

IBM WebSphere Application Server Enterprise,
Version 5.0.2



Process Choreographer

Note

Before using this information, be sure to read the general information under “Trademarks and service marks” on page vii.

Compilation date: July 17, 2003

© Copyright International Business Machines Corporation 2002, 2003. All rights reserved.

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

Contents

Trademarks and service marks vii

Chapter 1. Using Process Choreographer 1

Process Choreographer overview	2
Process Choreographer scenarios for clustering	3
Process Choreographer and network deployment	8

Chapter 2. Planning to use Process Choreographer 11

Chapter 3. Configuring the business process container 13

Creating the database for the business process container	13
Creating a DB2 database for Process Choreographer	15
Creating a DB2 for z/OS database for Process Choreographer	17
Creating an Oracle database for Process Choreographer	18
Creating a Cloudscape database for Process Choreographer	19
Creating a Sybase database for the Process Choreographer	20
Create a Microsoft SQL Server database for the Process Choreographer	21
Creating the queue manager and queues for the business process container	22
Creating clustered queue managers and queues for the business process container	23
Configuring the business process container on a cluster	27
Using the Install Wizard to configure the business process container	28
Business Process Container Install Wizard settings	29
Configuring the business process container manually	44
Configuring the JDBC provider and data source for the business process container	45
Configuring a DB2 JDBC provider and data source for the business process container	45
Configuring an Oracle JDBC provider and data source for the business process container	47
Configuring a Cloudscape JDBC provider and data source for the business process container	48
Configuring a Sybase JDBC provider and data source for the business process container	49
Configuring a Microsoft SQL Server JDBC provider and data source for the business process container	50
Configuring the queue resources for the business process container	51

Configuring queue resources for the business process container using the JMS provider embedded in WebSphere	52
Configuring the queue resources for the business process container using WebSphere MQ or MQSeries	53
Configuring MQ resources for the business process container using the Administrative Console	54
Creating and configuring the scheduler service for Process Choreographer	57
Installing the business process container	58
Business Process Container settings	59
Activating the business process container	61
Verifying that the business process container works	61
Troubleshooting the business process container	62

Chapter 4. Uninstalling the business process container 65

Chapter 5. Managing processes 67

Installing process applications	67
Uninstalling process applications	67
Stopping and starting process templates	68
Querying and replaying failed messages	68
Process modules collection	69
Process module settings	70
Process templates collection	70
Process template settings	70

Chapter 6. Using the Process Choreographer Web client 73

About the Process Choreographer Web client	74
Starting the Process Choreographer Web client	76
Working with work items	77
Displaying work items in your To Do list	77
Displaying information about a work item	78
Claiming a work item	78
Completing a work item	79
Querying work items	80
Working with process instances	80
Working with process instances you administer	81
Working with process instances you have started	81
Displaying information about a process instance	82
Monitoring a process instance	82
Administering compensation for a process instance	82
Terminating a process instance	83
Querying process instances	83
Working with process templates	84
Displaying information about a business process template	84
Starting a new process instance	84
Querying process templates	85
Customizing the Process Choreographer Web client	85

Adapting the look and feel	85
Layout of the Process Choreographer Web client user interface	86
Creating user-defined JSPs	87
Creating JSPs for displaying messages	88
Creating JSPs for processing user input	89
Integrating user-defined JSPs in the business process model	90
Message-mapping JSPs	91
Troubleshooting the Process Choreographer Web client	93
Process Choreographer Web client page directory	93
Activity page	94
Activity Information page	95
Administered By Me page	96
Compensation in Doubt page	97
Created By Me page	98
Define Process Instance List page	99
Define Process Template List page	99
Define Work Item List page	100
My Templates page	101
My To Dos page	101
Process Input Message page	102
Process Instance page	103
Process Instance Monitor page	104
Process Output Message page	104
Process Template page	105
User-Defined Process Instance List page	105
User-Defined Process Template List page	106
User-Defined Work Item List page	106
Process Choreographer Web client roles and actions	108

Chapter 7. Developing applications using the Process Choreographer API. 111

Accessing the Process Choreographer EJB interface	111
Accessing the Process Choreographer JMS interface	112
Developing applications for non-interruptible processes	113
Executing a non-interruptible process using the EJB interface	113
Executing a non-interruptible process using JMS	114
Characteristics of non-interruptible business processes	114
Developing applications for interruptible processes	115
Starting an interruptible process using the EJB interface	115
Processing person activities using the EJB interface	115
Sending an event to a process instance using the EJB interface	116
Analyzing results of a process using the EJB interface	116
Using worklists to query information	117
Starting an interruptible process using the JMS interface	117
Sending an event to a process instance using the JMS interface	118
Analyzing results of a process using the JMS interface	118
Characteristics of interruptible processes	119
Event activities	119

Person activities	119
Developing administration applications for interruptible processes	119
Canceling a claimed activity	120
Forcing the completion of an activity	120
Retrying the execution of a stopped activity	121
Deleting a process instance	121
Terminating a process instance using the EJB interface	122
Terminating a process instance using the JMS interface	122
Managing worklists	123
Authorization for EJB renderings	123
Required authorizations for process requests	124
Required authorizations for activity requests	124
Authorization for JMS renderings	125
Structure of a Process Choreographer JMS message	126
Queries on business-process objects	127
Predefined views for queries on business process objects	128
PROCESS_TEMPLATE view	128
PROCESS_ATTRIBUTE view	129
PROCESS_INSTANCE view	129
ACTIVITY view	130
ACTIVITY_ATTRIBUTE view	131
EVENT view	131
WORK_ITEM view	131
Select clause	132
Where clause	133
Order-by clause	134
Threshold parameter	134
Timezone parameter	134
Query results	135

Chapter 8. Configuring the staff service for Process Choreographer . . 137

About the staff service in Process Choreographer	139
Predefined staff verbs and their parameters	142
Troubleshooting the staff service and the staff plug-ins	148
Staff service settings	149
Startup	149
Staff plugin provider collection	149
Name	149
Description	150
Jar File	150
Staff plugin provider settings	150
Staff plugin configuration collection	150
Staff plugin configuration settings	151

Chapter 9. Using compensation in service choreography. 153

Chapter 10. Troubleshooting Process Choreographer. 155

Using process-related trace information	155
Using process-related audit trail information	155
BPEAuditLogDelete utility for Process Choreographer	155

Process Choreographer - structure of the audit trail database table 156
Process Choreographer - audit event types . . 158
Using process-related messages 159

**Chapter 11. Process Choreographer:
Resources for learning 161**

**Chapter 12. Process Choreographer
glossary 163**

Trademarks and service marks

The following terms are trademarks of IBM Corporation in the United States, other countries, or both:

- AIX
- CICS
- Cloudscape
- DB2
- Everyplace
- iSeries
- IBM
- Informix
- iSeries
- MQSeries
- OS/390
- Redbooks
- SupportPac
- ViaVoice
- VisualAge
- WebSphere
- zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product and service names may be trademarks or service marks of others.

Chapter 1. Using Process Choreographer

Process Choreographer provides the IBM WebSphere Application Server the ability to choreograph all kinds of business processes. These processes typically require both human and IT resources. The types of processes can vary greatly, ranging from Web services or Web page navigation to business transaction support. Processes can be automatic recoverable processes or processes that require human interaction.

With Process Choreographer you can combine business-process technology with the other services that are available in WebSphere, that is, services offered by the J2EE architecture. Process Choreographer can be used to script EJBs, allowing the manipulation of processes as EJBs. It allows you to create Web services consisting of processes of other Web services.

The Process Choreographer framework provides a meta-model based on the IBM Flow Composition Model (FCM), which is part of Studio Application Developer. For a detailed description of the Process Choreographer architecture, refer to the *Process Choreographer Concepts and Architecture* white paper in the WebSphere Developer Domain at <http://www7b.software.ibm.com/wsdd/zones/was/wpc.html>.

The Process Choreographer Web client provides a Web browser interface for work items involving people. For a high-level view of how Process Choreographer can be used to integrate your business processes, see “Process Choreographer overview” on page 2.

If you want to use Process Choreographer in a distributed environment, see “Process Choreographer and network deployment” on page 8 and “Process Choreographer scenarios for clustering” on page 3.

To define your business processes, and create process modules that can be installed in a WebSphere business process container, use the IBM WebSphere Studio Application Developer Integration Edition.

How to use, manage, and develop business-process applications is described in the following topics:

- Chapter 2, “Planning to use Process Choreographer”, on page 11
- Chapter 3, “Configuring the business process container”, on page 13
- Chapter 4, “Uninstalling the business process container”, on page 65
- Chapter 5, “Managing processes”, on page 67
- Chapter 6, “Using the Process Choreographer Web client”, on page 73
- Chapter 7, “Developing applications using the Process Choreographer API”, on page 111
- Chapter 8, “Configuring the staff service for Process Choreographer”, on page 137
- Chapter 9, “Using compensation in service choreography”, on page 153
- Chapter 10, “Troubleshooting Process Choreographer”, on page 155

You can find supplemental information about Process Choreographer listed in Chapter 11, “Process Choreographer: Resources for learning”, on page 161.

Process Choreographer overview

Process Choreographer is a powerful tool for executing complex business processes. In the paragraphs that follow, certain terms specific to Process Choreographer appear in **bold type** and are explained in context.

Each **process** is designed as a series of **activities**, and these activities are assigned as **work items** to various people in your organization. Work items can be almost any business task: filling out a form, approving a document or drawing, writing a letter, and so on. The process is the design for how a series of tasks is to be done. When a process is started (by someone in your organization or even anonymously by someone filling out a form at a Web site), certain information is recorded and passed on to the first activity. The associated work item then shows up in the worklists of the potential work item owners - all the people who are authorized to work on work item. In a typical implementation of Process Choreographer, everyone can look at the contents of a work item. They may also be able to see information about the process, such as who started it, when it is due, and so on.

When you **claim** a work item, you become the **owner** of that work item. Only you and the **business process administrator** can work on that particular work item in that particular instance of the process. You can save intermediate stages of a work item if it is complex or elaborate. When you have finished the work item, you **complete** the work item. The resulting information is saved and is then available to subsequent activities in the process. Navigation of the process continues until all activities have been completed.

Information about processes and work items in Process Choreographer appear as Web pages. If you know how to use a Web browser, you already know a lot about how Process Choreographer works.

Process Choreographer benefits

The benefits of Process Choreographer are many and diverse, depending on your organization's goals and the kinds of things it does. The following are just some examples.

Faster work completion

If your organization's processes are mostly paper-based, Process Choreographer can dramatically reduce the time it takes to accomplish tasks. Automating processes and moving documents electronically virtually eliminates the time required to move paper around your organization. In addition, the instructions and routing rules contained in the process definitions assure that problems are caught early and that repetitive tasks are automated.

Gains in productivity

Process Choreographer saves valuable time by helping you organize your work and set priorities. Work arrives with all the supporting documents. Your work may be listed by priority or due date, so you can work on the most important or most time-sensitive tasks first.

Improvements in process control

When procedures and rules live only inside the heads of workers, consistency and quality can suffer. Process Choreographer can be a powerful knowledge

management tool for standardizing your organization's processes while providing great flexibility for modifying and improving them.

Improvements in customer service

When your organization's processes are standardized and work is completed faster, your customers - internal and external - benefit. Processes can even be designed that allow customers to initiate work and track its progress. Process Choreographer is a cost-effective way to increase customer satisfaction. Your organization benefits from repeat business and customer retention.

More effective collaboration

Over time, the process definitions created under Process Choreographer can become a storehouse of "best practices" for the organization, which can be shared across the enterprise.

Process Choreographer scenarios for clustering

The main advantages of using WebSphere clusters and Network Deployment (ND) to create and administer Process Choreographer instances are:

- Increased workload capacity.
- Improved resource utilization.
- Workload sharing.
- Easier administration.

Configuration options

Process Choreographer can be configured in many different ways, so with cluster configurations, there are even more possibilities. Some of the main options to consider before you start creating application servers are described below:

Number of nodes in the WebSphere cell

One or more. All nodes are administered from a single deployment manager.

Number of nodes in each WebSphere cluster

One or more. Horizontal WebSphere clustering can increase service availability and increase the total workload capacity.

Number of application servers in each node

One or more. Vertical WebSphere clustering can increase resource utilization.

Database host

- Remote, on a dedicated machine.
- Local to one of the application servers in the cluster.

To be able to share workloads, all business process containers in the same WebSphere cluster must use the same database. It is recommended to host the database on a dedicated machine, preferably one with a hot standby.

Application messaging queues

- Local queues.
- Remote queues.

Connection (WebSphere MQ queue managers)

- One central (remote) queue manager hosting the queues for the application servers within one cluster.
- One local queue manager per application server.
- Two local queue managers per node, and WebSphere MQ clustering used to balance workload across several application servers.

Note: Workload balancing between different Process Choreographer instances requires that the queue managers used by each application server's Business Process Container are members of the same WebSphere MQ cluster.

Database system

You can use any of the supported databases except Cloudscape.

Hot standby machines

- None.
- For the database.
- For a central queue manager.

Note: Do not be confused. This topic refers to two different types of "cluster". A **WebSphere Cluster** groups application servers together to share workload and increase service availability. A **WebSphere MQ Cluster**, previously known as an MQSeries cluster, groups together WebSphere MQ queue managers and can be used to achieve intraprocess workload balancing.

High availability

To achieve high availability of Process Choreographer services, you should consider the following:

- By creating cloned application servers in a WebSphere cluster, the services provided by the application servers become highly available.
- The Process Choreographer database is a single point of failure that can be protected using a hot standby system.
- A central queue manager can be protected by hot standby hardware.

Vertical clustering to maximize resource utilization

Since Java cannot address more than 2 GB of memory, you might have to create multiple application server instances on the same node so that Process Choreographer can make full use of the available memory.

Workload balancing

If you want different instances of Process Choreographer to be able to share the same workloads, they must:

- Use the same Process Choreographer database.
- Use one of the following queue manager configurations:

Central queue manager

A central queue manager hosts the four queues that are needed by Process Choreographer. All Process Choreographer instances in the WebSphere cluster read from the same queues.

WebSphere MQ cluster

Each application server has two queue managers, one hosts four local queues, and is used for getting, the other hosts no queues and is only

used for putting. All the queue managers of all Process Choreographer instances in the WebSphere cluster are made members of a WebSphere MQ cluster. The result of only putting to queue managers that host no queues, is that the messages are distributed evenly across all the "get" queue managers in the cluster. After using the install wizard to install and configure the business process container on the cluster, you must manually change the two connection factories per application server to point to the local "get" and "put" queue managers.

For more details about WebSphere clustering, see *Balancing workloads with clusters* (not in this document).

Process Choreographer database

It is recommended to host the databases on a dedicated machine, preferably one with a hot standby. It can be on a machine that is outside the WebSphere cell, however the deployment manager must have access to all the databases.

It is also possible to host some or all of the Process Choreographer databases on the same machine as the application server, again, the deployment manager must have access to all the databases.

Things to keep in mind when planning the database are:

- All Business Process Containers in the same WebSphere cluster access the same database. By contrast, any Business Process Container that is not in a WebSphere cluster must have its own database.
- To enable access to a remote Process Choreographer database, you must install the appropriate database client on all application servers that do not have a local database.
- The Deployment Manager requires access to all databases for Process Choreographer instances in the WebSphere cell, regardless of whether they are in a cluster, or not. You must enable this access before you can use the deployment manager to install a business process.
- Your database can be any of the supported databases except Cloudscape, which does not support remote access.
- Each database, used by Process Choreographer instances in the same WebSphere cell, must have a unique name. The same database name must be used on the network manager as on the application server.
- The database is a single point of failure. This can only be solved by using a high-availability hot standby solution such as HACMP on AIX.

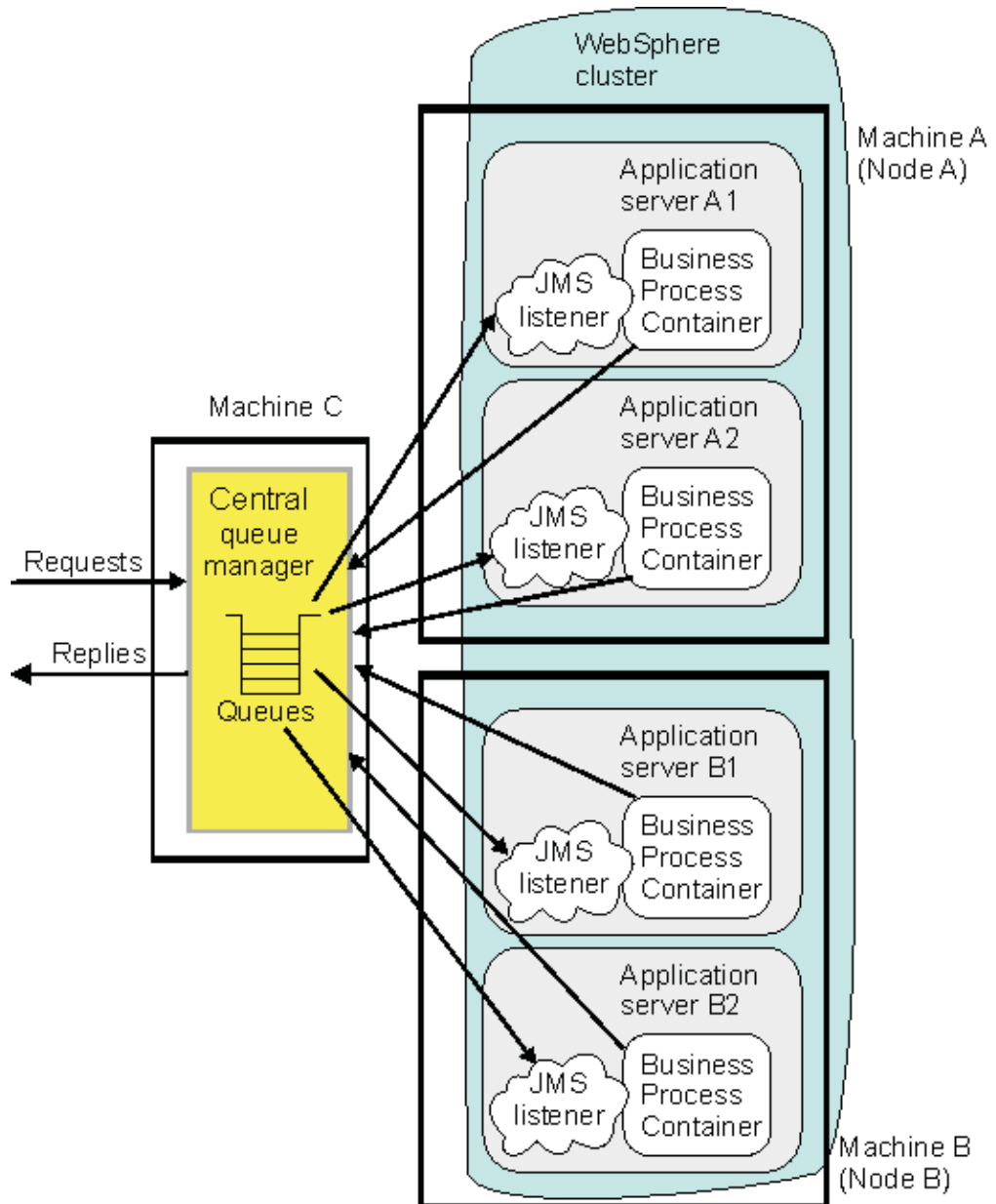
WebSphere MQ

Process Choreographer uses MQ queues for receiving requests and sending replies. Therefore each application server hosting Process Choreographer requires either:

- Access to a central queue manager that hosts all queues.
- A local queue manager that is not a member of a WebSphere MQ cluster.
- Two local queue managers, which are members of a WebSphere MQ cluster.

Note: Process Choreographer also supports the embedded messaging that comes with the Enterprise Application Server, however it is not recommended for complex configurations because it does not support WebSphere MQ clustering.

By using a central queue manager for all queues, administration becomes easier. This is because there is just one queue manager, and all cloned business process containers use it. However, using a central queue manager creates a single point of failure, which should therefore be hosted on a high availability system.



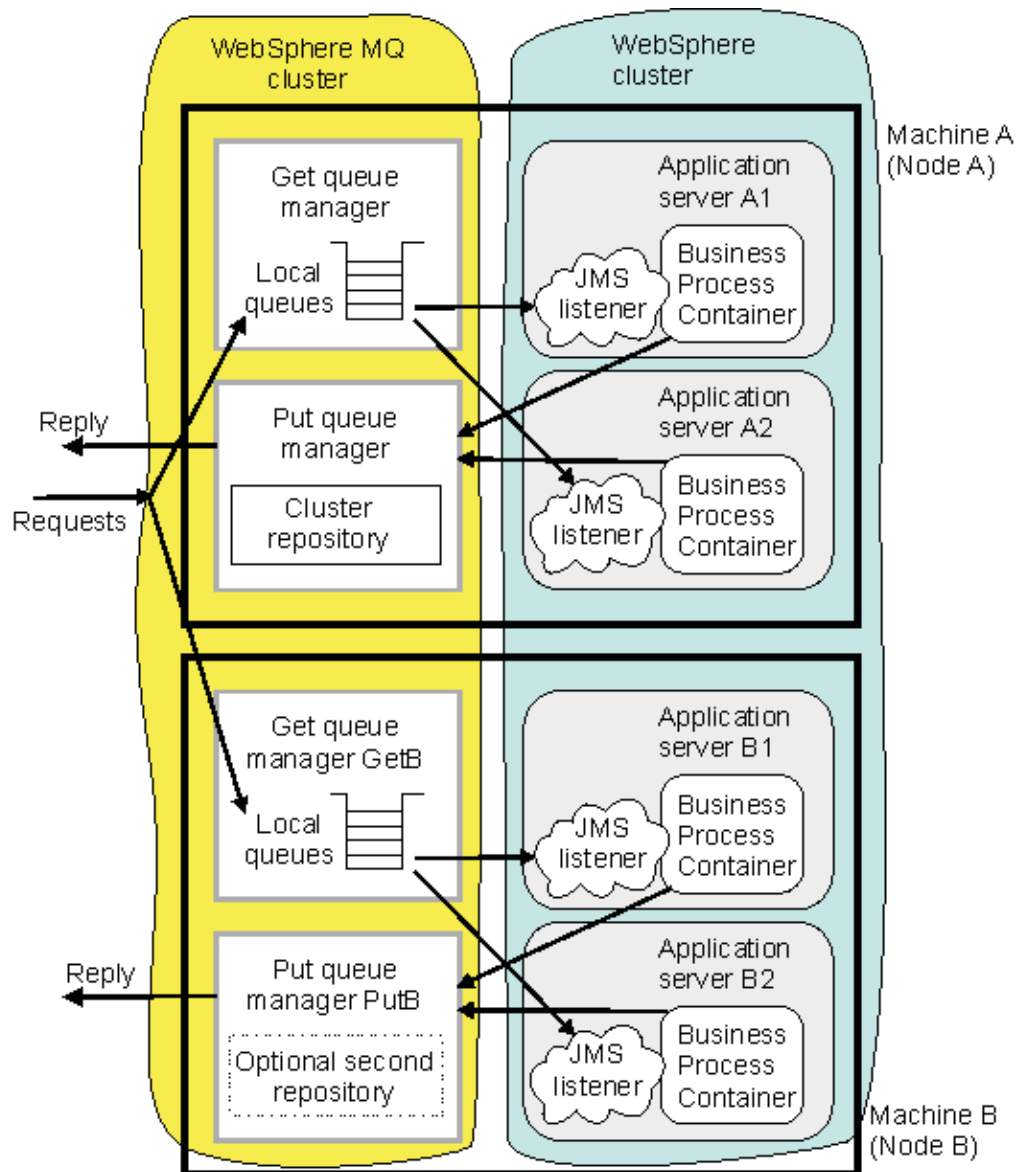
This is the standard, stand-alone Process Choreographer configuration. Each business process container uses its own (local or remote) database, and has one local queue manager. This approach does not offer intraprocess workload sharing, and is not described here.

