUT_MESSAGE_UPDATED	51103
TASK_FAULT_MESSAGE_UPDATED	51104
TASK_WORKITEM_DELETED	51201
TASK_WORKITEM_CREATED	51202
TASK_WORKITEM_TRANSFERRED	51204

Table 19. Task instance events (continued)

Audit event	Event code
TASK_WORKITEM_REFRESHED	51205

Table 20. Task template events

Audit event	Event code
TASK_TEMPLATE_INSTALLED	52001
TASK_TEMPLATE_UNINSTALLED	52002

Table 21. Escalation instance events

Audit event	Event code
ESCALATION_FIRED	53001
ESCALATION_WORKITEM_DELETED	53201
ESCALATION_WORKITEM_CREATED	53202
ESCALATION_WORKITEM_TRANSFERRED	53204
ESCALATION_WORKITEM_REFRESHED	53205

Structure of the audit trail database view for business processes:

The AUDIT_LOG_B database view provides audit log information about business processes.

To read the content of the audit trail, use SQL or any other administration tool that supports the reading of database tables and views.

Audit events are related to process entities. The audit event types depend on the entity to which the event refers. The audit event types include:

- Process instance events (PIE)
- Activity instance events (AIE)
- Events related to variables (VAR)
- Control link events (CLE)
- Process template events (PTE)
- Scope-related events (SIE).

For a list of the audit event type codes, see “Audit event types for business processes” on page 310.

The following table describes the structure of the AUDIT_LOG_B audit trail view. It lists the names of the columns, the event types, and gives a short description for the column.

Inline tasks are logged in the AUDIT_LOG_B audit trail view and not in the TASK_LOG audit trail view. For example, claiming an inline participating task results in an ACTIVITY_CLAIMED event; a task-related event is not generated.

Table 22. Structure of AUDIT_LOG_B audit trail view

Name	PTE	PIE	AIE	VAR	CLE	SIE	Description
ALID	x	x	x	x	x	x	Identifier of the audit log entry.

Table 22. Structure of AUDIT_LOG_B audit trail view (continued)

Name	PTE	PIE	AIE	VAR	CLE	SIE	Description
EVENT_TIME	x	x	x	x	x	x	Timestamp of when the event occurred in Coordinated Universal Time (UTC) format.
EVENT_TIME_UTC	x	x	x	x	x	x	Timestamp of when the event occurred in Coordinated Universal Time (UTC) format.
AUDIT_EVENT	x	x	x	x	x	x	The type of event that occurred.
PTID	x	x	x	x	x	x	Process template ID of the process that is related to the current event.
PIID		x	x	x	x	x	Process instance ID of the process instance that is related to the current event.
VARIABLE_NAME				x			The name of the variable related to the current event.
SIID						x	The ID of the scope instance related to the event.
PROCESS_TEMPL_NAME	x	x	x	x	x	x	Process template name of the process template that is related to the current event.
TOP_LEVEL_PIID		x	x	x	x	x	Identifier of the top-level process that is related to the current event.
PARENT_PIID		x	x	x	x	x	Process instance ID of the parent process, or null if no parent exists.
VALID_FROM	x	x	x	x	x	x	Valid-from date of the process template that is related to the current event.
VALID_FROM_UTC	x	x	x	x	x	x	Valid-from date of the process template that is related to the current event in Coordinated Universal Time (UTC) format.
ATID			x				The ID of the activity template related to the current event.
ACTIVITY_NAME			x			x	Name of the activity on which the event occurred.