This more complex technique allows intraprocess workload sharing for Process Choreographer services in a WebSphere cluster. The business processes in the cluster must all be running on either UNIX or Windows machines, not on a mixture.

Each application server requires two local queue managers, one for putting and one for getting. All the queue managers are made members of the same WebSphere MQ cluster. On Windows, the queue managers should all use the same binding protocol. On UNIX, the "put" and "get" queue managers must use different protocols, for example, by modifying the queue connection factories so that all put queue managers use the "Binding" protocol (inter-process communications) while all the get queue managers use the default "client" (TCP/IP) protocol.

Each Business Process Container in the WebSphere cluster must be customized to reflect its own queue managers.

It is recommended that more than one queue manager in the WebSphere MQ cluster is made a cluster repository.



The WebSphere MQ cluster (of queue managers) is parallel to the WebSphere cluster (of members).

How the WebSphere cluster is created

In theory, there are several different sequences that you could follow to create a cluster for Process Choreographer. The recommended sequence is:

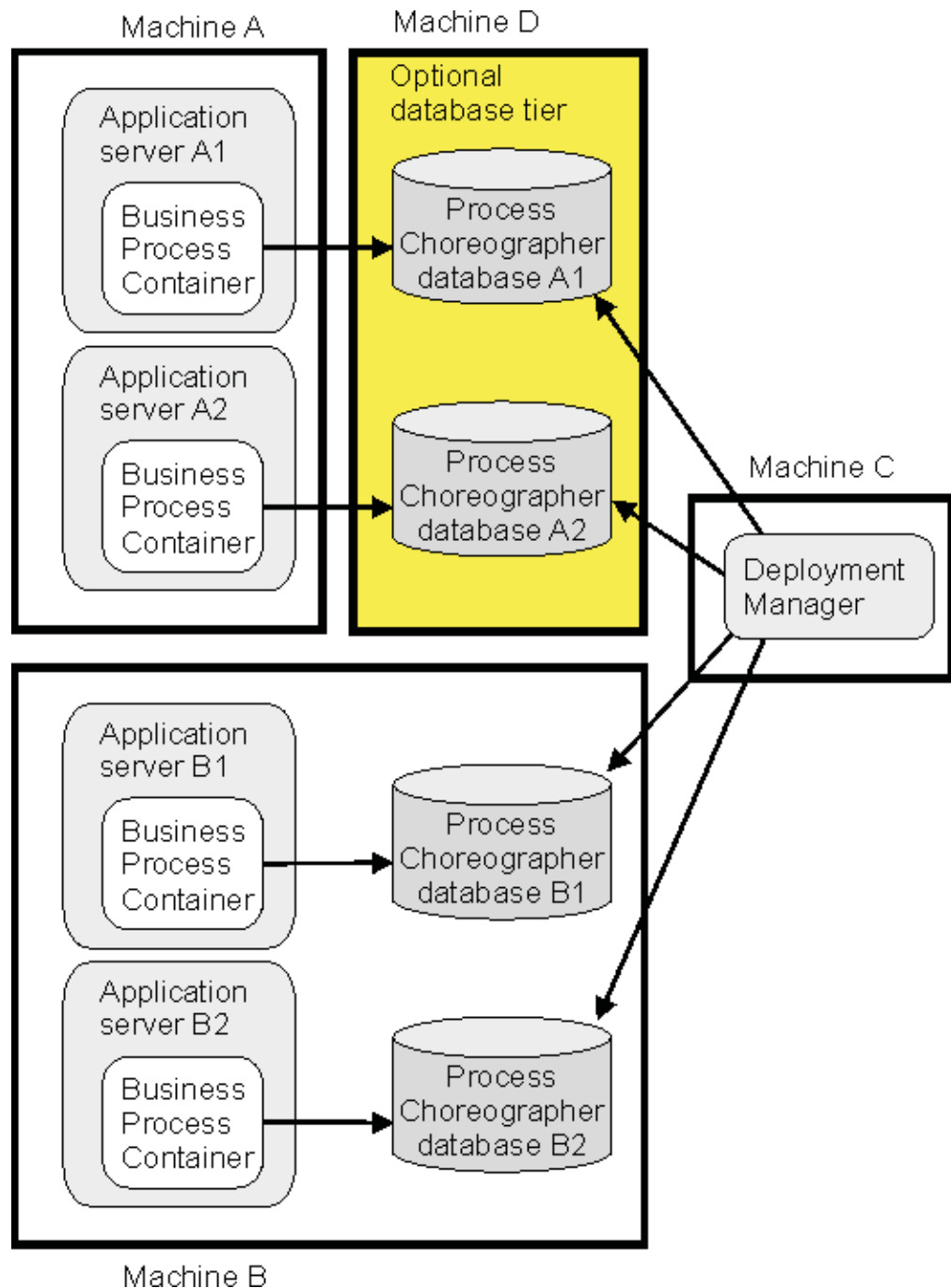
1. Install the prerequisites on each machine:
 - WebSphere MQ.
 - Database server or client.
2. On the first node in the cell, define the necessary application servers.
3. Create the required clones of the application server.
4. By using the Install Wizard on the Deployment Manager machine to install the business process container on any one of the application servers in the cluster, the business process container will be simultaneously installed on all the other application servers in the cluster.
5. If your WebSphere MQ configuration is a WebSphere MQ cluster of local queue managers, because each queue manager has a different name, you must modify the connection factories in each of the cloned application servers to reflect its unique differences from the cluster-wide, standard Process Choreographer Install Wizard configuration.

Process Choreographer and network deployment

The main advantage of using Network Deployment (ND) to create and administer Process Choreographer instances is that it makes clustering scenarios possible, where all the business process containers in a cluster are administered centrally.

Deployment manager must have access to the Process Choreographer database

The deployment manager must have access to all the Process Choreographer databases that are used by business process containers in the cell. You must install an appropriate database client on the deployment manager machine, and add the database driver to the deployment manager's classpath.



Customization required after installing and configuring Process Choreographer on a cluster

If you are creating a clustered setup that uses WebSphere MQ clusters of queue managers, you must perform some manual customization to make each Process Choreographer instance use its own queue managers. The necessary actions are described in Chapter 3, “Configuring the business process container”, on page 13.

For more information about using clustering with Process Choreographer, see “Process Choreographer scenarios for clustering” on page 3.

Installing a business process application

When you install a business process application, you add configuration data to the WebSphere configuration repository and process meta data to the Process Choreographer databases.

A business process application consists of at least one FAR file and at least one WAR file or EJB jar. You install these applications in the same way as other J2EE applications; use either the administration scripts or the application install windows on the administrative console. However, because FAR files are not J2EE modules, they do not appear on the application install windows.

Currently, all the FAR files belonging to your business process application are distributed to the application servers and WebSphere clusters where you install any of the EJB jars or WAR files belonging to your application. It is recommended that you install all your modules on the same application servers and WebSphere clusters. As a consequence, you must configure business process containers on all these application servers and WebSphere clusters.

If you want to separate WAR files from the FAR files of your application, put them in a different EAR file and install this file separately. You cannot separate EJB jars from FAR files.

When you install a process application, all the stand-alone servers and at least one application server of each WebSphere cluster where you want to install the application modules must be running. The corresponding database servers must also be running.

If only some of the required servers are available at installation time, install the application on the servers and WebSphere clusters that are running. You can map the application to further servers and WebSphere clusters later. See *Editing a business process application* for information on how to do this.

Restrictions when editing a business process application

When you edit a business process application, you change the mapping of the application modules to the application servers and WebSphere clusters in the cell. You can use the administrative console or administration scripts to change this mapping.

Only the EJB modules and Web modules appear in the administrative console. If you change the mapping of your EJB or Web modules, the mapping of the process modules (FAR files) is changed accordingly.

When you edit a business process:

- All application servers and at least one member of each WebSphere cluster that you want to change (remove or add) must be running. The associated database servers must also be running.
- If you remove the mapping of a module to an application server or a WebSphere cluster, then the process templates of that application must be stopped and all instances of the templates must be removed from the database belonging to the corresponding application server or WebSphere cluster.

Chapter 2. Planning to use Process Choreographer

For each application server where you want to use Process Choreographer, you must configure the business process container before installing any enterprise applications that contain processes. Before configuring the business process container, plan the following:

Steps for this task

1. To gain more understanding about the Process Choreographer architecture, refer to the architecture white paper in the WebSphere Developer Domain at <http://www7b.software.ibm.com/wsdd/zones/was/wpc.html>.

2. Which database system will you use?

One of:

- DB2
- DB2 for z/OS
- Cloudscape
- Oracle - using an OCI driver and the Oracle 32-bit library.
- Sybase ASE - since Process Choreographer uses distributed transactions, you will need to purchase and install the DTM feature for Sybase ASE.
- **5.0.2** Microsoft SQL Server

Check the required software levels at

<http://www.ibm.com/software/webservers/appserv/doc/latest/prereq.html>.

3. Decide which machine will host the database.

If the database machine is remote, you need a suitable database client.

4. If you are not are not going to create a WebSphere cluster setup, decide whether you want to use the JMS messaging service provided by:

- The messaging service that is embedded in WebSphere.
- A separate MQ (MQSeries or WebSphere MQ) installation.

Note: If you intend to use embedded messaging, you must have selected this option when you installed WebSphere. If you want to use MQ messaging, it must be installed before you start configuring the business process container. Embedded messaging cannot be used in a cluster setup.

5. If you intend to use Process Choreographer in a WebSphere cluster environment, plan your Process Choreographer cluster. For more information on how to plan clusters, see "Process Choreographer scenarios for clustering" on page 3.

6. Decide if you will use the Install Wizard (recommended) rather than configuring the business process container manually:

- If you are going to use the Install Wizard, plan the install Wizard settings. For more information on these settings, see "Business Process Container Install Wizard settings" on page 29.
- If you are going to configure the business process container manually, plan the business process container settings. For more information on these settings, see "Business Process Container settings" on page 59.

Results

You are ready to configure the business process container. For information on how to configure the business process container, see Chapter 3, “Configuring the business process container”, on page 13.

Chapter 3. Configuring the business process container

Before you begin

You must have completed planning to use Process Choreographer.

You must configure the necessary resources and install the business process container before you can use it.

Steps for this task

1. If you are preparing a clustered Process Choreographer setup, create the cluster.
2. If you are using an external JMS provider (WebSphere MQ or MQSeries), create the queue manager and queues.
3. Create the database.
4. If you are preparing a clustered Process Choreographer setup perform Configuring the business process container on a cluster.
5. If you are not preparing a clustered Process Choreographer setup, decide whether you want to configure the business process container using the **Install Wizard** (recommended) or **manually**, and perform one of:
 - Using the Install Wizard to configure the business process container.
 - Configuring the business process container manually.
6. Activate the business process container.
7. Verify that the business process container works.

In case of problems, see “Troubleshooting the business process container” on page 62.

Results

The business process container is configured and working.

What to do next

Now you can install and run enterprise applications that contain processes, as described in Chapter 5, “Managing processes”, on page 67.

Creating the database for the business process container

Before you begin

Your database system must already be installed and available. In a clustered Process Choreographer setup, one database serves all business process containers in the WebSphere cluster. In a non-clustered setup, the database is dedicated to the business process container on one application server.

The business process container requires a database.

Steps for this task

1. If your database server is not on the same machine as your Enterprise Application Server: Copy the ddl scripts for your database system to your database server machine.

On Windows, copy the ddl files from %WAS_HOME%\ProcessChoreographer. On UNIX, copy them from \$WAS_HOME/ProcessChoreographer.

2. If you will use a UNIX machine to host the database server, create a dedicated file system for the databases.
3. On the machine hosting the database server, create the database according to the description for your database system.
 - Creating a DB2 database.
 - Creating a DB2 for z/OS database.
 - Creating an Oracle database.
 - Creating a Cloudscape database.
 - Creating a Sybase database.
 - Creating an SQL Server database.
4. On each machine that will run Process Choreographer without a local database, you must make the remote database accessible:
 - a. Install a suitable database client on the Enterprise Application Server machine.
 - b. Make the new database known to the database client.

For DB2

the database must be cataloged and accessed via an alias name.

For Oracle

The TCP net service name (TNS) is used to access the database.

For Sybase and MS SQL Server

The node that hosts the database is used as the key to access the database.

- c. Verify that you can access the remote database using the client.
5. If you are preparing a clustered Process Choreographer setup, or you intend to use Network Deployment, you must also make the database accessible to the Deployment Manager. On the Deployment Manager machine, perform the following actions:
 - a. Install a suitable database client on the Enterprise Application Server machine.
 - b. Make the new database known to the database client:

For DB2

the database must be cataloged and accessed via an alias name.

For Oracle

The TCP net service name (TNS) is used to access the database.

For Sybase

The node that hosts the database is used as the key to access the database.

- c. Locate and copy the appropriate database driver. For the driver's name, refer to the default values for Classpath (data source) for your database.
 - d. Add the database driver to the classpath of the Deployment Manager.
 - e. Restart the deployment manager.

Results

The Process Choreographer database exists.

Creating a DB2 database for Process Choreographer

Steps for this task

1. Change to the directory where the configuration scripts for Process Choreographer are located:
 - If your database server is on the **same** machine as your Enterprise Application Server: On Windows, enter: `cd %WAS_HOME%\ProcessChoreographer`. On UNIX, enter: `cd $WAS_HOME/ProcessChoreographer`.
 - If your database server is on a **different** machine than your Enterprise Application Server, change to the directory where you copied the ddl scripts.
2. Install DB2 UDB on the machine that will host the database.
3. Install the DB2 runtime client on:
 - All remote application servers that will use the database server.
 - On the Deployment Manager machine if you will use ND to administer Process Choreographer, for example if you are creating a clustered Process choreographer setup.
4. If you want to use an existing database, to create the tablespace and schema skip to step 10.

Note: Make sure that the database supports Unicode (UTF-8). Without Unicode support, it cannot store all characters that can be handled in Java, and you may run into code page conversion problems when a client uses an incompatible code page.

5. If you are using DB2 Version 7.2, to avoid deadlocks, be sure that the DB2 flag `DB2_RR_TO_RS` is set to YES. If necessary, enter the command:
`db2set DB2_RR_TO_RS=YES`

then restart the DB2 instance to activate the change.

6. If you want to administer the DB2 instance remotely, create an administrative DB2 instance, `db2as`.
7. Create a DB2 instance on the database machine. The default is `db2inst1`.
8. If you have an SMP machine, set the number of processors that can be used by DB2.

On UNIX, use `/usr/lpp/db2_07_01/adm/db2licm -l`.

9. If you want to create a new database named `BPEDB`:
 - a. Make sure that you are using a user ID that has administrator rights for the database system.
 - b. In the DB2 command line processor, enter the command to run the quick database creation script:

```
db2 -tf createDatabaseDb2.ddl
```

- c. Make sure that the script's output contains no errors.

In some cases, the CLI packages are not bound to the new database. To be sure that this happens, for a database named `BPEDB`:

On Windows, enter:

```
db2 connect to BPEDB
db2 bind %DB2PATH%\bnd\db2cli.lst blocking all grant public
```

On UNIX, enter:

```
db2 connect to BPEDB
db2 bind $DB2DIR/bnd/@db2cli.lst blocking all grant public
```

A DB2 database named BPEDB has been created.

10. To create the tablespace and schema:

- a. **(Optional)** Analyze the results of your experiences during development and system testing.

The size of your database depends on many factors. Non-interruptible processes require very little space. Each process template may require tens or hundreds of kilobytes. If possible, distribute tablespace containers across different logical disks, and implement an appropriate security policy. Consider the performance implications of your choices for buffer pools and log file settings.

- b. Edit the tablespace creation script `createTablespaceDb2.ddl` according to the instruction at the top of the file.
- c. Make sure that you have administrator rights for the database system. The user ID you use to create the schema must be the one that you specify for WebSphere to use to access the database.
- d. Make sure that you are attached to the correct instance. Check the environment variable `DB2INSTANCE`.
- e. To connect to a database named *databaseName*, in the DB2 command line processor, enter the command:

```
db2 connect to databaseName
```

- f. To create the tablespace, enter the command:

```
db2 -tf createTablespaceDb2.ddl
```

Make sure that the script's output contains no errors. If there were any errors, you can drop the tablespace using the script `dropTablespaceDb2.ddl`.

- g. To create the schema (tables and views), in the DB2 command line processor, enter the command:

```
db2 -tf createSchemaDb2.ddl
```

Make sure that the script's output contains no errors. If there were any errors, you can use `clearSchemaDb2.ddl` to clear the schema, and `dropSchemaDb2.ddl` to drop the schema.

11. On each application server that will remotely access the database (and on the deployment manager machine if you are creating a clustered setup or if you want to use Network Deployment):

- a. Catalog the database by entering the command:

```
db2 catalog database databaseName as databaseAlias at node nodeName
```

For more information about cataloging a database refer to the DB2 documentation.

- b. Verify that you can connect to the database by entering the commands:

```
db2 connect to databaseName user userID
db2 connect reset
```

Results

The DB2 database for Process Choreographer exists.

Creating a DB2 for z/OS database for Process Choreographer

This topic describes how to create a DB2 z/OS database and how to verify that it is reachable from the application server machine. WebSphere Application Server must already be running on a UNIX or Windows machine.

Steps for this task

1. On the z/OS machine that will host the database:
 - a. Log on the native z/OS environment.
 - b. If multiple DB2 systems are installed, decide which subsystem you want to use.
 - c. Note which IP port the DB2 subsystem is listening to.
 - d. Using the DB2 administration menu, create a new database and note the name of the database.
 - e. Create a storage group and note the name.
 - f. Decide which user ID will be used to connect to the database from the remote machine running WebSphere Application Server. Normally, for security reasons, this user ID should not be the one you used to create the database.
 - g. Grant the user ID the rights to access the database and storage group. It must also be permitted to create new tables for the database.
2. On the Application Server machine:
 - a. Make sure that you have DB2 Connect Gateway (Version 7.2 FP 8 or higher) installed. It is part of the DB2 UDB EE package, but it can also be installed separately.
 - b. Catalog the remote database using the following commands, either in a script or in a DB2 command line window:

```
catalog tcpip node zosnode remote hostname server IP_port ostype mvs;  
catalog database subsystem as subsystem at node zosnode authentication dcs;  
catalog dcs database subsystem as subsystem parms ',,INTERRUPT_ENABLED'
```

Note: There is an important difference between DB2 UDB and DB2 for z/OS. DB2 UDB does not have the concept of subsystem, but DB2 for z/OS does. To avoid confusion between 'Database name' and 'Subsystem name', it is important to understand that because DB2 for z/OS runs in a subsystem, the catalog node and catalog database commands must identify the appropriate subsystem. On DB2 UDB, the subsystem name is not a known concept, so the 'database'; that it connects to is really the DB2 for z/OS subsystem.

- c. Verify that you can establish a connection to the remote subsystem by entering the following command:

```
db2 connect to <i>subsystem</i> user <i>userid</i> using  
<i>password</i>
```
- d. Change to the *ProcessChoreographer* subdirectory in the WAS install root directory.
- e. Edit the script createTablespaceDb2V**<i>x</i>**z0s.dd1 (where **<i>x</i>** is either 6 or 7, depending on the DB2 for z/OS version number). Replace **@STG@** with the storage group name and replace **@DBNAME@** with the database (not subsystem) name.

- f. Run your customized version of createTablespaceDb2V <i>x</i> z0s.ddl, as described in the script's header.
Note: If this does not work, you can find a script in the same directory for dropping the table space.
- g. Edit the script createSchemaDb2V <i>x</i> z0s.ddl (where *x* is either 6 or 7, depending on the DB2 for z/OS version number). As you did for the previous script, replace @STG@ with the storage group name and replace @DBNAME@ with the database (not subsystem) name.
- h. Run your customized version of createSchemaDb2V <i>x</i> z0s.ddl, as described in the script's header.
Note: If this does not work, you can find scripts in the same directory for clearing and dropping the schema.
- i. To avoid deadlocks, be sure that the DB2 flag DB2_RR_TO_RS is set to YES. If necessary, restart the DB2 instance to activate the change.

Results

The DB2 for z/OS database for Process Choreographer exists.

Creating an Oracle database for Process Choreographer

There is no script to quickly create a default Oracle database for Process Choreographer.