Table 22. Structure of AUDIT_LOG_B audit trail view (continued)

Name	PTE	PIE	AIE	VAR	CLE	SIE	Description
ACTIVITY_KIND			x				<p>Kind of the activity on which the activity occurred. Possible values are:</p> <p>KIND_EMPTY 3 KIND_INVOKE 21 KIND_RECEIVE 23 KIND_REPLY 24 KIND_THROW 25 KIND_TERMINATE 26 KIND_WAIT 27 KIND_COMPENSATE 29 KIND_SEQUENCE 30 KIND_SWITCH 32 KIND_WHILE 34 KIND_PICK 36 KIND_FLOW 38 KIND_SCRIPT 42 KIND_STAFF 43 KIND_ASSIGN 44 KIND_CUSTOM 45 KIND_RETHROW 46</p> <p>These are the constants defined for ActivityInstanceData.KIND_*</p>
ACTIVITY_STATE			x				<p>State of the activity that is related to the event. Possible values are:</p> <p>STATE_INACTIVE 1 STATE_READY 2 STATE_RUNNING 3 STATE_SKIPPED 4 STATE_FINISHED 5 STATE_FAILED 6 STATE_TERMINATED 7 STATE_CLAIMED 8 STATE_TERMINATING 9 STATE_FAILING 10 STATE_WAITING 11 STATE_EXPIRED 12 STATE_STOPPED 13</p> <p>These are the constants defined for ActivityInstanceData.STATE_*</p>
CONTROL_LINK_NAME					x		Name of the link that is related to the current link event.
PRINCIPAL		x	x	x	x	x	Name of the principal. This is not set for PROCESS_DELETED events.
VARIABLE_DATA				x			Data for variables for variable updated events.

Table 22. Structure of AUDIT_LOG_B audit trail view (continued)

Name	PTE	PIE	AIE	VAR	CLE	SIE	Description
EXCEPTION_TEXT		x	x			x	Exception message that caused an activity or process to fail. Applicable for: PROCESS_FAILED ACTIVITY_FAILED SCOPE_FAILED
DESCRIPTION		x	x	x	x	x	Description of activity or process, containing potentially resolved replacement variables.
CORR_SET_INFO		x					The string representation of the correlation set that was initialized at process start time. Provided with the processCorrelationSetInitialized event (42027).
USER_NAME		x	x				The name of the user whose work item has been changed. This is applicable for the following events: <ul style="list-style-type: none"> • Process instance work item deleted • Activity instance work item deleted • Process instance work item created • Activity instance work item created

Table 22. Structure of AUDIT_LOG_B audit trail view (continued)

Name	PTE	PIE	AIE	VAR	CLE	SIE	Description
ADDITIONAL_INFO		x	x			x	<p>The contents of this field depends on the type of the event:</p> <p>ACTIVITY_WORKITEM_TRANSFERRED, PROCESS_WORK_ITEM_TRANSFERRED The name of the user that received the work item.</p> <p>ACTIVITY_WORKITEM_CREATED, ACTIVITY_WORKITEM_REFRESHED, ACTIVITY_ESCALATED The list of all of the users for which the work item was created or refreshed, separated by ','. If the list contains only one user, USER_NAME field is filled with the user name of this user. If the list contains only one user, the USER_NAME field is filled with the user name of this user and the ADDITIONAL_INFO field will be empty (null).</p> <p>PROCESS_EVENT_RECEIVED, SCOPE_EVENT_RECEIVED If available, the type of operation that was received by an event handler. The following format is used: '{' port type namespace '}' port type name ':' operation name. This field is not set for 'onAlarm' events.</p>

Structure of the audit trail database view for human tasks:

The TASK_AUDIT_LOG database view provides audit log information about human tasks.

Inline tasks are logged in the AUDIT_LOG_B view , whereas all other task types are logged in the TASK_AUDIT_LOG view.

To read the content of the audit trail, use SQL or any other administration tool that supports the reading of database tables and views.

Audit events are related to task entities. The audit event types depend on the entity to which the event refers. The audit event types include:

- Task instance events (TIE)
- Task template events (TTE)
- Escalation instance events (EIE)

The following table describes the structure of the TASK_AUDIT_LOG audit trail view. It lists the names of the columns, the event types, and gives a short description for the column.

Inline tasks are logged in the AUDIT_LOG_B audit trail view and not in the TASK_AUDIT_LOG audit trail view. For example, claiming an inline participating task results in an ACTIVITY_CLAIMED event; a task-related event is not generated.

Table 23. Structure TASK_AUDIT_LOG audit trail view

Name	TIE	TTE	EIE	Description
ALID	x	x	x	The identifier of the audit log entry.
AUDIT_EVENT	x	x	x	The type of event that occurred. For a list of audit event codes, see "Audit event types for human tasks" on page 313.
CONTAINMENT_CTX_ID	x	x		The identifier of the containing context, for example, ACOID, PTID, or PIID.
ESIID			x	The identifier of the escalation instance that is related to the current event.
ESTID			x	The identifier of the escalation template that is related to the current event.
EVENT_TIME	x	x	x	The time when the event occurred in Coordinated Universal Time (UTC) format.
FAULT_NAME	x			The name of the fault message. This attribute is applicable to the following events: TASK_FAILED TASK_FAULT_MESSAGE_UPDATED
FAULT_NAME_SPACE	x			The namespace of the fault message type. This attribute is applicable to the following events: TASK_FAILED TASK_FAULT_MESSAGE_UPDATED
FOLLOW_ON_TKIID	x			The ID of the follow-on task instance.
NAME	x	x	x	The name of the task instance, task template, or escalation instance that is associated with the event.
NAMESPACE	x	x		The namespace of the task instance, task template, or escalation instance that is associated with the event.
NEW_USER				The new owner of a transferred work item. This attribute is applicable to the following events:
	x			TASK_WORKITEM_CREATED
	x			TASK_WORKITEM_TRANSFERRED
			x	ESCALATION_WORKITEM_CREATED
OLD_USER			x	ESCALATION_WORKITEM_TRANSFERRED
	x			TASK_WORKITEM_TRANSFERRED
	x			TASK_WORKITEM_DELETED
			x	ESCALATION_WORKITEM_TRANSFERRED
			x	ESCALATION_WORKITEM_DELETED

Table 23. Structure TASK_AUDIT_LOG audit trail view (continued)

Name	TIE	TTE	EIE	Description
PARENT_CONTEXT_ID	x			The ID of the parent context of the task, for example, an activity template or a task instance. This is only set for subtasks and follow-on tasks.
PARENT_TASK_NAME	x			The name of the parent task instance or template. This is only set for subtasks and follow-on tasks.
PARENT_TASK_NAMESP	x			The namespace of the parent task instance or template. This is only set for subtasks and follow-on tasks.
PARENT_TKIID	x			The identifier of the parent task instance.
PRINCIPAL	x	x	x	The name of the principal whose request triggered the event.
TASK_KIND	x	x		The kind of the task. Possible values are: KIND_HUMAN 101 KIND_ORIGINATING 103 KIND_PARTICIPATING 105 KIND_ADMINISTRATIVE 106
TASK_STATE	x			The state of the task or task template. Possible values for task templates are: STATE_STARTED 1 STATE_STOPPED 2 Possible values for task instances are: '1' :STATE_INACTIVE' '2' :STATE_READY' '3' :STATE_RUNNING' '5' :STATE_FINISHED' '6' :STATE_FAILED' '7' :STATE_TERMINATED' '8' :STATE_CLAIMED' '12' :STATE_EXPIRED' '101':FORWARDED'
TKIID	x		x	The identifier of the task instance.
TKTID	x	x		The identifier of the task template.
TOP_TKIID	x			The identifier of the top task instance.
VALID_FROM		x		Valid-from date of the task template that is related to the current event.

Table 23. Structure TASK_AUDIT_LOG audit trail view (continued)

Name	TIE	TTE	EIE	Description
WORK_ITEM_REASON	x		x	<p>The reason for the assignment of the work item. Possible values are:</p> <p>POTENTIAL_OWNER 1 EDITOR 2 READER 3 OWNER 4 POTENTIAL_STARTER 5 STARTER 6 ADMINISTRATOR 7 POTENTIAL_SENDER 8 ORIGINATOR 9 ESCALATION_RECEIVER 10 POTENTIAL_INSTANCE_CREATOR 11</p> <p>The reason is set for all events related to work items: ESCALATION_RECEIVER is set for escalation work item related events, while the other reasons apply to task work item related events.</p>

Troubleshooting business rules manager

Some areas to examine if you experience problems with business rules manager are: login error, login conflict, and access conflict.