Steps for this task

1. Install the Oracle server on the machine that will host the database. Be sure that you are using OCI drivers and the 32-bit Oracle library lib32.
2. For the root user, set the environment variables ORACLE_BASE and ORACLE_HOME.
3. Install the database client on:
 - All remote application servers that will use the database server.
 - On the Deployment Manager machine if you are using ND to administer Process Choreographer, for example if you are creating a clustered Process choreographer setup.
4. On UNIX, create soft links to the following Oracle libraries in the /usr/lib directory:
 - For Oracle 8i: Link to: libwtc8.so, libclntsh.so.8.0, and libocijdbc8.so.
 - For Oracle 9i: Link to: libwtc9.so, libclntsh.so.9.0, and libocijdbc9.so.
5. Create an Oracle database using the Database Configuration Assistant. Make sure that you select the JServer option for the database. It is recommended that you use a Unicode codepage when creating the database. The text data you pass to the APIs must be compatible with the selected code page.
6. Change to the directory where the configuration scripts for Process Choreographer are located:
 - If your database server is on the **same** machine as your Enterprise Application Server: On Windows, enter: cd %WAS_HOME%\ProcessChoreographer. On UNIX, enter: cd \$WAS_HOME/ProcessChoreographer.
 - If your database server is on a **different** machine than your Enterprise Application Server, change to the directory where you copied the ddl scripts.

7. Edit the tablespace creation script according to the instructions at the top of the file:
 - For Oracle 8i: Edit createTablespaceOracle8.ddl
 - For Oracle 9i: Edit createTablespaceOracle9.ddl
8. Make sure that you are using the user ID that has administrator rights for the database system.
9. If you do not want the schema to be created in the default instance, set the environment variable ORACLE_SID.
10. To create the tablespace, run the script createTablespaceOracleX.ddl, where 'X' is your Oracle version digit (8 or 9).

For test purposes you can use the same location for all tablespaces and pass the path as command line argument to the script, for example, on Windows, using Oracle 8i, user ID bpeuser, password bpepwd, database name BPEDB, and tablespace path d:\mydb\ts, enter:

```
sqlplus bpeuser/bpepwd@BPEDB @createTablespaceOracle8.ddl d:\mydb\ts
```

If you get any errors creating the tablespace, you can use dropTablespaceOracleX.ddl to drop the tablespace, where 'X' is your Oracle version digit (8 or 9).

11. To create the schema, run the script createSchemaOracleX.ddl, where 'X' is your Oracle version digit (8 or 9).

For example, on Windows using Oracle 9i, enter:

```
sqlplus bpeuser/bpepwd@BPEDB @createSchemaOracle9.ddl
```

If you get any errors creating the schema (tables and views), you can use clearSchemaOracleX.ddl to clear the schema, and dropSchemaOracleX.ddl to drop the schema.

Results

The Oracle database for Process Choreographer exists.

Creating a Cloudscape database for Process Choreographer

Cloudscape is a database system implemented in Java. It comes with WebSphere Application Server Enterprise as three JAR files (db2j.jar, db2jtools.jar, and db2jcvview.jar). The Cloudscape license that comes with WebSphere is only for development and test, not for production purposes.

To create a Cloudscape database named BPEDB:

Steps for this task

1. Make sure that you have administrator rights.
2. Either change to the directory where you want the new database to be created, or edit the script createDatabaseCloudscape.ddl located in the ProcessChoreographer subdirectory of the IBM WebSphere Application Server installation directory, and add the fully qualified path to the database name, for example, on Windows change the connect command line to:

```
connect "d:\db\BPEDB;create=true" as BPEDB
```

Note: Cloudscape only allows one local connection. If WebSphere Application Server is running and accessing a Cloudscape database, attempting to open a second connection to the database from the command line will be rejected.

3. At the command prompt, enter the command to run the database creation script using the Cloudscape command line processor:

On Windows, enter:

```
java -Djava.ext.dirs=%WAS_HOME%\cloudscape50\lib
-Dij.protocol=jdbc:db2j: com.ibm.db2j.tools.ij
%WAS_HOME%\ProcessChoreographer\createDatabaseCloudscape.ddl
```

On UNIX, enter:

```
java -Djava.ext.dirs=$WAS_HOME/cloudscape50/lib
-Dij.protocol=jdbc:db2j: com.ibm.db2j.tools.ij
$WAS_HOME/ProcessChoreographer/createDatabaseCloudscape.ddl
```

Results

The Cloudscape database for Process Choreographer exists.

Creating a Sybase database for the Process Choreographer

Steps for this task

1. Make sure that you have administrator rights.
2. Install the Sybase server, with the DTM option, on the machine that will host the database.
3. Install the database client on:
 - All remote application servers that will use the database server.
 - On the Deployment Manager machine if you are using ND to administer Process Choreographer, for example if you are creating a clustered Process choreographer setup.
4. Make sure that you have configured and enabled the DTM option for Sybase ASE:
 - a. Set enable DTM to 1 in the Sybase server configuration.
 - b. Set enable xact coordination to 1 in the Sybase server configuration.
 - c. Add the role dtm_tm_role to the Sybase administration user ID, for example, user ID sa.
 - d. Restart the Sybase server.
5. Change to the directory where the configuration scripts for Process Choreographer are located:
 - If your database server is on the **same** machine as your Enterprise Application Server: On Windows, enter: cd %WAS_HOME%\ProcessChoreographer. On UNIX, enter: cd \$WAS_HOME/ProcessChoreographer.
 - If your database server is on a **different** machine than your Enterprise Application Server, change to the directory where you copied the ddl scripts.
6. If you want to create a non-production database for stand-alone development, evaluation, or demo purposes, you only need to enter the command:

```
isql -S serverName -U userid -P password -i createDatabaseSybase120.ddl
```

A Sybase database named BPEDB has been created.

7. If you prefer to create your database manually:
 - a. Inspect the options used in the quick database creation script `createDatabaseSybase120.ddl`, and make sure that you include these in the database you create.
 - b. Create the database.
 - c. Run the script to create the schema, by entering the command:

```
isql -S serverName -U userid -P password -D databaseName -i createSchemaSybaseversion.ddl
```

Where

serverName

is the name of the Sybase server as defined in the `dsedit` tool.

userid is the user ID to be used.

password

is the password for *userid*.

databaseName

is the name of the database.

version

is 120 for Sybase 12.0, and 125 for Sybase 12.5.

Results

The Sybase database for Process Choreographer exists.

Create a Microsoft SQL Server database for the Process Choreographer

Steps for this task

1. Make sure that you have administrator rights.
2. Change to the directory where the configuration scripts for Process Choreographer are located:
 - If your database server is on the **same** machine as your Enterprise Application Server, enter: `cd %WAS_HOME%\ProcessChoreographer`.
 - If your database server is on a **different** machine than your Enterprise Application Server, change to the directory where you copied the `ddl` scripts.
3. Install the SQL Server, on the machine that will host the database.
4. Make sure that the database server and the Distributed Transaction Coordinator are running.

Note: To enable XA connections, refer to Vendor-specific data sources minimum required settings for instructions on how to install "Stored Procedures for JTA" from the WebSphere CD.

5. If you want to create a non-production SQL Server database for stand-alone development, evaluation, or demo purposes, you only need to enter one of the following commands:

For SQL Server 7:

```
isql -U userid -P password -i createDatabaseMsSql7.ddl
```

For SQL Server 2000:

```
isql -U userid -P password -i createDatabaseMsSql2000.ddl
```

An SQL Server database named `BPEDB` has been created.

6. If you prefer to create your database manually:
 - a. Create the database, for example, named BPEDB.
 - b. Run the script to create the schema, by entering one of the following commands:

For SQL Server 7:

```
isql -S serverName -U userid -P password -d databaseName -i createSchemaMsSql7.dd1
```

For SQL Server 2000:

```
isql -S serverName -U userid -P password -d databaseName -i createSchemaMsSql2000.dd1
```

Where

userid is the user ID to be used.

password
is the password for *userid*.

databaseName
is the name of the database you created, for example, BPEDB.

Results

The SQL Server database for Process Choreographer exists.

Creating the queue manager and queues for the business process container

If you are using WebSphere MQ or MQSeries as an external JMS provider, you must create the queue manager and queues.

Steps for this task

1. If you are not creating a WebSphere cluster setup, perform the following:
 - a. Make sure that your user ID has the authority to create WebSphere MQ queues.
 - b. Create the queue manager and queues:

On Windows, enter:

```
cd %WAS_HOME%\ProcessChoreographer  
createQueues.bat queueManager
```

On UNIX, enter:

```
cd $WAS_HOME/ProcessChoreographer  
createQueues.sh queueManager
```

Where

queueManager

is the name of an existing queue manager, or the name to be given to a new queue manager. If the named queue manager already exists, it will be used to create the queues. If the queue manager does not exist, it will be created and started before the default queues are created.

2. If you are creating a WebSphere cluster setup that uses a WebSphere MQ cluster, only perform Creating clustered queue managers and queues.

3. If you are creating a WebSphere cluster setup that uses a central queue manager, perform the following:
 - a. Copy the create queues script file from the WebSphere machine to the machine that will host the central queue manager:

On Windows, copy the file:

```
%WAS_HOME%\ProcessChoreographer\createQueues.bat
```

On UNIX, copy the file:

```
$WAS_HOME/ProcessChoreographer/createQueues.sh
```

- b. On the machine that will host the queue manager, make sure that WebSphere MQ is installed, and that your user ID has the authority to create WebSphere MQ queues.
- c. Create the queue manager and queues:

On Windows, enter:

```
cd %WAS_HOME%\ProcessChoreographer
createQueues.bat queueManager
```

On UNIX, enter:

```
cd $WAS_HOME/ProcessChoreographer
createQueues.sh queueManager
```

Where

queueManager

is the name to be given to the new queue manager.

- d. Add a listener for the new queue manager by entering the command:

```
runmqclsr -t tcp -p port -m queueManager
```

Where *port* is the port that it will be listen to.

- e. On UNIX, add definitions for the port and queue manager service:
 - 1) Add the port for the queue manager to the `/etc/services` file:

```
<Service:Name> <port>/tcp
<Service:Name>    name of the queue manager service
<port>            port for the queue manager
```

- 2) Add the same service to the `/tc/inetd.conf` file:

```
<Service:Name> stream tcp nowait mqm /usr/mqm/bin/amqcrsta amqcrsta -m QueueManager
<Service:Name>    name of the queue manager service (the same as in /etc/services)
<Service:Name>    name of the queue manager
```

Results

The queue manager and queues exist.

Creating clustered queue managers and queues for the business process container

If you are creating a WebSphere cluster setup of Process Choreographer using a WebSphere MQ cluster, you must create the queue managers, queues, cluster, repositories, channels, and listeners.

Steps for this task

1. If your WebSphere cluster consists of **UNIX** nodes, perform the following on each node:

- a. Make sure that your user ID has the authority to create WebSphere MQ queues.
- b. Create the "get" and "put" queue managers, make them members of the WebSphere MQ cluster, and create the queues by entering the commands:

```
cd $WAS_HOME/ProcessChoreographer
createQueues.sh getQueueManager clusterName putQueueManagerName
```

Where

getQueueManager

The unique name to be given to the "get" queue manager. It will host all the local queues.

clusterName

the name of the WebSphere MQ cluster.

putQueueManager

The unique name to be given to the "put" queue manager.

If the queue managers already exist they will be used. If they do not exist they will be created and used.

- c. Start the WebSphere MQ command processor by entering the command:

```
runmqsc getQueueManager
```

- d. **(Optional)** For complex setups, it is recommended to allow remote administration of the queue manager by entering the following MQ command:

```
DEFINE CHANNEL('SYSTEM.ADMIN.SVRCONN') TYPE(CHLTYPE)
```

- e. If this queue manager is to be a repository for the WebSphere MQ cluster enter the MQ command:

```
ALTER QMGR REPOS('clusterName') REPOSNL(' ')
```

- f. Define a sender and receiver channel for the queue manager to each repository that is not hosted on this machine by entering the following MQ commands:

For each cluster receiver channel:

```
DEFINE CHANNEL('TO.repositoryQueueManager.TCP') +
  CHLTYPE(CLUSRCVR) +
  CLUSTER('clusterName') +
  CLUSNL(' ') +
  CONNAME('repositoryIP-Address(port)') +
  DESCR('Cluster receiver channel at repositoryQueueManager TCP/IP') +
  MAXMSGL(4194304) +
  TRPTYPE(TCP) +
  MCAUSER('principal') +
  REPLACE
```

For each cluster sender channel:

```
DEFINE CHANNEL('TO.repositoryQueueManager.TCP') +
  CHLTYPE(CLUSSDR) +
  CONNAME('repositoryIP-Address(port)') +
```



```

CLUSTER('clusterName') +
CLUSNL(' ') +
DESCR('Cluster sender channel to repositoryQueueManager TCP/IP') +
MAXMSGL(4194304) +
TRPTYPE(TCP) +
MCAUSER('targetPrincipal') +
REPLACE +
NPMSPEED (NORMAL)

```

Where

repositoryQueueManager

The name of the queue manager hosting a repository.

clusterName

The name of the WebSphere MQ cluster that all the queue managers are a member of.

repositoryIP-Address

The IP address of the node where the repository queue manager

port The IP port that the repository queue manager is using.

principal, targetPrincipal

The MCAUSER to be used for the receive and send channels. For more information about this refer to the WebSphere MQ documentation.

- g. For each queue manager, start a listener by entering the MQ command:

```
runmqclsr -t tcp -p port -m QueueManager
```

2. If your WebSphere cluster consists of **Windows** nodes, perform the following on each node:

- a. Make sure that your user ID has the authority to create WebSphere MQ queues.
- b. Create the "get" queue manager, make it a member of the WebSphere MQ cluster, and create the queues by entering the commands:

```
cd %WAS_HOME%\ProcessChoreographer
createQueues.bat queueManager clusterName putQueueManager
```

Where

getQueueManager

The unique name to be given to the "get" queue manager.

clusterName

the name of the WebSphere MQ cluster that the WebSphere cluster nodes will use.

putQueueManager

The unique name to be given to the "put" queue manager.

If the queues already exist they will be used. If they do not exist they will be created and used.

- c. Start the WebSphere MQ command processor by entering the command:

```
runmqsc queueManager
```

- d. **(Optional)** For complex setups, it is recommended to allow remote administration of the queue manager by entering the following MQ command:

```
DEFINE CHANNEL('SYSTEM.ADMIN.SVRCONN') TYPE(CHLTYPE)
```

- e. If this queue manager is to be a repository for the WebSphere MQ cluster enter the MQ command:

```
ALTER QMGR REPOS('clusterName') REPOSNL(' ')
```

- f. Define a sender and receiver channel for the queue manager to each repository that is not hosted on this machine by entering the following MQ commands:

For each cluster receiver channel:

```
DEFINE CHANNEL('TO.repositoryQueueManager.TCP') +  
  CHLTYPE(CLUSRCVR) +  
  CLUSTER('clusterName') +  
  CLUSNL(' ') +  
  CONNAME('repositoryIP-Address(port)') +  
  DESCR('Cluster receiver channel at repositoryQueueManager TCP/IP') +  
  MAXMSGL(4194304) +  
  TRPTYPE(TCP) +  
  MCAUSER('principal') +  
  REPLACE
```

For each cluster sender channel:

```
DEFINE CHANNEL('TO.repositoryQueueManager.TCP') +  
  CHLTYPE(CLUSSDR) +  
  CONNAME('repositoryIP-Address(port)') +  
  CLUSTER('clusterName') +  
  CLUSNL(' ') +  
  DESCR('Cluster sender channel to repositoryQueueManager TCP/IP') +  
  MAXMSGL(4194304) +  
  TRPTYPE(TCP) +  
  MCAUSER('principal') +  
  REPLACE +  
  NPMSPEED (NORMAL)
```

Where

repositoryQueueManager

The name of the queue manager hosting a repository.

clusterName

The name of the WebSphere MQ cluster that all the queue managers are a member of.

repositoryIP-Address

The IP address of the node where the repository queue manager

port The IP port that the repository queue manager is using.

principal

The MCAUSER to be used. For more information about this refer to the WebSphere MQ documentation.

- g. For each queue manager, start a listener by entering the MQ command:

```
runmqtsr -t tcp -p port -m QueueManager
```

3. To verify the status of the channels on a machine, enter the MQ command:

```
display chstatus(*)
```

Results

The queue managers, queues, cluster, repositories, channels, and listeners exist.

Configuring the business process container on a cluster

If you are preparing a clustered Process Choreographer setup, you can install and configure the business process container on a WebSphere cluster, but you must also customize the connection factories separately for each application server in the cluster.

Steps for this task

1. Using the Process Choreographer Install Wizard on the deployment manager machine, install and configure the business process container on any application server in the cluster. This causes the business process container to be installed and configured on all the other application servers in the cluster.
2. You must modify the queue connection factory definitions one-by-one. For each application server in the WebSphere cluster:
 - a. Select **Resources > JMS > WebSphere MQ JMS Providers**.
 - b. Click **WebSphere MQ Queue Connection Factory**.
 - c. Select the connection factory `BPECF` and set the property values for the type of queue manager configuration you are using:
 - For a central queue manager:

Host	The hostname of the machine that is hosting the central queue manager.
Port	The port number that the central queue manager is using.
Transport Type	Client
Client ID	The MCA user ID to be used.
CCSID	On AIX and Solaris: 819. On Windows: 437

- For a cluster of queue managers:

Host	The hostname of the application server's node.
Transport Type	Binding

- d. Select the connection factory `BPEFC` and set the property values for the type of queue manager configuration you are using:
 - For a central queue manager:

Host	The hostname of the machine that is hosting the central queue manager.
Port	The port number that the central queue manager is using.
Transport Type	Client
Client ID	The MCA user ID to be used.
CCSID	On AIX and Solaris: 819. On Windows: 437

- For a cluster of queue managers on UNIX:

Host	The hostname of the application server's node.
Port	The port number used by this application server's "put" queue manager.
Transport Type	Client
Client ID	The MCA user ID to be used.
CCSID	On AIX and Solaris: 819.

- For a cluster of queue managers on Windows:

Host	The hostname of the application server's node.
Transport Type	Binding

Results

The business process containers have been installed in the cluster and are configured.

What to do next

Now you can activate the business process containers.

Using the Install Wizard to configure the business process container

You must configure the necessary resources and install the business process container before you can use it. This topic describes how to do so using the Install Wizard.

Steps for this task

1. Make sure that you are logged on with a user ID with sufficient administration rights. On Windows, use the user ID that will be used to start WebSphere.
2. Select **Servers > Application Servers > *serverName***

Where *serverName* is the name of the application server where you want to install the business process container. In a cluster, you can select any application server, and the business process container will be installed simultaneously on all application servers in the cluster.

Note: When installed on a non-clustered application server, the name of the business process container will be `BPEContainer_<i>serverName</i>`, on a cluster it will be named `BPEContainer_<i>clusterName</i>`.
3. In the **Additional Properties** section, click **Business Process Container**.
4. Scroll down, past the Business Process Container settings. Near the bottom of the page click on the link for the **Business Process Container Install Wizard**.

Note: Where possible, the Install Wizard offers appropriate default values in the parameter fields. However, with some combinations of browser and platform no defaults are provided. In this case, you can view the recommended values on the Install Wizard settings page.
5. Select the database configuration:
 - a. Select **Create a new XA datasource**.
 - b. In the drop-down list, select the database you are using.
 - c. For the **Implementation class name** use the default class name provided for the JDBC driver implementation.
 - d. If you are using a Cloudscape database, skip to step 5h.
 - e. For **Classpath** enter the location of the Java archive or zip file for the data source.
 - f. The **Data source user name** must be a user ID that has the authority to create and administer the database.
 - g. Enter the **Data source password** for the Data source user name.
 - h. Make sure that the options in the **Custom properties** field match your database requirements. For more information, see the Install Wizard settings page and the product documentation for your database system.
 - i. Click **Next** to go to the next step in the Install Wizard.
6. Select the JMS provider and security configuration:

- a. In the drop-down list for **JMS provider**, select the messaging service that the business process container will use.
 - b. For the **Queue manager**, use the default provided (WASQM or WAS_<i>hostname</i>_<i>server</i>).
 - c. If you are using external messaging (WebSphere MQ or MQSeries) and you have not defined the WebSphere environment variable `{MQ_INSTALL_ROOT}`, make sure that the **Classpath** points to the MQ Java lib directory.
 - d. For the **JMS user ID**, enter a user ID that has administration rights for the messaging service. On UNIX, use root. On Windows, use the default.
 - e. For the **JMS password**, enter the password for the JMS user ID.
 - f. For the **Scheduler calendar** field, if you have your own scheduler calendar, enter its JNDI name. Otherwise, if you leave it blank, the default value, BPEScheduler, will be used.
 - g. For the **Security role mapping**, enter the user or group from your user registry that is to be mapped onto the role of Business Process Administrator.
 - h. For the **JMS API user ID**, enter the user ID that is to be used when processing asynchronous API calls.
 - i. For the **JMS API password**, enter the password for the JMS API User ID.
 - j. Click **Next** to go to the next step in the Install Wizard.
7. To select the JMS resources, either select **Create new JMS resources using default values** and click **Next**, or perform the following:
 - a. Select **Select existing JMS resources**.
 - b. Use the **Connection Factory** drop-down list to select BPECF.
 - c. Use the **Internal queue** drop-down list to select BPEIntQueue.
 - d. Use the **External request processing queue** drop-down list to select BPEApiQueue.
 - e. Use the **Hold queue** drop-down list to select BPEHldQueue.
 - f. Use the **Retention Queue** drop-down list to select BPERetQueue.
 - g. Click **Next**.
 8. Check that the information on the summary page is correct.

Note: The summary includes reminders of which external resources are necessary. If you have not already created them, you can continue configuring the business process container, but you must create the resources before you activate the business process container. Printing the summary page will help you to create the correct resources.

 - a. To make corrections, click **Previous**.
 - b. To install the business process container and define its resources click **Finish**.
 9. The progress is shown on the **Installing** page:
 - a. If the container did not install successfully, check for any error messages that can help you correct the problem, then repeat this task from step 1.
 - b. If the container was installed successfully, click **Save Master Configuration**, then click **Save**.

Business Process Container Install Wizard settings

Use the Install Wizard to install and configure the business process container.

This page describes the Install Wizard's fields, in the order that they appear in the Install Wizard.