Login error

Upon logging in, you receive a login error message.

The login error message:

Unable to process login. Please check User ID and password and try again.

This error occurs when global security is enabled and either the userid, the password, or both, are incorrect.

Note: Login errors occur only when global security is enabled.

1. Click **OK** on the error message.
You return to the login page.
2. Enter valid **User ID** and **Password**.
Make sure that Caps Lock key is not on, if passwords are case sensitive.
Make sure the userid and password are spelled correctly.
Check with the system administrator to see that the userid and password are correct.
3. Click the **Login** button.

If you resolve the login error, you will now be able to login to the business rules manager. If the error is not resolved, contact your system administrator.

Login conflict error

This event occurs when another user with the same userid is already logged in to the application.

The login conflict message is:

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

Usually this error occurs when a user closed the browser without logging out. When this condition occurs, the next attempted login before the session timeout expires results in a login conflict.

Note: Login conflict occurs only when global security is enabled.

There are three options that you can choose.

- Return to the login page.
Use this option if you want to open the application with a different userid.
- Logout the other user with the same userid.
Use this option to logout the other user and start a new session.

Note: Any unpublished local changes made in the other session are lost.

- Inherit the context of the other user with the same userid and logout that user.
Use this option to continue work already in progress. All unpublished local changes in the previous session that have been saved are not lost. The business rules manager opens to the last page displayed in the previous session.

Access conflict errors

Access conflicts occur when a business rule is updated in the database by one user at the same time another user is updating the same rule.

This error is reported when you publish your local changes to the database.

These are the actions to correct access conflict errors.

- Publish the Rule page.
- Find the source of the business rule that is causing the error and check if your changes on the local machine are still valid. Your change may no longer be required after the changes done by another user.
- If you choose to continue working in the business rule manager, you must reload Rule Pages in the error from the database as your local changes of Rule pages in error are no longer usable. You can still use local changes in other Rule pages that are not in error.
- Reload a Rule page, by clicking **Reload** in the Publish and Revert page of the rule for which the error was reported.

Troubleshooting the Common Base Event browser

There are four primary conditions under which you are unable to access the Common Base Event browser.

Conditions

“Cannot find server”

WebSphere Process Server (or network server) is unavailable. When you attempt to launch the event browser URI, a “Cannot find server” browser page will be returned, which indicates that the server is unavailable. In this case, you need to contact the IBM Help Desk to determine the cause of the problem.

“File not found”

WebSphere Process Server is available; however, the event browser

application may not be installed or started on the server. When you attempt to launch the event browser URI, a “File not found” browser page will be returned, which indicates that the server is available, but the URI is not available on that server. In this case, you need to contact the IBM Help Desk to determine the cause of the problem.

Logon panel appears

The WebSphere Process Server and the event browser are available; however, you have not been mapped to the proper role to allow access to the event browser. You will be prompted with a logon panel. When you enter your userID and password, attempting to log in, the login will fail. In this case, you need to contact the IBM Help Desk to get the proper authorization to launch the event browser.

Error message on “Get event data” panel

The WebSphere Process Server and the event browser are available, and you have the proper authority to gain access; however, the Common Event Infrastructure server is unavailable. An error message will be displayed on the event browser **Get Events** panel, when you click the **Get Events** button. The error information is logged to the message log.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

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
577 Airport Blvd., Suite 800
Burlingame, CA 94010
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries, or both: IBM, IBM (logo), AIX, CICS, Cloudscape, DB2, DB2 Connect, DB2 Universal Database, developerWorks, Domino, IMS, Informix, iSeries, Lotus, MQSeries, MVS, OS/390, Passport Advantage, pSeries, Rational, Redbooks, Tivoli, WebSphere, z/OS, zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

This product includes software developed by the Eclipse Project (<http://www.eclipse.org/>).



IBM Websphere Process Server for z/OS version 6.0.1



Printed in USA