Step 1 database configuration:

XA data source
 Implementation class name
 Classpath (data source)
 Data source user name
 Data source password
 Custom properties

Step 2 JMS provider and security:

JMS provider
 Queue manager
 Classpath (JMS provider)
 JMS user ID
 JMS password
 Scheduler calendar
 Security role mapping
 JMS API user ID
 JMS API password

Step 3 JMS resources:

JMS resources (new or existing)
 Connection factory
 Internal queue
 External request processing queue
 Hold queue
 Retention queue

Note: These fields cannot be changed after they have been applied. If you want to change any of these values after the process container has been configured, you must uninstall the business process container before reinstalling and configuring it again.

XA data source

If the data source does not already exist, you must create a new one.

Mandatory	Yes
Data type	Radio buttons and drop-down lists
Choices	<ul style="list-style-type: none"> • Select an existing XA data source. • Create a new XA data source: <ul style="list-style-type: none"> – Cloudscape – DB2 – Sybase 12 – Sybase 12.5 – Oracle V8 – Oracle V9 – DB2 z/OS – SQL Server 2000 – SQL Server 7

Implementation class name

The Java class name of the JDBC driver implementation.

Mandatory	Yes
-----------	-----

Data type	String
Default	<p>For Cloudscape com.ibm.db2j.jdbc.DB2XDataSource</p> <p>For DB2 com.ibm.db2.jdbc.DB2XDataSource</p> <p>For DB2 z/OS com.ibm.db2.jdbc.DB2XDataSource</p> <p>For Oracle oracle.jdbc.xa.client.OracleXDataSource</p> <p>For Sybase com.sybase.jdbc2.jdbc.SybXDataSource</p> <p>For SQL Server com.ibm.websphere.jdbcx.sqlserver.SQLServerDataSource</p>

Classpath (data source)

The path to the Java archive or zip file for the data source. If the database is remote, this is the path on the client machine.

Mandatory	<p>For Cloudscape No</p> <p>For DB2, DB2 z/OS, Oracle, Sybase, and SQL Server Yes</p>
Data type	String

Default	<p>For DB2 and DB2 z/OS The path depends on the DB2 installation root directory, typical defaults are:</p> <p>On AIX /home/db2inst1/sql1lib/java12/db2java/db2java.zip</p> <p>On Solaris /export/home/db2inst1/sql1lib/java12/db2java/db2java.zip</p> <p>On Windows c:\Program Files\SQLLIB\java\db2java.zip</p> <p>Note: You can only use the variable <code>{DB2_INSTALL_ROOT}</code> in the path if has been defined as a WebSphere variable.</p> <p>For Oracle The path depends on the Oracle home directory.</p> <p>On Windows <code><i>OracleHome</i>\jdbc\lib\classes12.zip</code></p> <p>On UNIX <code><i>OracleHome</i>/jdbc/lib/classes12.zip</code> where <i>OracleHome</i> is normally something like <code>/opt/oracle/product/9.2.0</code></p> <p>For Sybase The path depends on the Sybase home directory.</p> <p>On Windows <code><i>SybaseHome</i>\jConnect-5_2\classes\jconn2.jar</code></p> <p>On UNIX <code><i>SybaseHome</i>/jConnect-5_2/classes/jconn2.jar</code> where <i>SybaseHome</i> is typically <code>/opt/sybase</code></p> <p>For SQL Server</p> <p>On Windows <code><i>WAS_LIBS_DIR</i>\sqlserver.jar, <i>WAS_LIBS_DIR</i>\base.jar, <i>WAS_LIBS_DIR</i>\util.jar, <i>WAS_LIBS_DIR</i>\spy.jar</code></p> <p>On UNIX <code><i>WAS_LIBS_DIR</i>/sqlserver.jar, <i>WAS_LIBS_DIR</i>/base.jar, <i>WAS_LIBS_DIR</i>/util.jar, <i>WAS_LIBS_DIR</i>/spy.jar</code> where <code><i>WAS_LIBS_DIR</i></code> is the <code><samp>libs</code> subdirectory of your WebSphere Application Server installation directory.</p>
---------	--

Data source user name

A user ID that has the authority to create and administer the database.

Mandatory	<p>For Cloudscape No</p> <p>For DB2, DB2 z/OS, Oracle, Sybase, and SQL Server Yes</p>
Data type	String
Default	<p>For DB2 on UNIX Can vary depending on your system. Typically, the instance owner, for example, db2inst1.</p> <p>For DB2 on Windows Can vary depending on your system. Typically, the instance owner, for example, db2admin.</p> <p>For DB2 on z/OS A valid DB2 user ID that has administration rights.</p> <p>For Oracle The default is system. Note: When an Oracle database is created, the user system with the password manager is automatically created and granted all system privileges for the database.</p> <p>For Sybase The default is sa. Note: At installation time, the Sybase system administrator's login name is sa. It is not a UNIX user ID, it is specific to Adaptive Server, and is used with the <code>isql</code> command to log into Adaptive Server.</p>

Data source password

The password for the data source user name.

Mandatory	<p>For Cloudscape No</p> <p>For DB2, DB2 z/OS, Oracle, Sybase, and SQL Server Yes</p>
Data type	String
Default	None

Custom properties

Extra parameters that are required by the database system.

Mandatory	Yes
Data type	String
Data format	Multiple lines of <code><i>Property</i>=<i>Value</i></code>

Properties for Cloudscape	<p>databaseName=\${WAS_INSTALL_ROOT}/ProcessChoreographer/BPEDB Required string. Defines which database to access.</p> <p>remoteDataSourceProtocol=<i>protocol</i> Optional string. If the database is remote and the data source accesses the database using a client, set this property to specify which client/server protocol to use. Currently, the only protocol supported is rmi.</p> <p>shutdownDatabase=shutdown Optional string. If set to 'shutdown', the database will shutdown when a java.sql.Connection object is obtained from the data source. For example, if the data source is an XADataSource, a <code>getXAConnection().getConnection()</code> is necessary to cause the database to shutdown.</p> <p>dataSourceName=<i>name</i> Optional string. Name for the ConnectionPooledDataSource or XADataSource. Not used by the data source object. Only used for information purposes.</p> <p>description=<i>description</i> Optional string. Description of the data source. Not used by the data source object. Only used for information purposes.</p> <p>connectionAttribute=<i>attributes</i> Optional string. Connection attributes specific to Cloudscape. Refer to the Cloudscape documentation for a complete list of attributes.</p> <p>createDatabase=create Optional string. If set to 'create', and the database specified in the databaseName parameter does not already exist, it will be created. The database will be created when a connection object is obtained from the data source.</p> <p>enableMultithreadedAccessDetection=false Optional boolean. If set to true, it will automatically detect multithreaded access to a connection and its corresponding Statements, ResultSets, and MetaDatas.</p> <p>preTestSQLString=<i>test string</i> Optional string. If you have enabled pre-test connection in <code>j2c.properties</code>, this pre-test string is used to make sure that the connection is good. For example, <code>select count(*) from testTable</code>. If no string is specified, the default, <code>SELECT COUNT(*) FROM rra.x1x1x0x4x</code>, will be used.</p>
---------------------------	---

<p>Properties for DB2</p>	<p>databaseName=<i>name</i> Required string. Defines which database to access, for example, BPEDB.</p> <p>description=<i>description</i> Optional string. Description of the data source. Not used by the data source object. Only used for information purposes.</p> <p>portNumber=<i>port</i> Optional integer. Specifies the TCP/IP port number where the JDBC provider resides.</p> <p>connectionAttribute=<i>attributes</i> Optional string. Connection attributes specific to DB2. Refer to the DB2 documentation for a complete list of features. For example, connectionAttribute=cursorhold=0.</p> <p>loginTimeout=0 Optional integer. If set to zero, there will be no timeout. Non-zero values specify the maximum number of seconds allowed to establish a connection to the database.</p> <p>enableMultithreadedAccessDetection=false Optional boolean. If set to true, it will automatically detect multithreaded access to a connection and its corresponding Statements, ResultSets, and MetaDatas.</p> <p>preTestSQLString=<i>test string</i> Optional string. If you have enabled pre-test connection in <code>j2c.properties</code>, this pre-test string is used to make sure that the connection is good. For example, <code>select count(*) from testTable</code>. If no string is specified, the default, <code>SELECT COUNT(*) FROM rra.x1x1x0x4x</code>, will be used.</p> <p>enableSQLJ=false Optional boolean. This value is used to indicate whether SQLJ operations may be performed with this data source. If enabled, this data source can be used for both JDBC and SQLJ calls. Otherwise, only JDBC usage is permitted. This flag should always be set to false for DB2 v7.2, which does not support SQLJ.</p>
---------------------------	---

Properties for DB2 z/OS	<p>databaseName=<i>subsystemName</i> Required string. Defines which subsystem contains the DB2 z/OS database.</p> <p>description=<i>description</i> Optional string. Description of the data source. Not used by the data source object. Only used for information purposes.</p> <p>portNumber=<i>port</i> Optional integer. Specifies the TCP/IP port number where the JDBC provider resides.</p> <p>connectionAttribute=<i>attributes</i> Optional string. Connection attributes specific to DB2. Refer to the DB2 documentation for a complete list of features. For example, connectionAttribute=cursorhold=0.</p> <p>loginTimeout=0 Optional integer. If set to zero, there will be no timeout. Non-zero values specify the maximum number of seconds allowed to establish a connection to the database.</p> <p>enableMultithreadedAccessDetection=false Optional boolean. If set to true, it will automatically detect multithreaded access to a connection and its corresponding Statements, ResultSets, and MetaDatas.</p> <p>preTestSQLString=<i>test string</i> Optional string. If you have enabled pre-test connection in <code>j2c.properties</code>, this pre-test string is used to make sure that the connection is good. For example, <code>select count(*) from testTable</code>. If no string is specified, the default, <code>SELECT COUNT(*) FROM rra.x1x1x0x4x</code>, will be used.</p> <p>enableSQLJ=false Optional boolean. This value is used to indicate whether SQLJ operations may be performed with this data source. If enabled, this data source can be used for both JDBC and SQLJ calls. Otherwise, only JDBC usage is permitted.</p>
-------------------------------	---

Properties for Oracle	<p>driverType=<i>driverType</i> Required string. Defines the type of the driver. Possible values are: thin or oci8.</p> <p>databaseName=<i>name</i> Optional string. Defines which database to access, for example, BPEDB.</p> <p>serverName=<i>hostname</i> Optional string. The name of the server where Oracle resides.</p> <p>portNumber=<i>port</i> The TCP/IP port number where the JDBC driver resides. Used with the thin driver setup. The default during the installation of Oracle is 1521.</p> <p>networkProtocol=<i>protocol</i> Optional string. Specifies which protocol is used, for example, TCP/IP or IPC.</p> <p>dataSourceName=<i>name</i> Optional string.</p> <p>URL=<i>url</i> When driverType=thin, this is a required string, otherwise it is optional. The URL specifies the database from which the data source will obtain connections. For example, 'jdbc:oracle:thin:@localhost:1521:sample'.</p> <p>TNSEntryName=<i>name</i> Required string. The entry name used for the Oracle OCI driver. For example, BPEDB.</p> <p>loginTimeout=0 Optional integer. If set to zero, there will be no timeout. Non-zero values specify the maximum number of seconds allowed to establish a connection to the database.</p> <p>description=<i>description</i> Optional string. Description of the data source. Not used by the data source object. Only used for information purposes.</p> <p>enableMultithreadedAccessDetection=false Optional boolean. If set to true, it will automatically detect multithreaded access to a connection and its corresponding Statements, ResultSets, and MetaDatas.</p> <p>preTestSQLString=SELECT USER FROM DUAL Optional string. If you have enabled pre-test connection in j2c.properties, this pre-test string is used to make sure that the connection is good. For example, SELECT USER FROM DUAL. If no string is specified, the default, SELECT COUNT(*) FROM rra.x1x1x0x4x, will be used.</p>
-----------------------	---

<p>Properties for Oracle (continued)</p>	<p>oraclelogPrintMask=62 Optional integer. The oraclelogPrintMask controls which information is printed with each trace message. Oracle9i requires the use of classes12_g.zip. Default is 62 which is ([OracleLog.FIELD_OBJECT for 9i / OracleLog.FIELD_CONN for 8i] 32 OracleLog.FIELD_CATEGORY 16 OracleLog.FIELD_SUBMOD 8 OracleLog.FIELD_MODULE 4 OracleLog.FIELD_TIME 2). Possible values: OracleLog.FIELD_TIME 2, OracleLog.FIELD_MODULE 4, OracleLog.FIELD_SUBMOD 8, OracleLog.FIELD_CATEGORY 16, OracleLog.FIELD_OBJECT 32, OracleLog.FIELD_THREAD 64.</p> <p>oraclelogModuleMask=1 Optional integer. The oraclelogModuleMask controls which modules write debug output. Oracle9i requires the use of classes12_g.zip. Default is 1 which is (OracleLog.MODULE_DRIVER 1). Possible values (OracleLog.MODULE_DRIVER 1, OracleLog.MODULE_DBACCESS 2)“</p> <p>oraclelogCategoryMask=47 Optional integer. The oraclelogCategoryMask controls which category to be output. Oracle9i requires the use of classes12_g.zip. Default is 47 which is (OracleLog.USER_OPER 1 OracleLog.PROG_ERROR 2 OracleLog.ERROR 4 OracleLog.WARNING 8 OracleLog.DEBUG1 32). Possible values (OracleLog.USER_OPER 1, OracleLog.PROG_ERROR 2, OracleLog.ERROR 4, OracleLog.WARNING 8, OracleLog.FUNCTION 16, OracleLog.DEBUG1 32, OracleLog.SQL_STR 128)“</p> <p>transactionBranchesLooselyCoupled=false Optional boolean. This property is introduced as a result of Oraclebug 2511780, Oracle Patch for 2511780 must be installed before setting this property to true, failure to do that would cause a program error.</p>
--	--

Properties for Sybase	<p>databaseName=<i>name</i> Required string. Defines which database to access, for example, BPEDB.</p> <p>serverName=<i>hostname</i> Required string. The name of the server where Sybase resides.</p> <p>portNumber=<i>port</i> Required integer. The TCP/IP port number where the JDBC driver resides. The default during the installation of Sybase is 4100.</p> <p>networkProtocol=<i>protocol</i> Optional string. Specifies which protocol is used, for example, socket or SSL. When set to socket, SSL encryption is not used.</p> <p>dataSourceName=<i>name</i> Optional string. Name for the data source. Only used for information purposes.</p> <p>version=<i>number</i> Optional integer. The version number of the driver.</p> <p>resourceManagerName=<i>name</i> Optional string. The name of the resource manager.</p> <p>loginTimeout=0 Optional integer. If set to zero, there will be no timeout. Non-zero values specify the maximum number of seconds allowed to establish a connection to the database.</p> <p>description=<i>description</i> Optional string. Description of the data source. Not used by the data source object. Only used for information purposes.</p> <p>connectionProperties=<i>properties</i> Required string. Properties required for connection. As a minimum you should include connectionProperties=SELECT_OPENS_CURSOR=true. Refer to the Sybase documentation for more information about this and other properties.</p> <p>enableMultithreadedAccessDetection=false Optional boolean. If set to true, it will automatically detect multithreaded access to a connection and its corresponding Statements, ResultSets, and MetaDatas.</p> <p>preTestSQLString=<i>test string</i> Optional string. If you have enabled pre-test connection in <code>j2c.properties</code>, this pre-test string is used to make sure that the connection is good. For example, <code>select count(*) from testTable</code>. If no string is specified, the default, <code>SELECT COUNT(*) FROM rra.x1x1x0x4x</code>, will be used.</p>
-----------------------	--

Properties for SQL Server	<p>databaseName=<i>name</i> Required string. Defines which database to access, for example, BPEDB.</p> <p>serverName Required string. This is a required property. The TCP/IP address of the SequeLink server in dotted format or host name format.</p> <p>portNumber=1433 Required integer. The TCP/IP port number where the MS SQL Server resides.</p> <p>selectMethod=direct Optional string. Determine whether or not Microsoft SQL Server 'server cursors' are used for SQL queries. Values are 'cursor' or 'direct'. The default is 'direct'. See the ConnectJDBC documentation for more information.</p> <p>dataSourceName=<i>name</i> Optional string. Name for the data source. Only used for information purposes.</p> <p>spyAttributes=<i>attributes</i> Optional string. The SPY attributes. See the ConnectJDBC documentation for a list of attributes.</p> <p>loginTimeout=0 Optional integer. When non-zero, specifies the maximum time to attempt to connect to the database. Zero implies no limit.</p> <p>description=<i>description</i> Optional string. Description of the data source. Not used by the data source object. Only used for information purposes.</p> <p>enable2Phase=true Required boolean. Process Choreographer requires two-phase connections. Do not change this value.</p> <p>maxPooledStatements=0 Optional integer. The maximum number of pooled PreparedStatements for this connection. The default is 0.</p> <p>sendStringParametersAsUnicode="" Optional boolean. Determines whether string parameters are sent to the SQL Server database as Unicode or in the default character encoding of the database. For more information, refer to the ConnectJDBC documentation.</p> <p>enableMultithreadedAccessDetection=false Optional boolean. If set to true, it will automatically detect multithreaded access to a connection and its corresponding Statements, ResultSets, and MetaDatas.</p> <p>preTestSQLString=<i>test string</i> Optional string. If you have enabled pre-test connection in <code>j2c.properties</code>, this pre-test string is used to make sure that the connection is good. For example, <code>select count(*) from testTable</code>. If no string is specified, the default, <code>SELECT COUNT(*) FROM rra.x1x1x0x4x</code>, will be used.</p>
---------------------------	--

JMS provider

Specifies which messaging service the business process container will use.

Mandatory	Yes
-----------	-----

Data type	Drop-down list
Choices	<p>For external WebSphere MQ WebSphere MQ JMS Provider</p> <p>For the messaging embedded in WebSphere WebSphere JMS Provider</p>

Queue manager

The name of the queue manager that is to be used by the business process container.

Mandatory	Yes
Data type	String
Value	<p>For external WebSphere MQ Your queue manager name, for example, WAS_<i>nodeName</i>_<i>server</i>.</p> <p>For embedded messaging Use the default: WAS_<i>nodeName</i>_<i>server</i>.</p>

Classpath (JMS provider)

The path to the MQ Java lib directory.

Mandatory	<p>For embedded messaging The WebSphere environment variable <code>\${MQ_INSTALL_ROOT}</code> will already be defined, and you must not enter a classpath here.</p> <p>For external WebSphere MQ This is only mandatory if you have not defined the WebSphere environment variable <code>\${MQ_INSTALL_ROOT}</code> to point to the WebSphere MQ installation root directory.</p>
Data type	String
Default	<p>The default value for the classpath depends on the local WebSphere MQ installation:</p> <p>For AIX <code>/usr/mqm/java/lib</code></p> <p>For Solaris <code>/opt/mqm/java/lib</code></p> <p>For Windows <code>C:\Program Files\MQSeries\Java\lib</code></p>

JMS user ID

Used to authenticate the connection to the JMS provider. This user ID must have administration rights for the messaging service.

Mandatory	Yes
Data type	String
Restrictions	If you are using embedded messaging, the JMS user ID must not be longer than 12 characters. For example, the default Windows NT user ID, Administrator, is not valid for use with WebSphere's embedded messaging, because it contains 13 characters.
Default	The user ID you used to log into the administrative console.

For UNIX	Use root. The user ID must be a member of the group mqm.
For Windows	Use the default user ID. This must be the same user ID used to start WebSphere.

JMS password

The password for the JMS user ID.

Mandatory	Yes
Data type	String
Default	None

Scheduler calendar

JNDI name of the scheduler UserCalendar to be used by the business process container.

Mandatory	No
Data type	String
Default	BPEScheduler

Security role mapping

The user or group from the domain's user registry that is to be mapped onto the role of Business Process Administrator.

Mandatory	Yes
Data type	String
Default	<p>For UNIX Depends on the local settings. AIX example: Administrator</p> <p>For Windows Administrators</p>
Restrictions	The user registry can be the local operating system, LDAP, or custom registry. The user or group specified must already exist in the user registry being used.

JMS API user ID

The user ID that the Process Choreographer MDB will use when processing asynchronous API calls.

Mandatory	If WebSphere security is enabled (even if you will not use the JMS API).
Data type	String

Description	<p>If WebSphere security is enabled and you will not use the JMS API, you must specify a valid user ID - it does not need any special authorizations.</p> <p>If WebSphere security is enabled and you will use the JMS API, this user ID must either be one that is given the appropriate authorities when the process is modelled, or more commonly, it must be a member of a group that has been granted the necessary authorities during modeling. The possible staff authorities associated with processes are: Administrator, Reader, and Starter. For activities, a user ID can only perform sendEvent if it is a potential owner of the associated receiveEvent.</p> <p>If you want to allow all actions on processes via the JMS API, you can specify a user ID that is a member of the J2EE role BPESystemAdministrator, however, in a production system, the more fine-grained security approach is recommended.</p>
-------------	--

JMS API password

The password for the JMS API User ID.

Mandatory	If WebSphere security is enabled (even if you will not use the JMS API)
Data type	String

JMS resources (new or existing)

You must either create new JMS resources or select existing JMS resources.

Mandatory	Yes
Data type	Radio buttons
Choices	<ul style="list-style-type: none"> Create new JMS resources using default values <ul style="list-style-type: none"> Connection factory: BPECF Internal queue BPEIntQueue External request processing queue BPEApiQueue Hold queue BPEHldQueue Retention queue BPERetQueue Select existing JMS resources

Connection factory

The queue connection factory to be used by the business process container.

Mandatory	Only if you chose Select existing JMS resources
Data type	Drop-down list
Default	BPECF

Internal queue

The JNDI name of the queue for internal business process container messages.

Mandatory	Only if you chose Select existing JMS resources
Data type	Drop-down list
Default	BPEIntQueue

External request processing queue

The JNDI name of the queue for external (API) requests to the business process container.

Mandatory	Only if you chose Select existing JMS resources
Data type	Drop-down list
Default	BPEApiQueue

Hold queue

The JNDI name of the queue that is to hold any messages that cannot be processed by the business process container.

Mandatory	Only if you chose Select existing JMS resources
Data type	Drop-down list
Default	BPEHoldQueue

Retention queue

JNDI name of the queue that contains messages that temporarily could not be processed, and will be retried.

Mandatory	Only if you chose Select existing JMS resources
Data type	Drop-down list
Default	BPERetQueue

Configuring the business process container manually

Before you can use the business process container, you must install it and configure the necessary resources. This topic describes how to do this manually.

Steps for this task

1. Configuring the JDBC provider and data source.
2. Configuring the queue resources.
3. Creating and configuring the scheduler.
4. Installing the business process container.

Results

The business process container has been installed and configured.

What to do next

You must activate the business process container.

Configuring the JDBC provider and data source for the business process container

The business process container requires a JDBC provider and data source.

Steps for this task

1. On your Enterprise Application Server machine, configure your JDBC provider and data source according to the description for your database system:
 - Configuring a DB2 JDBC provider and data source.
 - Configuring an Oracle JDBC provider and data source.
 - Configuring a Cloudscape JDBC provider and data source.
 - Configuring a Sybase JDBC provider and data source.
 - Configuring an SQL Server JDBC provider and data source.

Results

The Process Choreographer database exists and the JDBC provider and data source have been defined.

What to do next

Configure the queue resources.

Configuring a DB2 JDBC provider and data source for the business process container

Steps for this task

1. Change to the Process Choreographer directory:

On Windows, enter:

```
cd %WAS_HOME%\ProcessChoreographer
```

On UNIX, enter:

```
cd $WAS_HOME/ProcessChoreographer
```

2. Run the configuration script.

On Windows, enter:

```
%WAS_HOME%\bin\wsadmin -f create_ds_Db2.jacl  
[-node <nodeName>] -server <serverName> -dbPath <databasePath>  
[-dbName <databaseName>] -userid <userid> -passwd <password>
```

On UNIX, enter:

```
$WAS_HOME/bin/wsadmin.sh -f create_ds_Db2.jacl  
[-node <nodeName>] -server <serverName> -dbPath <databasePath>  
[-dbName <databaseName>] -userid <userid> -passwd <password>
```

Where:

wsadmin

is located in the bin subdirectory of the IBM WebSphere Application Server installation directory.

create_ds_Db2.jacl

is the script file that performs the configuration within WebSphere. It is located in the ProcessChoreographer subdirectory of the IBM

WebSphere Application Server installation directory. For example, on a default Windows installation, it is in the directory `c:\WebSphere\AppServer\ProcessChoreographer`.

nodeName

is optional node name.

serverName

is the application server name.

databasePath

is the location where your DB2 system is installed. Note: On Windows you must use forward slashes (/) instead of back slashes (\) to specify the path, for example, "c:/sqllib".

databaseName

is the optional database name.

userid is the user ID for accessing the database. This must be the same user ID that was used to create the database.

password

is the password for *userid*.

3. If you had problems with step 2, you can configure the JDBC provider and data source using the administrative console:
 - a. In the Administrative Console, click **Resources > JDBC Providers**.
 - b. Click **New**.
 - c. Enter your values for the following fields:

Name	BPEJdbcDriverDB2
Description	JDBC Provider for Process Choreographer
Classpath	e:/sqllib/java12/db2java.zip
Implementation Class Name	COM.ibm.db2.jdbc.DB2XADataSource

- d. Click **Apply** and **Save**.

The provider BPEJdbcDriverDB2 is now listed in the JDBC Provider Panel.

- e. Select your new provider, BPEJdbcDriverDB2.
- f. In **Additional Properties**, select **Data Sources**.
- g. Click **New**.
- h. Enter your values for the following fields:

Name	BPEDataSourceDb2
JNDI Name	jdbc/BPEDB
Description	DataSource for Process Choreographer
Category	Process Choreographer
Statement Cache Size	0
Datasource Helper Class Name	com.ibm.websphere.rsadapter.DB2DataStoreHelper

- i. Click **Apply** and **Save**.

The data source BPEDataSourceDb2 has been created and is now listed in the Data Source panel.

Results

The DB2 JDBC provider and data source have been configured for the business process container.

What to do next

Configure the queue resources.

Configuring an Oracle JDBC provider and data source for the business process container

Before you begin

To configure an Oracle JDBC provider and data source, you must be using an OCI driver.

Steps for this task

1. Change to the Process Choreographer directory:

On Windows, enter:

```
cd %WAS_HOME%\ProcessChoreographer
```

On UNIX, enter:

```
cd $WAS_HOME/ProcessChoreographer
```

2. Run the configuration script.

On Windows, enter:

```
%WAS_HOME%\bin\wsadmin -f create_ds_Oracle.jacl  
[-node <nodeName>] -server <serverName>  
-dbPath <databasePath> [-tnsName <databaseName>]  
-userid <userid> -passwd <password>
```

On UNIX, enter:

```
$WAS_HOME/bin/wsadmin.sh -f create_ds_Oracle.jacl  
[-node <nodeName>] -server <serverName>  
-dbPath <databasePath> [-tnsName <databaseName>]  
-userid <userid> -passwd <password>
```

Where:

wsadmin

is located in the bin subdirectory of the IBM WebSphere Application Server installation directory.

create_ds_Oracle.jacl

is the script file that performs the configuration within WebSphere. It is located in the ProcessChoreographer subdirectory of the IBM WebSphere Application Server installation directory. For example, on a default Windows installation, it is in the directory `c:\WebSphere\AppServer\ProcessChoreographer`.

nodeName

is the optional node name.

serverName

is the application server name.

databasePath

is the location where your Oracle system is installed. Note: On Windows you must use forward slashes (/) instead of back slashes (\) to specify the path, for example, "c:/oracle/ora90".

databaseName

is the optional TNS name. It should match the name used for the Process Choreographer database in the file tnsnames.ora.

userid is the user ID for accessing the database. This must be the same user ID that was used to create the database.

password

is the password for *userid*.

For example, on Windows:

```
%WAS_HOME%\bin\wsadmin -f create_ds_Oracle.jacl
-server server1 -dbPath "c:/oracle/ora90" -tnsName BPEDB
-userid system -passwd OraclePassword
```

Results

The Oracle JDBC provider and data source have been configured for the business process container.

What to do next

Configure the queue resources.

Configuring a Cloudscape JDBC provider and data source for the business process container

Steps for this task

1. Change to the Process Choreographer directory:

On Windows, enter:

```
cd %WAS_HOME%\ProcessChoreographer
```

On UNIX, enter:

```
cd $WAS_HOME/ProcessChoreographer
```

2. Run the configuration script.

On Windows, enter:

```
%WAS_HOME%\bin\wsadmin -f create_ds_Cloudscape.jacl
[-node <nodeName>] -server <serverName> -dbPath <databasePath>
```

On UNIX, enter:

```
$WAS_HOME/bin/wsadmin.sh -f create_ds_Cloudscape.jacl
[-node <nodeName>] -server <serverName> -dbPath <databasePath>
```

Where:

wsadmin

is located in the bin subdirectory of the IBM WebSphere Application Server installation directory.

create_ds_Cloudscape.jacl

is the script file that performs the configuration within WebSphere. It is located in the ProcessChoreographer subdirectory of the IBM WebSphere Application Server installation directory.

nodeName

is the optional node name.

serverName

is the application server name.

databasePath

is the path to the Process Choreographer database directory. Note, on Windows use forward slashes (/) instead of back slashes (\). For example, "c:/mydbs/BPEDB".

For example, on Windows:

```
%WAS_HOME%\bin\wsadmin -f create_ds_Cloudscape.jacl
-server server1 -dbPath "c:/mydbs/BPEDB"
```

Results

The Cloudscape JDBC provider and data source have been configured for the business process container.

What to do next

Configure the queue resources.

Configuring a Sybase JDBC provider and data source for the business process container

Steps for this task

1. Change to the Process Choreographer directory:

On Windows, enter:

```
cd %WAS_HOME%\ProcessChoreographer
```

On UNIX, enter:

```
cd $WAS_HOME/ProcessChoreographer
```

2. Run the configuration script.

On Windows, enter:

```
%WAS_HOME%\bin\wsadmin -f create_ds_Sybase.jacl
[-node <nodeName>] -server <serverName> -dbPath <databasePath>
[-dbServer <dbServerName>] [-dbPort <port>] [-dbName <databaseName>]
-userid <userid> -passwd <password>
```

On UNIX, enter:

```
$WAS_HOME/bin/wsadmin.sh -f create_ds_Sybase.jacl
[-node <nodeName>] -server <serverName> -dbPath <databasePath>
[-dbServer <dbServerName>] [-dbPort <port>] [-dbName <databaseName>]
-userid <userid> -passwd <password>
```

Where:

wsadmin

is located in the bin subdirectory of the IBM WebSphere Application Server installation directory.

create_ds_Sybase.jacl

is the script file that performs the configuration within WebSphere. It is located in the ProcessChoreographer subdirectory of the IBM WebSphere Application Server installation directory. For example, on a default Windows installation, it is in the directory `c:\WebSphere\AppServer\ProcessChoreographer`, and on UNIX, it is in the directory `$WAS_HOME/ProcessChoreographer`.

nodeName

is the optional node name.

serverName

is the application server name.

databasePath

is the location where your Sybase system is installed. Note: On Windows you must use forward slashes (/) instead of back slashes (\) to specify the path, for example, "`c:/Sybase`".

dbServerName

is the name of the database server machine.

dbPort

is the port used on the database server.

databaseName

is the optional name of the Process Choreographer database.

userid is the user ID for accessing the database. This must be the same user ID that was used to create the database.

password

is the password for *userid*.

For example, on Windows:

```
%WAS_HOME%\bin\wsadmin -f create_ds_Sybase.jacl
-server server1 -dbPath "c:/sybase" -dbServer mymachine
-dbPort 4100 -dbName BPEDB -userid sa -passwd SybaseAdminPassword
```

Results

The Sybase JDBC provider and data source have been configured for the business process container.

What to do next

Configure the queue resources.

Configuring a Microsoft SQL Server JDBC provider and data source for the business process container

Steps for this task

1. Change to the Process Choreographer directory by entering the command:
`cd %WAS_HOME%\ProcessChoreographer`

2. Run the configuration script by entering the command.

```
%WAS_HOME%\bin\wsadmin -f create_ds_MsSql.jacl  
[-node <nodeName>] -server <serverName>  
[-dbServer <dbServerName>] [-dbPort <port>] [-dbName <databaseName>]  
-userid <userid> -passwd <password>
```

Where:

wsadmin

is located in the bin subdirectory of the IBM WebSphere Application Server installation directory.

create_ds_MsSql.jacl

is the script file that performs the configuration within WebSphere. It is located in the ProcessChoreographer subdirectory of the IBM WebSphere Application Server installation directory. For example, on a default Windows installation, it is in the directory `c:\WebSphere\AppServer\ProcessChoreographer`.

nodeName

is the optional node name.

serverName

is the application server name.

dbServerName

is the name of the database server machine.

dbPort

is the port used on the database server. The default value is 1433.

databaseName

is the optional name of the Process Choreographer database. The default value is BPEDB.

userid is the user ID for accessing the database. This must be the same user ID that was used to create the database.

password

is the password for *userid*.

For example:

```
%WAS_HOME%\bin\wsadmin -f create_ds_MsSql.jacl  
-server server1 -dbServer mymachine  
-dbPort 1433 -dbName BPEDB -userid sa -passwd Password
```

Results

The Microsoft SQL Server JDBC provider and data source have been configured for the business process container.

What to do next

Configure the queue resources.

Configuring the queue resources for the business process container

The business process container uses a JMS provider for sending and receiving messages. If you have installed the IBM WebSphere Application Server "embedded

messaging” option, it is recommended to use it. Otherwise, you must have already installed your external messaging system and performed Creating the queue manager and queues.

Steps for this task

1. Configure the queues, JMS provider, and message listener service, by performing one of the following:
 - **Embedded messaging:** Configuring queue resources using the JMS provider embedded in WebSphere.
 - **External messaging:** Configuring queue resources using WebSphere MQ or MQSeries.

Results

The queue resources needed by the business process container have been created.

What to do next

Create and configure the scheduler

Configuring queue resources for the business process container using the JMS provider embedded in WebSphere

Each business process container needs message queues, a JMS provider, and a message listener service. This topic explains how to create these resources using WebSphere’s embedded messaging.

Steps for this task

1. Make sure that you have the embedded messaging option installed on your WebSphere Application Server.
2. Change to the Process Choreographer directory:

On Windows, enter:

```
cd %WAS_HOME%\ProcessChoreographer
```

On UNIX, enter:

```
cd $WAS_HOME/ProcessChoreographer
```

3. Configure the queue resources, by running `config_queue_emb.jacl` with the appropriate parameters.

On Windows, enter:

```
%WAS_HOME%\bin\wsadmin -f config_queue_emb.jacl  
-userid <userid> -passwd <password>  
[-node <nodeName>] -server <serverName>  
[-internalQueue <internalQueueName>]  
[-apiQueue <apiQueueName>]  
[-holdQueue <holdQueueName>]  
[-retentionQueue <retentionQueueName>]
```

On UNIX, enter:

```
$WAS_HOME/bin/wsadmin.sh -f config_queue_emb.jacl  
-userid <userid> -passwd <password>  
[-node <nodeName>] -server <serverName>  
[-internalQueue <internalQueueName>]
```

```
[-apiQueue <apiQueueName>]
[-holdQueue <holdQueueName>]
[-retentionQueue <retentionQueueName>]
```

The default values for the optional parameters are:

node	<local node>
internalQueueName	BPEIntQueue
apiQueueName	BPEApiQueue
holdQueueName	BPEHldQueue
retentionQueueName	BPERetQueue

4. **(Optional)** You can check or change the configuration settings using the Administrative Console.

Results

The queue resources needed by the business process container have been created.

What to do next

Create and configure the scheduler.

Configuring the queue resources for the business process container using WebSphere MQ or MQSeries

Each business process container needs a JMS provider, and message listener ports defined for it. This topic explains how to create these resources.

Steps for this task

1. Make sure that you know the password for a user ID that has the authority to create MQ queues.

2. Change to the Process Choreographer directory:

On Windows, enter:

```
cd %WAS_HOME%\ProcessChoreographer
```

On UNIX, enter:

```
cd $WAS_HOME/ProcessChoreographer
```

3. Configure the queue resources by running script `config_queue_mq.jacl` with the appropriate parameters:

On Windows, enter:

```
%WAS_HOME%\bin\wsadmin -f config_queue_mq.jacl
-userid <userid> -passwd <password>
[-node <nodeName>] -server <serverName>
-mqPath <MQPath> [-qmName <queue manager name>]
[-qmNamePut <cluster queue manager name>]
[-internalQueue <internalQueueName>] [-apiQueue <apiQueueName>]
[-holdQueue <holdQueueName>] [-retentionQueue <retentionQueueName>]
```

For example, for a Windows configuration based on the defaults:

```
%WAS_HOME%\bin\wsadmin -f config_queue_mq.jacl
-userid Administrator -passwd adminPassword
-server server1 -mqPath C:\Progra~1\MQSeries
```

On UNIX, enter:

```
$WAS_HOME/bin/wsadmin.sh -f config_queue_mq.jacl
  -userid <userid> -passwd <password>
  [-node <nodeName>] -server <serverName>
  -mqPath <MQPath> [-qmName <queue manager name>]
  [-qmNamePut <cluster queue manager name>]
  [-internalQueue <internalQueueName>] [-apiQueue <apiQueueName>]
  [-holdQueue <holdQueueName>] [-retentionQueue <retentionQueueName>]
```

For example, for an AIX configuration based on the defaults:

```
$WAS_HOME/bin/wsadmin -f config_queue_mq.jacl
  -userid mqm -passwd mqmPassword -server server1 -mqPath /usr/mqm
```

The default values for the optional parameters are:

node <local node>

qmName

WAS_<nodeName>_<serverName>

qmNamePut

WAS_<nodeName>_<serverName>. This parameter is only required if you are configuring a cluster setup that uses a WebSphere MQ cluster with separate "put" and "get" queue managers.

internalQueueName

BPEIntQueue

apiQueueName

BPEApiQueue

holdQueueName

BPEHldQueue

retentionQueueName

BPERetQueue

4. If you have problems using the script `config_queue_mq.jacl`, you can also configure the resources manually using the Administrative Console.
5. **(Optional)** You can check or change the configuration settings using the Administrative Console.

Results

The queue resources needed by the business process container have been created.

What to do next

Create and configure the scheduler.

Configuring MQ resources for the business process container using the Administrative Console

Before you begin

You must have already created the queues for the business process container.

It is recommended that you configure the resources using the script provided. If you must create the resources manually, this topic describes how to do it using the Administrative Console:

Steps for this task

1. Select **Resources > JMS > WebSphere MQ JMS Providers**.
2. Click **WebSphere MQ Queue Connection Factory**.
3. Click **New**.
4. Enter the following values:

Field	Example value
Name	BPECF
JNDI Name	jms/BPECF
User ID	<i>userID</i>
Password	<i>password</i>
Queue Manager	WAS_QueueManager

Leave the other fields blank or accept the default values.

5. Click **Apply**, and click **Save**.
The MQ Queue Connection factory BPECF has been created and is listed on the MQ queue connection factory panel.
6. Define the MQ Queue Destination for the external message queue:
 - a. Select **Resources > JMS > WebSphere MQ JMS Providers**.
 - b. Click **WebSphere MQ Queue Destination**.
 - c. Click **New**.
 - d. Enter the following values:

Field	Example value
Name	BPEApiQueue
JNDI Name	jms/BPEApiQueue
Specified Priority	3
Specified Expiry	3
Base Queue Name	BPEApiQueue

Leave the other fields blank or accept the default values.

- e. Click **Apply** then **Save**.
7. Define the MQ Queue Destination for the internal messages queue:
 - a. Click **WebSphere MQ Queue Destination**.
 - b. Click **New**.
 - c. Enter the following values:

Field	Example value
Name	BPEIntQueue
JNDI Name	jms/BPEIntQueue
Specified Priority	3
Specified Expiry	3
Base Queue Name	BPEIntQueue

Leave the other fields blank or accept the default values.

- d. Click **Apply**, then **Save**.
8. Define the MQ Queue Destination for the hold queue:
 - a. Click **WebSphere MQ Queue Destination**.

- b. Click **New**.
- c. Enter the following values:

Field	Example value
Name	BPEHldQueue
JNDI Name	jms/BPEHldQueue
Specified Priority	3
Specified Expiry	3
Base Queue Name	BPEHldQueue

Leave the other fields blank or accept the default values.

- d. Click **Apply**, then **Save**.
9. Define the MQ Queue Destination for the retention queue:
- a. Click **WebSphere MQ Queue Destination**.
 - b. Click **New**.
 - c. Enter the following values:

Field	Example value
Name	BPERetQueue
JNDI Name	jms/BPERetQueue
Specified Priority	3
Specified Expiry	3
Base Queue Name	BPERetQueue

Leave the other fields blank or accept the default values.

- d. Click **Apply**, then **Save**.

Now the four queue destinations have been defined and are listed on the queue panel.

10. Create the listener port for external messages:
- a. Click **Servers > Applications > YourServer > Message Listener Service > Listener Ports**.
 - b. Click **New**.
 - c. Enter the following values:

Field	Example value
Name	BPEApiListenerPort
Description	Process Choreographer API
Connection Factory JNDI Name	jms/BPECF
Destination JNDI Name	jms/BPEApiQueue
Max Sessions	5
Max Retries	10
Max Messages	1

- d. Click **Apply**, then **Save**.

11. Create the listener port for the hold queue:
- a. In **Servers > Applications > YourServer > Message Listener Service > Listener Ports**, click **New**.
 - b. Enter the following values:

Field	Example value
Name	BPEHoldListenerPort
Description	Process Choreographer hold
Connection Factory JNDI Name	jms/BPECF

Destination JNDI Name	jms/BPEHoldListenerPort
Max Sessions	5
Max Retries	10
Max Messages	1

c. Click **Apply**, then **Save**.

12. Create the listener port for internal messages:

a. In **Servers > Applications > YourServer > Message Listener Service > Listener Ports**, click **New**.

b. Enter the following values:

Field	Example value
Name	BPEInternalListenerPort
Description	Process Choreographer internal
Connection Factory JNDI Name	jms/BPECF
Destination JNDI Name	jms/BPEInternalListenerPort
Max Sessions	5
Max Retries	10
Max Messages	1

c. Click **Apply**, then **Save**.

The three listener ports have been created and are listed on the listener port panel.

Results

The queue resources needed by the business process container have been created.

What to do next

Create and configure the scheduler.

Creating and configuring the scheduler service for Process Choreographer

The business process container uses the scheduler service to provide time-dependent services. To create and configure the scheduler, perform the following:

Steps for this task

1. Change to the Process Choreographer directory:

On Windows, enter:

```
cd %WAS_HOME%\ProcessChoreographer
```

On UNIX, enter:

```
cd $WAS_HOME/ProcessChoreographer
```

2. Run the createScheduler.jacl script.

On Windows, enter:

```
%WAS_HOME%\bin\wsadmin -f createScheduler.jacl  
-server <Server> -node <Node>
```

On UNIX, enter:

```
$WAS_HOME/bin/wsadmin.sh -f createScheduler.jacl  
-server <Server> -node <Node>
```

Where:

Server is the name of the application server.

Node is the name of the node. This is optional. If the node is omitted, the local node is used.

If you have problems using the createScheduler.jacl, you can use the Administrative console to configure the resources manually.

For more information about the scheduler service, see (Using the scheduler service).

3. **(Optional)** You can check or change the configuration settings using the Administrative Console.

Results

The scheduler has been created and configured for Process Choreographer.

What to do next

Install the business process container.

Installing the business process container

Create a new business process container and enter its settings using the administrative console:

Steps for this task

1. Click **Servers > Application Servers > YourServer**.
2. In the **Additional Properties** section, click **Business Process Container**.
3. On the configuration page, if necessary, replace the default values with your values.
4. For **JMS API User ID** enter the user ID to be used by the business process container when processing asynchronous API calls.
5. For **JMS API password** enter the password for the JMS API user ID.
6. Click **Install**.

This causes the business process container's EAR file to be installed and configured. It is OK if you get an error message telling you that the container is already configured. However, if you want to change any of the container's configuration settings, you must uninstall the ProcessContainer.ear application and then repeat this task from step 1.

7. Click **Save**.

Results

The business process container has been installed and configured.

What to do next

You must activate the business process container.

Business Process Container settings

Use this page to manage process containers.

A Process Choreographer process container provides services to execute business processes within an application server. To view this administrative console page, click **Servers > Application Servers > *server_name* > Business Process Container**.

Note: These fields cannot be changed after they have been applied. If you want to change any of these values after the container has been configured, you must uninstall and reinstall the enterprise application file `BPEContainer.ear`. This can result in the loss of data such as pending messages in queues, process templates, and process instances in the Process Choreographer database.

Datasource

The JNDI name of the data source, which will store the process data.

Data type	String
Default	jdbc/BPEDB

Security Role For Process Administrator

The user or group from the domain's user registry that is to be mapped onto the role of Business Process Administrator.

Data type	String
Default	ProcessAdministrator
Restrictions	The user registry can be the local operating system, LDAP, or custom registry. The user or group specified must already exist in the user registry being used.

Listener Port For Internal Messages

This listener port provides the settings for the message-driven bean that handles messages exchanged within Process Choreographer processes.

Data type	String
Default	BPEInternalListenerPort

Listener Port For External Requests

This listener port provides the settings for the message-driven bean that handles requests from Process Choreographer API clients.

Data type	String
Default	BPEApiListenerPort

Listener Port For Unprocessed Messages

This listener port provides the settings for the queue that contains the messages that could not be processed.

Data type	String
Default	BPEHoldListenerPort

Retry Limit

Specifies the maximum number of retries for processing a message. When the limit is reached, the message is sent to the "Listener Port for unprocessed messages".

Data type	Integer
Default	5
Range	2 to 10 (recommended)

Retention Queue

Queue that contains those messages that temporarily could not be processed.

Data type	String
Default	jms/BPERetQueue

Retention Queue Factory

Factory for the retention queue.

Data type	String
Default	jms/BPECF

Retention Queue Message Limit

The maximum number of messages that can be stored in the retention queue. When the limit is reached, the messages are sent to the "Queue for internal messages" again and the process container switches into the quiesce mode.

Data type	Integer
Default	20

JMS API User ID

The user ID that the Process Choreographer MDB will use when processing asynchronous API calls. This is only required when WebSphere security is enabled (even if you will not use the JMS API).

Mandatory	If WebSphere security is enabled (even if you will not use the JMS API).
Data type	String
Description	<p>If you will not use the JMS API, you must specify a valid user ID - it does not need any special authorizations.</p> <p>If you will use the JMS API, this user ID must either be given the appropriate authorities when the process is modelled, or more commonly, it must be a member of a group that has been granted the necessary authorities. The possible staff authorities associated with processes are: Administrator, Reader, and Starter. A user ID can only perform <code>sendEvent</code> if it is a potential owner of the associated <code>receiveEvent</code> activity.</p> <p>If you want to allow all actions on processes via the JMS API, you can specify a user ID that is a member of the J2EE role <code>BPESystemAdministrator</code>, however, in a production system, the more fine-grained security approach is recommended.</p>

JMS API Password

The password for the JMS API user ID.

Data type	String
-----------	--------

Scheduler Calendar

The JNDI name of the scheduler `UserCalendar` to be used by the process container.

Data type	String
Default	BPEScheduler

Activating the business process container

To activate the business process container, you must restart your application servers.

Steps for this task

1. If you installed the business process container on a cluster of application servers, restart the cluster using RippleStart.
2. If you installed the business process container on one application server, restart the application server:
 - a. Stopping the server.
 - b. Starting the server.

Results

The business process container is ready to run business processes.

What to do next

Verify that the business process container works.

Verifying that the business process container works

Before you begin

The business process container must be configured and the database system and messaging service must have been started.

To verify that the business process container is working, you can either use the sample provided, or run your own application.

Steps for this task

1. If you want to run the **Process Choreographer** sample, open the Samples Gallery.
2. **(Optional)** If you want to use your own enterprise application that contains processes:

Install your application using the administrative console. If it contains processes, the process templates will be written into the Process Choreographer database. The process templates are automatically enabled, and process instances will be created as soon as requests arrive from a client.
3. In case of problems, see “Troubleshooting the business process container” on page 62.

Results

The business process container is working.

Troubleshooting the business process container

For information about process-specific messages, tracing, and audit trails, see Chapter 10, “Troubleshooting Process Choreographer”, on page 155. Here are some things to check if you have problems getting the business process container to work:

Steps for this task

1. 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 (or to run the scripts to create it).
2. I get a database error when I install an enterprise application that contains a process.
When an enterprise application is installed, any process templates are written into the Process Choreographer database. Make sure that the database system used by the business process container is running and accessible.
3. Can't invoke Cloudscape tools.
Make sure that you have set up the Java environment, and have included the necessary JAR files in the `classpath` environment variable.
4. Can't configure a JDBC provider or data source for Oracle
The jacl scripts provided with Process Choreographer assume that you are using an OCI driver. For performance reasons, a thin driver should not be used.
5. Problems using national characters.
Make sure that your database was created with support for Unicode character sets.
6. Problem creating the queues using MQSeries: MQSeries dll not found.
Add the MQSeries Java lib directory to your path environment variable. For example, on Windows, enter the command:

```
set path=MQInstallationDirectory\java\lib;%path%
```


where *MQInstallationDirectory* is the installation directory for WebSphere MQ.
7. Problem on AIX connecting to the queue manager.
Edit the file `/var/mqm/mqs.ini`, and add the following property to the definition for your queue manager:

```
IPCCBaseAddress=12
```
8. Preventing deadlocks in DB2.
To avoid deadlocks, be sure that the DB2 isolation level is set to 'read stability'. If necessary, enter the command `db2set DB2_RR_TO_RS=YES` and restart the DB2 instance to activate the change.
9. If Sybase reports the exception: The 'ALTER TABLE' command is not allowed within a multi-statement transaction in the '<i>databaseName</i>' database.
This can happen after you have installed a new PTF level that requires the database to be migrated. Start the Sybase `isql` command program, enter the following commands:

```
use master
go
sp_dboption databaseName, "ddl in tran", true
go
```

where *databaseName* is the name of your database.

Chapter 4. Uninstalling the business process container

Before you begin

Before you can uninstall the business process container, you must stop all process templates, delete all process instances, then stop and uninstall all enterprise applications that contain business processes.

Steps for this task

1. Ensure that all the stand-alone servers, the database, and the application server (or at least one application server per cluster) are running.
2. For each enterprise application that contains business processes:
 - a. Select **Applications > Business Process Modules**.
 - b. For each process archive (.far file), click **Templates > Select All > Stop > Save > Save**.
 - c. Wait for any running instances to finish.

Note: To see if any instances are running, you can log on to the Web client as administrator, and view the processes that are "Administered By Me". If the list is empty, no instances exist.
 - d. If there are any instances that have the autoDelete flag set to false, or that cannot be completed for some other reason, you must delete them manually.
 - e. Stop and uninstall the application.
 - f. If there are any more applications that contain business processes, repeat step 1.
3. Uninstall the business process container:
 - a. Select **Applications > Enterprise Applications > BPEContainer_Identifier**.
Where *Identifier* depends on where you installed the business process container.

Installed the business process container on	BPEContainer_Identifier
Application Server	BPEContainer_<i>serverName</i>
Node	BPEContainer_<i>nodeName</i>_<i>serverName</i>
Cluster	BPEContainer_<i>clusterName</i>

- b. Click **Stop**.
- c. Click **Uninstall > Save > Save**.
- d. If the uninstall fails, make sure that all business process applications have been stopped and uninstalled.

Results

The business process container has been uninstalled. All related resources (such as scheduler, data sources, listener ports, connection factories, queue destinations, queues, and database) remain, and can either be uninstalled separately, or retained for reuse.

Chapter 5. Managing processes

Before you begin

For each application server where you want to use Process Choreographer, the business process container must be installed and configured.

Managing processes involves the following:

Steps for this task

1. Installing process applications.
2. Uninstalling process applications.
3. Stopping and starting process templates.
4. Querying and replaying failed messages.

Installing process applications

To install an enterprise application that contains process modules:

Steps for this task

1. Make sure that your application server's business process container is configured.
2. Make sure that the application server is running.
Note: In this respect, Process Applications are different to normal J2EE Applications.
3. Install your application on your application server.

Results

All process modules in the enterprise application (all FAR files contained in the EAR file) are started automatically. This means that instances of the process templates will be created as soon as any requests arrive.

Uninstalling process applications

To uninstall an enterprise application that contains process modules:

Steps for this task

1. Ensure that all the stand-alone servers, the database, and the application server (or at least one application server per cluster) are running.
2. Stop all process templates in the application. This prevents new process instances from being created.
3. Make sure that no process instances are running. If necessary, a process administrator can use the Process Choreographer's Web client to stop running process instances.
4. To stop the application and uninstall it:
 - a. Select **Applications > Enterprise Applications** from the navigation pane on the left.
 - b. Select the application that you want to stop.

- c. Click **Stop**.
- d. Click **Uninstall**.

Results

The process application is uninstalled.

Stopping and starting process templates

Before you begin

Be sure that you are using a Console User ID that has either the role operator or administrator.

When an enterprise application that contains process modules is installed and deployed, the process templates that are contained in the process modules are started automatically. You can stop these process templates if required, and restart them again. To do this:

Steps for this task

1. In the navigation pane of the administrative console, select **Applications > Enterprise Applications**.
2. Click the application you want to manage.
3. In the Related Items section of the configuration properties page, click **Business Process Modules**.
4. Select the process module you want to manage.
5. In the Additional Properties section, click **Templates**.

The Process Templates page is displayed. It lists the process templates that are contained in the selected process module.

You can stop a running process template, and start it again at a later point in time, depending on the Valid from Time that is specified. The Valid from Time is specified in Coordinated Universal Time (UTC). It specifies when a process template is started. A process template will not be started until the Valid From Time is reached.

6. To stop a process template, check the check box next to the process template and click **Stop**.
7. To start a process template, check the check box next to the process template and click **Start**.

Note: As long as the started process exists, do not delete the user ID that you used to start the process from the user registry. The navigation of a process relies on the J2EE security context of the process starter, and it cannot continue if the user ID does not exist in the user registry. If you delete such a user ID by accident, you must recreate it to be able to continue navigating this process.

Querying and replaying failed messages

When there is a problem processing an internal message, it will end up on the Retention queue or Hold queue. To find out if there are any failed messages, and then replay them by sending them to the internal queue again:

Steps for this task

1. Be sure that you are using a user ID that has administrative rights.

2. Change to the Process Choreographer sample directory where the jacl scripts are located:

On Windows, enter:

```
cd %WAS_HOME%\ProcessChoreographer\sample
```

On UNIX, enter:

```
cd $WAS_HOME/ProcessChoreographer/sample
```

3. **Query** the number of failed messages on both the retention and hold queues by entering one of the following commands:

```
wsadmin -f queryNumberOfFailedMessages.jacl -cluster clusterName  
wsadmin -f queryNumberOfFailedMessages.jacl -node nodeName -server serverName
```

clusterName

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

nodeName

optional when specifying the server name, identifies the node. The default is the local node.

serverName

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

If you want to check for a server on the local node, you can just enter:

```
wsadmin -f queryNumberOfFailedMessages.jacl -server serverName
```

4. **Replay** all failed messages on the hold queue, retention queue, or both queues by entering one of the following commands:

```
wsadmin -f replayFailedMessages.jacl -cluster clusterName -queue replayQueue  
wsadmin -f replayFailedMessages.jacl -node nodeName -server serverName -queue replayQueue  
wsadmin -f replayFailedMessages.jacl -server serverName -queue replayQueue
```

where

replayQueue

must have one of the values: holdQueue, retentionQueue, or both.

clusterName

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

nodeName

optional when specifying the server name, identifies the node. The default is the local node.

serverName

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

Process modules collection

Use this page to view a list of the process modules defined for this application.

A process module contains process templates. When a process module is started, all its templates are enabled, and instances of the process templates can be created.

To view this administrative console page, click **Servers > Applications > *application_name* > Process Modules** .

Process module settings

Use this page to manage process modules.

A business process module contains one or more business process templates.

To view this administrative console page, click **Applications > Applications > *application_name* > Process Modules > *processmodule_name***.

Process templates collection

Use this page to manage business process templates associated with a process module.

To view this administrative console page, click **Applications > Applications > *application_name* > Process Modules > *processmodule_name* > Templates**.

Name

Name of the process template.

Data type String

Valid From

Specifies the point in time from which the process can be started.

You can stop a running process template, and start it again at a later point in time, depending on the Valid From time that is specified.

Data type String

Units Coordinated Universal Time (UTC)

Current State

Specifies the current state of the process template.

Range Started, Stopped

Started The selected process template is started, which means that new instances of it will be created when request messages arrive.

Stopped The selected process template is finished and then stopped. No new instances will be created.

Process template settings

Use this page to modify business process template settings.

A process template is the definition of one particular business process.

To view this administrative console page, click **Applications > Applications > *application_name* > Process Modules > *processmodule_name* > *processtemplate_name***.

Name

Name of the process template.

Data type String

Valid From

Specifies the point in time from which the process can be started.

You can stop a running process template, and start it again at a later point in time, depending on the Valid From time that is specified.

Data type String

Units Coordinated Universal Time (UTC)

Current State

Specifies the current state of the process template.

Range Started, Stopped

- | | |
|----------------|--|
| Started | The selected process template is started, which means that new instances of it will be created when request messages arrive. |
| Stopped | The selected process template is finished and then stopped. No new instances will be created. |

Chapter 6. Using the Process Choreographer Web client

This set of topics describes how you can use the Process Choreographer Web client to work with deployed business processes.

Before you begin

Before you can use the Process Choreographer Web client to work with processes and their associated activities, you must have started the Web client

You can use the Process Choreographer Web client to display information about process templates, process instances, and work items. You can also act on processes and work items; for example, to start new process instances, or to claim and complete a work item in your To Do list.

The process templates, process instances, and work items that you can see depend on the authority that has been assigned to your user ID or the role that you are working as.

You can use the Web client to work with business processes that you have deployed in WebSphere Application Server Enterprise. You can:

- Work with work items
 - Display work items in your To Do list
 - Display information about a work item
 - Claim a work item
 - Complete a work item
 - **5.0.2** Query work items
- Work with process instances
 - Work with process instances you administer
 - Work with process instances you have created
 - Display information about a process instance
 - **5.0.2** Monitor a process instance
 - **5.0.2** Administer compensation for a process instance
 - **5.0.2** Terminate a process instance
 - **5.0.2** Query process instances
- Work with process templates
 - View a process template
 - Start a process instance
 - **5.0.2** Query process templates

If you encounter problems when using the Web client, see “Troubleshooting the Process Choreographer Web client” on page 93 for information on how to solve these problems.

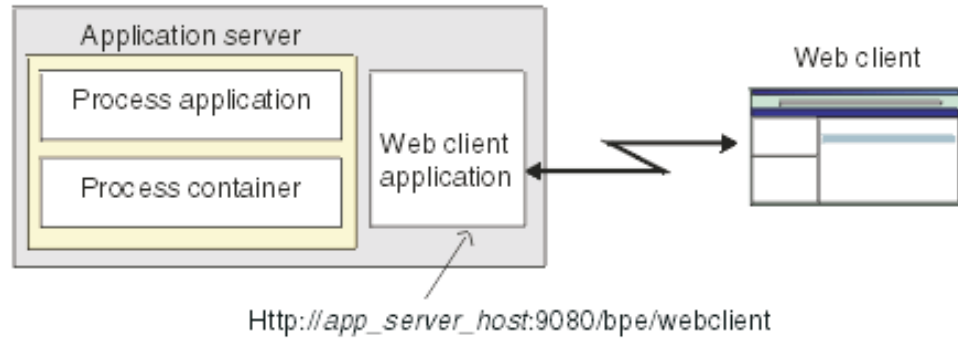
For information on how to adapt the Web client interface, see “Customizing the Process Choreographer Web client” on page 85.

About the Process Choreographer Web client

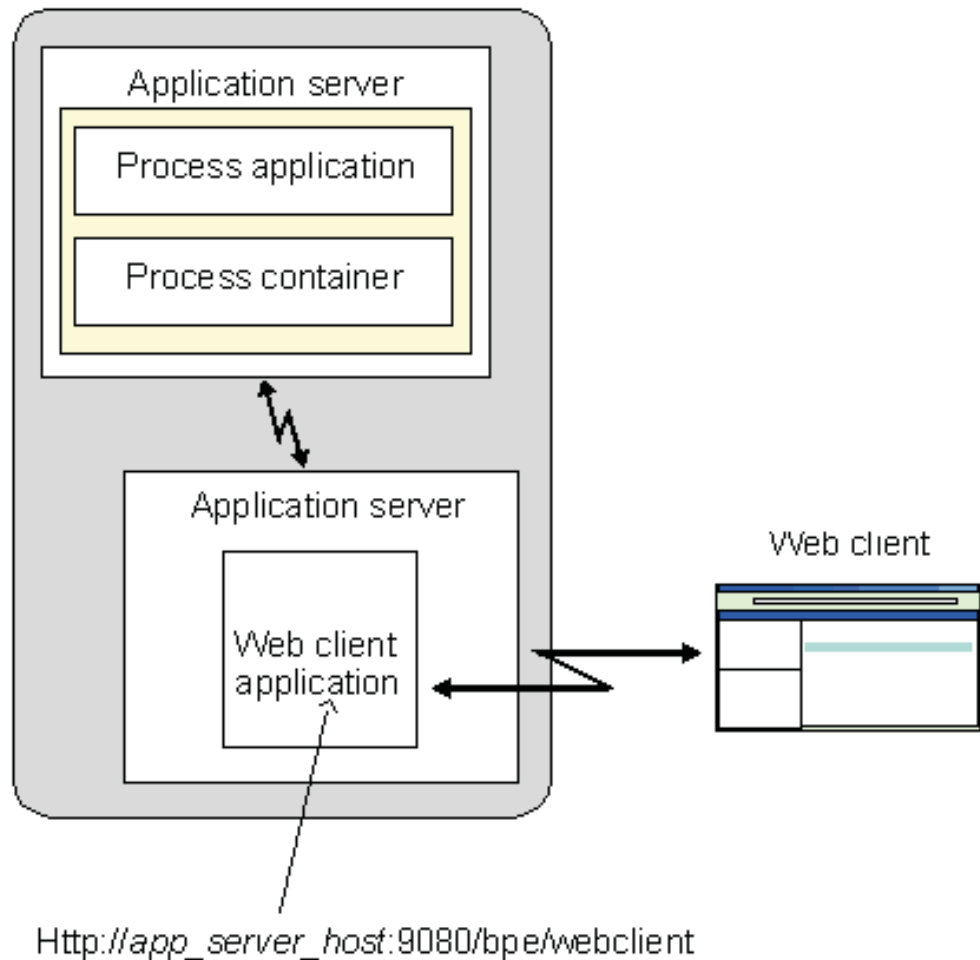
You can use the Process Choreographer Web client to work with business processes that you have deployed as applications.

The Web client can run as:

- A Web application on the host where your business process application is deployed.



- **5.0.2** A Web application on a host other than the one where your business process application is deployed.



Note: If you want to work with business process applications on several application servers at the same time, you need to start a Process Choreographer Web client for each application server.

You can use the Web client to display information about work items, process instances, and process templates. You can also act on process instances and work items; for example, you can start new process instances or claim and complete a work item in your To Do list.

The main concepts for using the Web client are as follows:

My To Do list

This lists the work items that you own or have been assigned to you as a potential owner. This list is displayed in the My To Dos page of the Web client.

When you claim a work item, you become the owner of that work item.

Process instances you have started

If you start a new process instance, you are registered as the creator of that process. You can use the Created By Me page of the Web client to display the processes that you have started.

Process instances you administer

You can be authorized as the administrator for a process template. You can use the Administered By Me page of the Web client to display the process instances that you can administer.

As an administrator, you can use the Web client to manage process instances and, if needed, to terminate process instances that have problems.

Process template

A process template defines the activities that form a business process, and is used to start a new business process. You can use the My Templates page to start a process instance or view information about a process instance.

For more information about the pages of the Web client, see “Process Choreographer Web client page directory” on page 93.

Starting the Process Choreographer Web client

Before you begin

With most types of installation, the Process Choreographer Web client is already installed on your system. Before you can start the Process Choreographer Web client, you must have completed the following prerequisite steps:

1. Configured the business process container. This step enables the Process Choreographer service and the Process Choreographer Web client. For information on how to configure the business process container, Chapter 3, “Configuring the business process container”, on page 13.
Depending on the type of installation you performed, your business process container might already be configured. To check whether you have a business process container configured on your server, click **Applications > Enterprise Applications** in the administrative console’s navigation bar. Your process container is available if an application named `BPEContainer_<hostName>_<serverName>` is in the list of enterprise applications. If a business process container is not available, you must configure one manually on your server before you can proceed. For more information on configuring a business process container manually, see “Configuring the business process container manually” on page 44.
2. Configured the WebSphere Application Server security environment for secured applications, including assigning users and groups to roles defined in business process applications and configuring authentication mechanisms. For more information about configuring the WebSphere Application Server security environment for secured applications, see Managing secure applications (not in this document).
3. Deployed the applications that use the business processes. For more information on deploying applications, see “Installing process applications” on page 67.

Note: If you want to work with business process applications on several application servers at the same time, you need to start a Process Choreographer Web client for each application server. If you are not working in an ND environment, a Web client is installed automatically on each server that has a business process container installed. In an ND environment, however, you have to install and configure the Web client manually.

To start the Process Choreographer Web client, complete the following steps.

Steps for this task

1. Use a Web browser to open the following URL:

`http://app_server_host:9080/bpe/webclient`

Where *app_server_host* is the network name for the host for the application server that provides the business process application that you want to work with.

The Web client displays an authentication window for you to provide a user ID and password.

2. Type a user ID and password then click **OK**.

Results

This task displays the initial page of the Process Choreography Web client, which shows the work items in your To Do list.

Working with work items

This set of topics describes how you can use the Process Choreographer Web client to work with work items of interruptible business processes.

For a non-interruptible process there is no human interaction, and the process output message is displayed immediately after the process finishes.

For an interruptible process, you can use the Process Choreographer Web client to work with activities that require human interaction. If you are authorized as a potential owner, editor, or reader of an activity, the work item is added automatically to your To Do list which is displayed in the My To Dos page of the Web client.

To work with a work item, you must claim the work item then perform the actions needed to complete it.

When you complete a work item, its output message is saved. The data is available to subsequent activities in the process.

You can display information about a work item that is in your To Do list, and can display information about the process that the work item is part of. You may also have been authorized to view information about other work items that you cannot otherwise work with.

You can use the Web client to work with work items. You can:

- Display work items in your To Do list
- Display information about a work item
- Claim a work item
- Complete a work item
- **5.0.2** Query work items

Displaying work items in your To Do list

If you are authorized to work with a new work item, it is added automatically to your To Do list. You can view your To Do list by displaying the My To Dos page.

The My To Dos page is displayed when you first start the Web client, and you can return to that page at any time from other Web client pages.

From the My To Dos page, you can select a work item to work on; for example, to claim a work item that you want to complete.

To display the My To Dos page, click **My To Dos** under Work Item Lists in the navigation pane of the Web client.

Note: The My To Dos page is not refreshed automatically. If you change a work item, or want to check the latest contents of your To Do list, you need to refresh the My To Dos page manually.

Results

This task displays the My To Dos page of the Process Choreographer Web client.

Displaying information about a work item

You can use the Process Choreographer Web client in the following ways to view information about a work item:

- On the My To Dos page, click the name of the work item.
This displays the Activity page, which provides the set of information and actions needed to work with the work item in its current state.
- On the Process Instance page, click the name of the activity listed under Activities.
This displays the Activity page, which provides the set of information and actions needed to work with the work item in its current state.
- On the Activity page, click **View more details about this activity**.
This displays the Activity Information page, which provides more detailed information about the activity that the work item is associated with. The Activity Information page also provides information about the process that the corresponding activity is part of.

Claiming a work item

To work with a work item, you must claim the work item then perform the actions needed to complete it. You can claim a work item that is in the ready state if you are a potential owner of that work item. If you claim a work item, you become the owner of that work item and are responsible for completing it.

You can claim a ready work item from its Activity page; for example, by completing the following steps:

Steps for this task

1. Click **My To Dos** under Work Item Lists in the navigation pane of the Web client.
This action displays the My To Dos page, which lists the work items that you can work with.
2. Click the work item name.
5.0.2 This action displays the Activity page, with **Claim** displayed as an available action. This page also displays information about the process the work item belongs to and the output message for the work item. You cannot change the fields displayed until you have claimed the work item.

- 5.0+ 5.0.1+** This action displays the Activity page, with **Claim Activity** displayed as an available action. This page also displays the required user input and the current output message for the work item. You cannot change the fields displayed until you have claimed the work item.
3. **(Optional)** For more detailed information about the activity associated with the work item, you can click **View more details about this activity**, but should return to the Activity page to claim the work item.
 4. Claim the activity.

5.0.2 Click **Claim**.

5.0+ 5.0.1+ Click **Claim Activity**.

Results

When you have claimed a work item, the Activity page is displayed with the output message fields enabled for you to work with.

What to do next

Use the Activity page to complete the work item, as described in [Completing a work item](#).

Completing a work item

Before you begin

Before you can complete a work item, you must have claimed the work item, as described in [Claiming a work item](#).

After you have claimed a work item, you can perform the actions needed to complete it.

You do not have to complete a work item in one step. While working with a work item, you can save changes you make to the work item and leave the work item to do other things. For example, you may want to save your changes if your work is interrupted or if you need to get more information before completing the work item.

Steps for this task

1. Display the Activity page, for example, by clicking the name of the activity on your My To Dos page.
2. Complete the input for the work item.

The content of the **Activity Output Message** section, including the number and types of input fields, depends on the design of the activity in the process template.
3. **(Optional)** If you want to save your changes, click **Save**.

For example, you may want to save your changes if your work is interrupted or if you need to get more information before completing the work item.
4. If you have finished all the required input, and want to complete the work item, click **Complete**.

5.0.2 Click **Complete**.

Note: If you click **Complete**, you will not be able to make any more changes to the work item.

5.0+ **5.0.1+** Click **Complete Activity**.

Note: If you click **Complete Activity**, you will not be able to make any more changes to the work item.

If required input is missing, you cannot complete the work item. The Activity page indicates the input you still need to supply to complete the work item.

Results

If you have specified the required input, the work item is completed, and the Web client window refreshed to display your My To Dos page. Information about the completed work item is saved. It is then available to subsequent activities in the process.

Querying work items

You can define queries to find work items that meet a certain set of characteristics. For example, you can define a query to find all the work items that have been claimed by a particular user or all the work items that you have claimed.

Steps for this task

1. Click **Define Work Item List** under Work Item Lists in the navigation pane.
This action displays the Define Work Item List page.
2. Specify criteria for querying work items and click **Show List**.
This action displays the User-Defined Work Items List page.
Note: This page displays only those work items that meet **all** of your search criteria.
3. **(Optional)** You can work with the work items found by the query. If you are authorized as the Process Choreographer system administrator, you can also save the query so that it appears under Work Item Lists.

Working with process instances

This set of topics describes how you can use the Process Choreographer Web client to work with business processes instances.

You can use the Web client to display information about process instances or act on process instances. The Web client displays only those process instances that you are authorized to work with, and displays only those actions that you are authorized to perform on the process instances.

If you are a potential starter, administrator, or reader of a process template, you can also display information about the process template.

You use the Web client to do the following:

- Work with process instances you administer
- Work with process instances you have created
- Display information about a process instance
- **5.0.2** Monitor a process instance
- **5.0.2** Administer compensation for a process instance

- **5.0.2** Terminate a process instance
- **5.0.2** Query process instances

Working with process instances you administer

Before you begin

To administer processes, you need to have administer authority for the business process.

The Administered By Me page of the Process Choreographer Web client displays a list of only those process instances that you can administer.

Steps for this task

1. Display the Administered By Me page.
 - **5.0.2** Click **Administered By Me** under Process Instance Lists in the navigation pane.
 - **5.0 + 5.0.1 +** Click **Administered By Me** under Workitem Lists in the navigation pane.
2. Click the name of the process instance you want to work with.

The Web client displays a Process Instance page with information and available actions for the process instance. The information displayed includes a list of the current activities in the process. Optionally, you can click an activity name to work with that activity. If the process instance has finished, this page includes the process output message.
3. **(Optional)** To act on the process instance, click an action displayed under Available Actions.

The actions that are available depend on the current state of the process instance. For example, if a process instance is in the running state, **Terminate** is displayed.

Working with process instances you have started

The Created By Me page of the Process Choreographer Web client displays a list of the process instances that you have started.

Steps for this task

1. Display the Created By Me page.
 - **5.0.2** Click **Created By Me** under Process Instance Lists in the navigation pane.
 - **5.0 + 5.0.1 +** Click **Created By Me** under Workitem Lists in the navigation pane.
2. Click the name of the process instance you want to work with.

The Web client displays a Process Instance page with information and available actions for the process instance. The information displayed includes a list of the current activities in the process. Optionally, you can click an activity name to work with the corresponding work item. If the process instance has finished, this page includes the process output message.
3. **(Optional)** To act on the process instance, click an action displayed under Available Actions.

The actions that are available depend on your authorization and the current state of the process instance. For example, if the process instance is in the running state, **Terminate** is displayed.

Displaying information about a process instance

You can use the Process Choreographer Web client in the following ways to view information about any process instance:

- To display information about a process instance that you started:
 1. **5.0.2** Click **Created By Me** under Process Instance Lists in the navigation pane.
 2. **5.0+** **5.0.1+** Click **Created By Me** under Workitem Lists in the navigation pane.
 3. In the Created By Me page, click the name of the process instance.
- To display information about a process instance that you administer:
 1. **5.0.2** Click **Administered By Me** under Process Instance Lists in the navigation pane.
 2. **5.0+** **5.0.1+** Click **Administered By Me** under Workitem Lists in the navigation pane.
 3. In the Administered By Me page, click the name of the process instance.
- On an Activity page, click **View more details about this process**

Results

The Web client displays a Process Instance page with information and available actions for the process instance. The information displayed includes a list of the current activities in the process. You can click an activity name to display information about that activity. If the process instance has finished, this page includes the process output message.

Monitoring a process instance

You can view the progress of the activities in a process instance.

Steps for this task

1. On any of the process instance list pages, for example, Administered By Me, select the process instance you want to monitor, then click **Monitor**.

The Process Instance Monitor page is displayed. This gives a brief description of the process and shows the activities and their status.

2. **(Optional)** Click an activity to see more information about it.

Administering compensation for a process instance

If compensation has failed for a process, there are a number of administrative actions you can take.

Steps for this task

1. Under Process Instance Lists, click **Compensation In Doubt**.

The Compensation In Doubt page is displayed. This page contains information as to why the compensation action failed. It can help you to decide what actions to take to correct the failed compensation action. The following administrative actions are available:

Skip Compensating Action

Skip the current compensating action and continue with compensating the process instance.

Retry Compensating Action

If you have taken action to correct the failed compensation action, click **Retry Compensating Action** to try the compensation action again.

Stop Compensation

Stop the compensation process.

Update Compensation Environment

Retry compensation with a different environment.

2. Select the process and then click one of the available actions.

If you selected **Skip Compensating Action**, this might result in a non-compensated activity.

Terminating a process instance

Before you begin

To terminate a process instance, you must have appropriate access rights as a process administrator.

You might want to terminate a process instance 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.

To terminate a flow:

- Click **Terminate** on the Process Instance page.
- On any of the process instance list pages, for example, Administered By Me, select the check box next to the process instance, then click **Terminate**.

Results

This stops the process instance immediately without waiting for any outstanding activities. Process instances that are terminated are not compensated.

Querying process instances

You can define queries to find process instances that meet a certain set of characteristics. For example, you can define a query to find all the process instances that are currently in a failed state.

Steps for this task

1. Click **Define Process Instance List** under Process Instance Lists in the navigation pane.

This action displays the Define Process Instance List page.

2. Specify criteria for querying work items and click **Show List**.

This action displays the User-Defined Process Instance List page.

Note: This page displays only those process instances that meet **all** of your search criteria.

3. **(Optional)** You can work with the process instances found by the query. If you are authorized as the Process Choreographer system administrator, you can also save the query so that it appears under Process Instance Lists.

Working with process templates

This set of topics describes how you can use the Process Choreographer Web client to work with templates for business processes.

You can use the Web client to display information about process templates. If you are authorized to do so, you can start a process instance.

You use the Web client to do the following:

- View a business process template
- Start a business process
- **5.0.2** Query process templates

Displaying information about a business process template

To display information about a process template in the Process Choreographer Web client, complete the following steps:

Steps for this task

1. Click **My Templates** under Process Template Lists in the navigation pane.
This displays the My Templates page.
2. Click the process template name.
 - 5.0+ 5.0.1 +** Select a process template from the Templates drop-down list in the navigation pane and click **View**.
3. Select a process template and click **View Information**.

Results

This task displays a Process Template page with information about the process template. If you are authorized to start a process from the template, a **Start** action is also displayed.

Starting a new process instance

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

To start a new process instance, complete the following steps:

Steps for this task

1. Click **My Templates** under Process Templates Lists in the navigation pane.
This displays the My Templates page.
2. Select a process template.
 - 5.0.2** Click **Start** to display the Process Input Message page, which provides a description of the process template, fields for you to change the input message properties, and a button for you to complete starting the process instance.
 - 5.0+ 5.0.1 +** Select a process template from the Templates drop-down list in the navigation pane and click **Start**. This displays the Process Input Message page in the content pane, which provides a description of the process template, fields for you to change the input message properties, and a button for you to complete starting the process instance.

3. **(Optional)** If the process is an interruptible process, you can type in a process instance name.
4. Complete the input for the process input message.
5. To complete starting the process, click **Start**.

Results

If the business process contains an activity that requires human interaction, a work item is added to the To Do list of potential owners. If you are one of those potential owners, you can display your My To Dos page to work with the work item.

If the process instance is a non-interruptible process, it displays an output message when the process ends.

Querying process templates

You can define queries to find process templates that meet a certain set of characteristics. For example, you can define a query to find all the process templates that are valid from a certain date.

Steps for this task

1. Click **Define Template List** under Process Template Lists in the navigation pane.
This action displays the Define Process Template List page.
2. Specify criteria for querying process templates and click **Show List**.
This action displays the User-Defined Process Templates List page.
Note: This page displays only those process templates that meet **all** of your search criteria.
3. **(Optional)** You can work with the process templates found by the query. If you are authorized as the Process Choreographer system administrator, you can also save the query so that it appears under Process Template Lists.

Customizing the Process Choreographer Web client

Process Choreographer provides a ready-to-use Web user interface based on JSPs and servlets. You can use the Web interface as is or adapt it to fit your needs. Typically, user interfaces for business process applications often require customization, for example, to adapt the user interface to fit a certain look and feel.

The following tasks describe the different ways in which you can customize the Process Choreographer Web client:

- Adapt the look and feel
- Create user-defined JSPs

Adapting the look and feel

Process Choreographer provides a ready-to-use Web user interface based on JSPs and servlets. You can adapt the user interface to fit a certain look and feel without having to write any new JSPs. The Web client interface consists of a header, a navigation pane, and a content pane. A style sheet controls how the Web interface is rendered.

Steps for this task

1. **(Optional)** Modify the header

The Header.jsp is always displayed in the Web client. The default Header.jsp contains logos, images, and a link to the WebSphere Application Server Enterprise InfoCenter.

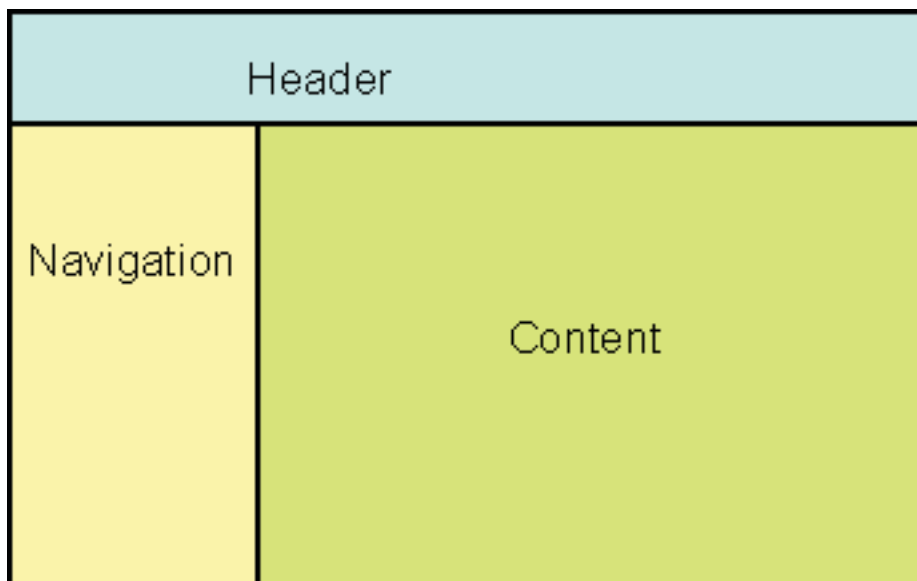
2. **(Optional)** Modify the style sheet

The default style sheet, style.css, contains styles for headings, paragraphs, and tables. It also contains classes, mainly for table and table-cell elements.

Layout of the Process Choreographer Web client user interface

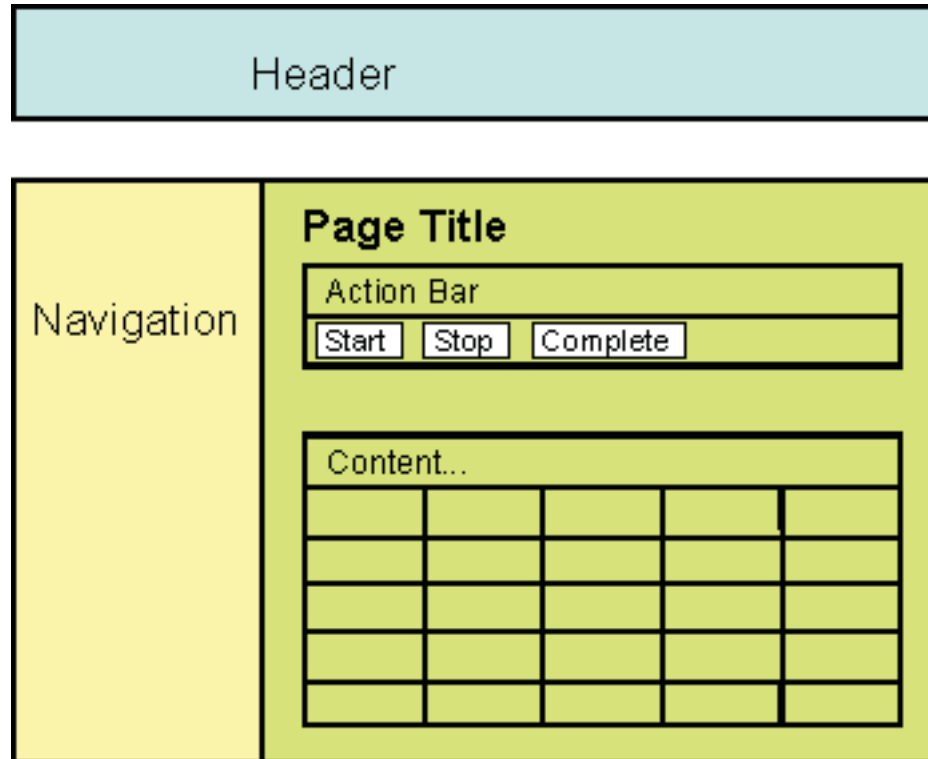
You can use the Process Choreographer Web client to work with business processes that you have deployed as applications.

The Web client interface consists of a header, a navigation pane, and a content pane:



The header and the navigation panes are always displayed. They are generated by Header.jsp and Navigation.jsp. Other JSPs use the `<jsp:include page=xxx>` tag to reference them. The information shown in the content pane depends on the JSP that is used to generate the page.

The layout of the Web Client is implemented with HTML tables. Each page consists of the main table and the Header.jsp pane:



The main table has one row and one column and includes the Navigation.jsp pane. It also provides a table cell for the page content. Depending on the content, this cell can also contain tables, forms, and labels. The HTML template for the page layout looks as follows:

```
<body>
  <jsp:include page="Header.jsp" flush="true"/>
  <table class="page">
    <tr>
      <jsp:include page="Navigation.jsp" flush="true"/>
      <td class="content">
        ...
      </td>
    </tr>
  </table>
</body>
```

Creating user-defined JSPs

When you model a process in WebSphere Studio Application Developer Integration Edition you can define business-process specific JSPs (or user-defined JSPs) for the process and for staff activities. The Web client uses these JSPs to display input and output messages for the process and for the activities for which user-defined JSPs have been defined. For example, if a message has non-primitive parts, it is recommended that you use user-defined JSPs to enhance the usability of the message for the user. You can also use user-defined JSPs to extend the Web client, for example, user input can be checked for correctness before it is sent to the business process container. If you have not defined any user-defined JSPs, the Web client generates a default rendering to display message data.

A message-mapping JSP is required for each user-defined JSP that receives user input data (either a JSP that displays a process input message or an activity output

message). The message-mapping JSP receives the user data, validates the input data, wraps it in an appropriate message object, and then forwards the message to the business process container.

User-defined JSPs are not self-contained Web pages. To include user-defined JSPs in the Web client, you must specify them when you model a business process in Studio Application Developer.

Steps for this task

1. **(Optional)** Create JSPs for displaying messages
2. **(Optional)** Create JSPs for processing user input
3. Integrate user-defined JSPs in the process model

Creating JSPs for displaying messages

The Process Choreographer Web client displays read-only messages for various purposes. For example, the process input message (after the process has been started) and the process output message cannot be edited by users. Some of these messages require further information to enhance their usability. You can create user-defined JSPs to display this additional information.

A user-defined JSP that displays messages and further information about these messages receives the message data with the help of the BusinessProcess session bean's remote interface.

Steps for this task

1. Make the BusinessProcess session bean's home remote interface available to the JSP.

```
BusinessProcess process = MessageUtilities.getProcess(request);
```

`com.ibm.bpe.client.MessageUtilities` is a public class in the package `com.ibm.bpe.client`. It provides several static methods that support message handling. The parameter `request` that is passed to the `getProcess()` method is the implicit `HttpServletRequest` object that is available in all JSPs.

2. Receive the input or output messages of the process or activity using either the `getInputMessage()` or the `getOutputMessage()` method.

When user-defined JSPs are called, they receive the ID of the object they are displaying. The ID can be received from the `HttpServletRequest` object by calling the `getParameter()` method. If the JSP is related to an activity, the activity instance ID (AIID) comes as a string with the `HttpServletRequest`. Similarly, JSPs that work with processes receive the process instance ID (PIID) of the process as a string. For example, to receive the process output message of a process as `org.apache.wsif.base.WSIFDefaultMessage`:

```
String piid = request.getParameter(Constants.WF_PIID);  
WSIFDefaultMessage outMsg = (WSIFDefaultMessage)process.getOutputMessage(piid).getObject();
```

The ID of the process is received using the `getParameter()` method. The `getOutputMessage(String piid)` method in `com.ibm.bpe.api.BusinessProcess` to return a `com.ibm.bpe.api.ClientObjectWrapper` object. This `ClientObjectWrapper` object contains a `WSIFDefaultMessage` object. To receive this message object, the `getObject()` method is called.

3. Access the message parts.

For example, a process output message has the following structure:

outputMessage	java.lang.String
flowID	int

You can access the message parts as shown in the following code snippet:

```
String outputMessage = (String)msg.getObjectPart("outputMessage");
int flowID = msg.getIntPart("flowID");
```

Creating JSPs for processing user input

User input fields often require additional information to enable users to understand the purpose of these fields. You can use user-defined JSPs to provide additional information about message parts in the standard Web Client. A typical user-defined JSP for displaying user input data can provide a detailed description for each type of input field including entry fields, check boxes, and radio buttons. General information about the process or the activity to which the input data is related can also be displayed.

Steps for this task

1. Make the BusinessProcess session bean's home remote interface available to the JSP.

```
BusinessProcess process = MessageUtilities.getProcess(request);
```

`com.ibm.bpe.client.MessageUtilities` is a public class in the package `com.ibm.bpe.client`. It provides several static methods that support message handling. Methods are available that allow access to process-specific and activity-specific data.

2. Display additional information about the process or activity.

The following code snippet is part of a user-defined JSP that uses information about the current activity to display the activity output message. Depending on the state of the activity, the JSP displays the information in different ways:

- When the activity is in state `READY`, general text about the activity's purpose is displayed.
- When the activity has been claimed, two radio buttons that correspond to `true` and `false`, respectively, are also displayed.
- After the activity has been finished, the option that has been chosen is presented as text.

The highlighted text shows the lines of code that provide this functionality:

```
<%
String aaid = request.getParameter("WF_AIID");
BusinessProcess process = MessageUtilities.getProcess(request);
ActivityInstanceData activity = process.getActivityInstance(aaid);
%>
<p>
A user has placed a stock order with a total estimated purchase price
of more than $100000.
</p>
<%if(activity.getExecutionState() == ActivityInstanceData.STATE_READY){%>
<p>
This order must be approved.
</p>
<%if(activity.getExecutionState() == ActivityInstanceData.STATE_CLAIMED){%>
<p>
This order must be approved. What do you want to do?
<label>
<input type="radio" name="Approved" value="true">
```

```

    Approve the order.
  </label>
  <label>
    <input type="radio" name="Approved" value="false" checked>
    Reject the order.
  </label>
</p>
<%}if(activity.getExecutionState() == ActivityInstanceData.STATE_FINISHED){>
  if (outMsg.getBooleanPart("approved")){%>
    <p>
    This order has been approved.
    </p>
  <%} else {%>
    <p>
    This order has not been approved.
    </p>
  <%}
} %>

```

What to do next

Create a message-mapping JSP for each user-defined JSP that processes user input.

Integrating user-defined JSPs in the business process model

To include user-defined JSPs in the Web client, you must specify them when you model a business process in WebSphere Studio Application Developer Integration Edition.

Steps for this task

1. Include user-defined JSPs for a process.

If you have created user-defined JSPs for a process:

- a. In the Process Editor, select the Client tab, then click **Add** to add user-defined settings to the process definition.
The Add Definition dialog box appears.
- b. Select the action for which you want to specify user-defined settings.
Fields for the user-defined JSPs appear.
- c. Select the JSPs that you want to use for the specified action.

The specified JSPs must be either part of the service project that holds the current process or part of another Web project in Studio Application Developer. If they are part of another project, this Web project must be part of the Enterprise Application project that holds the service project.

You can specify a subset of JSPs. For example, you can specify only an output-message JSP. In this case, the default Web client settings are used for the input message. If you have specified a user-defined JSP that accepts user input, you must also specify a message-mapping JSP.

2. Include user-defined JSPs for activities.

If you have created user-defined JSPs for activities:

- a. Open the process containing the activities in the Process Editor and select the Process tab.
- b. Double-click the activity for which you want to specify user-defined JSPs.
The Properties dialog box for the activity appears.
- c. In the navigation pane, click **Client**.
- d. Click **Add** to add user-defined Web client settings to the activity.
The Add Definition dialog box appears.

- e. Select **ActivityDisplay** and specify the appropriate JSPs.

For activities involving human interaction, you can specify JSPs only for ActivityDisplay.

Message-mapping JSPs

Message-mapping JSPs are required whenever user input fields are displayed by user-defined JSPs. Message-mapping JSPs:

- Receive the data that is provided by the user
- Process data-type checks and validation
- Wrap the data in an appropriate message
- Forward the data

Receiving data

Message data that is provided by the user in the input fields of a user-defined JSP can be received by the subsequent message-mapping JSP in the message-mapping JSP's `javax.servlet.http.HttpServletRequest` object. The following code sample shows how user input data can be accessed in a user-defined JSP:

```
String symbol = request.getParameter("Symbol_Input");
String number = request.getParameter("Number_Input");
String limit = request.getParameter("Limit_Input");
String datestr =request.getParameter("Date_Input");
[...]
```

Processing data

When all the user input is available, it is appropriate to perform type checks and validate the input data. For example, some input fields might be mandatory, or some input fields might be required to only accept values in a certain range or format. You can use a message-mapping JSP to do these checks.

Examples of type checking and data validation are given in the following code sample:

```
if (number.equals("")) {
    throw new ServletException("The required field 'number' has not been completed.
                               Please specify a number.");
}

try {
    int intValueOfNumber = Integer.valueOf(number).intValue();
} catch (NumberFormatException ex) {
    throw new ServletException("The specified number '"+number+"' is not valid. ");
}

SimpleDateFormat formatter = new SimpleDateFormat("MM/dd/yyyy");
Date date = formatter.parse(datestr, new ParsePosition(0));
if (date == null) {
    throw new ServletException("The specified date '"+datestr+"' has an invalid format.
                               Please specify the date as MM/dd/yyyy.");
}
```

When you check and validate input data, it is recommended that you use the built-in error feature of JSPs to display errors that occur. A user-defined error page, `error.jsp`, should be defined in the page header of the message-mapping JSP:

```
<%@ page errorPage="error.jsp" %>
```

Wrap the exceptions that occur when the data is checked in a ServletException. Thus, whenever a ServletException (wrapping an error that occurred during data validation and checking) occurs, the error page is displayed. Detailed information about the error can be displayed on the error page, for example, The required field 'user id' has not been completed. Users can use the browser back button to return to the Web client. They can correct their entries and then send a message (an input or output message of a process or an activity) with the corrected data.

Wrapping data

After the user input is received and validated, the data must be wrapped in a message object that can be processed by the business process container.

An org.apache.wsif.base.WSIFDefaultMessage can be used for all messages. Alternatively, you can use the Java message classes generated by WebSphere Studio Application Developer Integration Edition according to the WSDL message definitions. However, these Java message classes can only be used to send messages. If message data is to be retrieved, for example, it is always wrapped in a WSIFDefaultMessage. Therefore, it is consistent to use a WSIFDefaultMessage to send messages as well.

Suppose a process input message has the following structure:

symbol	java.lang.String
number	int
limit	double
datetime	java.util.Date
id	java.lang.String

The following code snippet shows how to create and fill a WSIFDefaultMessage that wraps these message parts.

```
WSIFDefaultMessage wsifProcessInMessage = new WSIFDefaultMessage();
wsifProcessInMessage.setObjectPart("symbol", symbol);
wsifProcessInMessage.setIntPart("number", intValueOfNumber);
wsifProcessInMessage.setDoublePart("limit", doubleValueOfLimit);
wsifProcessInMessage.setObjectPart("datetime", date);
wsifProcessInMessage.setObjectPart("id", id);
```

Forwarding data

To send a message to the business process container, you can use the static method forwardMessageToController() in com.ibm.bpe.client.MessageUtilities. The signature of this method is:

```
public static void forwardMessageToController(
    javax.servlet.http.HttpServletRequest request,
    javax.servlet.http.HttpServletResponse response,
    java.io.Serializable message,
    java.util.Map furtherParameters,
    java.util.Vector excludeRequestParameters)

    throws javax.servlet.ServletException,
           java.io.IOException
```

- For the request and response parameters, use the implicit HttpServletRequest and HttpServletResponse objects that are available in the current JSP.
- For the message parameter, use a com.ibm.bpe.api.ClientObjectWrapper object that contains a WSIFDefaultMessage object.

- The parameters `furtherParameters` and `excludeRequestParameters` are used for the parameters that are submitted via URL to the next page. Use `furtherParameters` to add parameters to the URL. This can be useful, for example, for transferring error messages. The vector `excludeRequestParameters`, contains the parameters that are to be removed from the URL. If it is empty, all the parameters received on the current page are forwarded to the next page. Add parameters that are used to transfer user input from input fields to this vector.

The following code snippet shows how a process input message might be forwarded:

```
Vector excludeRequestParameters = new Vector();
excludeRequestParameters.add("Symbol_Input");
excludeRequestParameters.add("Number_Input");
excludeRequestParameters.add("Limit_Input");
excludeRequestParameters.add("Date_Input");
excludeRequestParameters.add("ID_Input");
```

Troubleshooting the Process Choreographer Web client

You have encountered an error while using the Process Choreographer Web client. Further information might be available to help you resolve the problem you have encountered.

Note: This information is available in English only.

5.0.2 The error page contains a link to the Technical support search page.

Steps for this task

1. Copy the error message code that is shown on the Web client's Error page to the clipboard.
The error code has the format **BPEcnnnc**, where each **c** is a character and **nnnn** is a 4-digit number.
2. Go to the Technical support search page.
3. Paste the error code into the **Limit by adding search terms** field and click **Go**.
If the search does not return any documents or if the documents do not help you solve your problem, you can provide IBM with information about your error situation so that the technical support information can be updated for future reference.

To provide information on the error, go to the Technical support search page and open the first document in the list.

Note: This information will be used to update the technical support information only. To receive a response to your problem, submit a service request.

Process Choreographer Web client page directory

The Process Choreographer Web client comprises pages that you can use to work with business processes and work items.

All Web client pages contain the following common features:

- Navigation pane
This contains links to lists of work items, processes, and process templates:

- **5.0.2** Work Item Lists
Contains links for working with work items in your To Do list and specifying criteria for queries on work items.
- **5.0.2** Process Instance Lists
Contains links for working with process instances that you can administer or have created and specifying criteria for queries on process instances.
- **5.0.2** Process Template Lists
Contains links for working with process templates and for specifying criteria for queries on process templates.
- **5.0+** Workitem Lists
You can use this section to work with work items in you To Do list and to work with processes that you can administer or have created.
- **5.0+** Templates
You can use this section to display details about a process template or to start a process from a template.
- Content pane
You can use this pane to work with processes and work items. The content depends on the actions that you have taken with the Process Choreographer Web client. When you first start the Web client, this page displays the work items in your To Do list.

Activity page

Use this page to work with a work item and, optionally, to display more information about its corresponding activity or its business process.

This page is displayed if you click a work item in the My To Dos page or the Process Instance page.

The Activity page has the following sections:

Available Actions

Buttons that you can use to work with the work item. An action is available only if you have the authority to perform the action on the work item in its current state. If you are not authorized to perform actions on the work item in its current state, this section is empty. For example:

Claim Displayed if the work item is in a Ready state and you are a potential owner of the work item. If you click **Claim**, you become the work item owner (its state is changed to Claimed) and are responsible for its completion.

Complete

Displayed if the work item is in a Claimed state and you are the owner of the work item. If you have specified the required input, you can click **Complete** to complete the work item.

Save Changes

Displayed if the work item is in a Claimed state and you are the owner of the work item. If you change some of the output message properties, you can click **Save Changes**, to save the changes

without completing the work item. For example, you might save changes to a work item if your work is interrupted or if you need to obtain further information.

If you click **Save Changes**, the Activity page is replaced by the My To Dos page. You can display the Activity page later to complete the work item.

Cancel

Displayed if the work item is in a Claimed state and you are the owner of the work item. You can click **Cancel** to leave the work item without saving any changes you might have made to it.

Process Context

This section is displayed only if only if you are authorized to view information about the business process. It shows information about the work item and its associated process instance and process template. It also provides the following links for more information about the corresponding activity and its process:

- View more details about this activity
- View more details about this process

Activity Input Message

This section displays information about the activity. The content of this section, including the number and types of input fields, depends on the design of the activity in the process template.

Activity Output Message

This section displays fields in the output message. The content of this section, including the number and types of input fields, depends on the design of the activity within the process template.

You can change the fields in this section only if you are the owner or an editor of the work item.

When you click **Complete** to complete this work item, the information in the activity output message is stored; it can be used by any of the subsequent activities in the process.

Activity Information page

Use this page to view details about the activity and its process.

This page is displayed if you click a link for more information about an activity; for example, on the Activity page.

The Activity Information page has the following sections:

Activity Description

This section shows detailed information about an activity. The content of this section depends on the design of the activity in the process template, the state of the activity, and your authority for the activity. For example, this section can display the following details:

Activity Name

The name of the activity.

Description

A short description of the activity

Potential Owners

A list of the user IDs for potential owners of the work item associated with this activity.

It also provides a links for you to work with this activity.

Process Description

This section is displayed only if you are authorized to view information about the business process. It provides information about the properties of the business process that the activity is part of. The content of this section depends on the design of the activity within the process template. For example, this section can display the following details:

Process Instance Name

The identifier for the process.

Description

A brief description of the process.

Template Name

The name of the business process template from which the process was started.

Starter

The user ID that started the process.

State The current state of the process.

Started

The date and time when the process was started.

It also provides a link for you to view more details about this process.

Administered By Me page

Use this page to display information about each process that you can administer and, optionally, to select a process to administer.

To administer a process, click the process identifier in the list. This displays the Process Instance page, which provides details about the process and, if appropriate, provides buttons for you to act on the process. To carry out an action on one or more process instances, select the check box next to the process and click one of the Available Actions buttons.

This page is displayed if you click the Administered By Me link under Process Instance Lists in the navigator.

Applicability of the following list: [Fix Pack 5.0.2 and later]**Available Actions**

Buttons that you can use to work on selected process instances. An action is available only if you have the authority to carry out the action on the selected processes in their current state.

View Select **View** to display further information about the process instance.

Terminate

Select **Terminate** to discontinue the navigation of the process instance.

Delete Select **Delete** to explicitly delete a finished process instance that is not automatically deleted when it completes, or a process that was terminated by a user.

Monitor

Select **Monitor** to view the progress of the activities in the process instance.

Process

The business process identifier for the process. Click the link to display the Process Instance page, which provides further information about the process instance.

State The current state of the process.

Started

The date and time that the process was started.

Process Starter

The user ID of the person who started the process instance.

Compensation in Doubt page

Use this page to act on compensation actions that have failed.

To administer compensation, select the check box next to the process instance and click one of the Available Actions buttons.

This page is displayed if you click **Compensation in Doubt** under Process Instance Lists.

Available Actions

Buttons that you can use to carry out compensation actions. An action is available only if you have the authority to carry out the action on the selected process instance in its current state.

Skip Compensating Action

Skip the current compensating action and continue with compensating the process instance.

Note: This might result in a non-compensated activity.

Retry Compensating Action

If you have taken action to correct the failed compensation action, click **Retry Compensating Action** to try the compensation action again.

Stop Compensation

Stop the compensation process.

Update Compensation Environment

Retry compensation with a different environment.

This is useful if the reason for the failed compensation action is a security authentication timeout. If you have enabled security in WebSphere Application Server, this action updates the environment

for the failed compensation action with the EJB session details that you used to log on to the Process Choreographer Web client.

Process Instance Name

The name of the process instance for which compensation is running.

Reason

The reason the compensation action failed. The information can help you decide what actions to take to correct the failed compensation action.

Created

The date and time that the compensation action was created.

Started

The date and time that the compensation action was started.

Finished

The date and time that the compensation action finished.

Created By Me page

Use this page to display information about each process that you have started and, optionally, to select a process to work on.

To work on a process instance, click its process identifier in the **Process** list. This displays the Process Instance page, which provides details about the process instance and, if appropriate, provides buttons for you to administer the process. To carry out an action on one or more process instances, select the check box next to the process and click one of the Available Actions buttons.

This page is displayed if you click **Created By Me** under Process Instance Lists.

Applicability of the following list: [Fix Pack 5.0.2 and later]

Available Actions

Buttons that you can use to work on selected process instances. An action is available only if you have the authority to carry out the action on the selected processes in their current state.

View Select **View** to display further information about the process instance.

Terminate

Select **Terminate** to discontinue the navigation of the process instance.

Delete Select **Delete** to explicitly delete a finished process instance that is not automatically deleted when it completes, or a process that was terminated by a user.

Monitor

Select **Monitor** to view the progress of the activities in the process instance.

Process

The identifier for the process instance. Click the link to display the Process Instance page, which provides further information about the process instance.

State The current state of the process.

Started

The date and time that the process was started.

Define Process Instance List page

Use this page to specify criteria for querying process instances.

To see the list of process instances that match all of your selection criteria, click **Show List**. This list shows only those process instances that you are authorized to see.

State

Specifies the current state of the process instance. To select more than one state, hold down the Ctrl key and click the states you want to include in your query.

Running

The process instance has been started.

Finished

The process instance has completed normally.

Failed The running process instance has encountered an unhandled fault or a fault node. If compensation is not defined for the process, navigation of the process stops.

Terminated

The process instance is stopped because either its parent process has failed or it has received an explicit terminate request.

Compensated

The failed process instance has been compensated and is now in its end state.

Processes that have an activity in state STOPPED

Select this check box to include processes instances with an activity in state Stopped in the query.

Starter

The principal ID of the starter of the process instance. You can use the wildcard character, percent (%), to represent one or more characters. For example, to find all users whose ID starts with B, you can type B% here.

Template Name

The name of the process template associated with the process instance. You can use the wildcard character, percent (%), to represent one or more characters. For example, to find all process templates that start with P, you can type P% here.

Started

Specifies when the process instances were started. You can specify that the process started before or after a certain date or within a given range of dates.

Define Process Template List page

Use this page to specify criteria for querying process templates.

To see the list of process templates that match all of your selection criteria, click **Show List**. This list shows only those process templates that you are authorized to see.

State

Specifies the current state of the process template. To select more than one state, hold down the Ctrl key and click the states you want to include in your query.

Started

The process template is started.

Stopped

The process template is stopped.

Template Name

The name of the process template. You can use the wildcard character, percent (%), to represent one or more characters. For example, to find all process templates that start with P, you can type P% here.

Valid From

Specifies when the process template is started. You can specify that the process template started before or after a certain date or within a given range of dates.

Define Work Item List page

Use this page to specify criteria for querying work items.

To see the list of work items that match all of your selection criteria, click **Show List**. This list shows only those work items that you are authorized to see.

State

Specifies the current state of the work item. To select more than one state, hold down the Ctrl key and click the states you want to include in your query.

Ready The work item is ready to be claimed.

Claimed

The work item has been accepted by one of its potential owners.

Stopped

The work item has produced an unhandled fault.

Expired

The work item was not completed within the time allocated for it.

Finished

The work item has been completed.

Terminated

The process to which this work item belongs has been terminated.

Failed The process to which this work item belongs has failed.

Work Item Owner

The principal ID of the owner of the work item; only claimed or finished work items have owners. You can use the wildcard character, percent (%), to represent one or more characters. For example, to find all users whose ID starts with B, you can type B% here.

Activated

Specifies when the work item was activated. You can specify that the work item was activated before or after a certain date or within a given range of dates.

My Templates page

Use this page to view process templates that you can work with. If you are authorized to do so, you can start processes from this page. To carry out an action on one or more process templates, select the check box next to the process template and click the Available Actions button.

This page is displayed if you click **My Templates** under Process Templates Lists in the navigation pane.

Available Actions

Button that you can use to work on selected process templates. This action is available only if you have the authority to carry out the action on the selected templates in their current state.

Start Select **Start** to start a process instance from the selected process template. This displays the Process Input Message page, where you can change the properties of the input message before you start the process instance.

Process Template Name

The name of the process template. To view further information about the process template, click the link to display the Process Template page.

Valid-From Date

The date and time when the template was officially put into use.

Interruptible

Specifies whether process instances derived from this template are interruptible.

Auto Delete

Specifies whether process instances derived from this template are automatically deleted when they complete.

State The state of the process template.

Description

A brief description of the process.

Version

The version of the template.

Created

The date and time when the template was created.

My To Dos page

Use this page to work on work items in your To Do list. You can also view more detailed information on work items and their processes from this page.

To work on a work item, click its name in the **To Do Name** list. This displays the Activity page, which provides details about the work item and, if appropriate, provides buttons for you to act on the work item. To carry out an action on one or more work items, select the check box next to the work item and click one of the Available Actions buttons.

This page is displayed if you click the My To Dos link under Work Item Lists in the navigator.

Applicability of the following list: [Fix Pack 5.0.2 and later]

Available Actions

Buttons that you can use to work on selected work items. An action is available only if you have the authority to carry out the action on the selected work items in their current state.

View Click **View** to display further information about the work item.

Claim Available if the work item is in a Ready state and you are a potential owner of the work item. If you click **Claim**, you become the work item owner and are responsible for its completion. Its state is changed to Claimed.

Complete

Available if the work item is in a Claimed state and you are the owner of the work item. If you have specified the required input, you can click **Complete** to complete the activity.

Cancel

Available if the work item is in a Claimed state and you are the owner of the work item. You can click **Cancel** to leave the work item without saving any changes you might have made to it.

Restart

Available if the work item is in a Stopped state and you are the process administrator for the process instance.

Force Complete

Available if the work item is in a Stopped state and you are the process administrator for the process instance.

To Do Name

The name of the work item. This is a link that you can click to work with the work item.

State The current state of the work item.

Work items in Ready state appear on the My To Dos page of all potential work item owners. If you claim such a work item, you become the work item owner (its state is changed to Claimed) and are responsible for its completion.

Activated

The date and time that the work item was activated.

Process

The process instance that the work item belongs to. This is a link that you can click to work with the process instance.

Template Name

The name of the process template from which the process instance was started.

Process Input Message page

Use this page to change the input message before you complete the actions to start the business process.

This page is displayed when you start a business process from the Process Template page that requires an input message.

This page has the following sections:

- **Available Actions**
Click **Start** to start a process instance with the information shown on this page.
- **Process Template Description**
This section provides information about the properties of the process template from which the process is being started.
- **Process Instance Name (optional)**
The name of the new process instance. This field is displayed for interruptible processes only.
- **Process Input Message**
This section provides entry fields for the process input message. The number and types of fields depends on the process template.

Process Instance page

Use this page to display information about the process and, optionally, to act on the process. You can also view the activities for the process and, optionally, select an activity to work with.

This page is displayed if you click a process instance link on the Administered By Me page or Created By Me page or a link for more information about a process.

This page contains the following sections:

Available Actions

Buttons that you can use to work with the process. An action is available only if you have authority to perform the action on the process in its current state. If you are not authorized to perform any actions on the process instance in its current state, this section is empty.

Terminate

Select **Terminate** to discontinue the navigation of the process instance.

Delete Select **Delete** to explicitly delete a finished process instance that is not automatically deleted when it completes, or a process that was terminated by a user.

Monitor

Select **Monitor** to view the progress of the activities in the process instance.

Process Description

A table of information about the process, including:

Process Instance Name

The identifier for the process.

Template Name

The name of the business process template from which the process was started.

Description

A brief description of the process.

Starter

The user ID that started the process.

Administrators

The user IDs of the people who have the authority to carry out administrative actions on the process instance.

State The current state of the process.

Started

The date and time when the process was started.

Process Input Message

Input that is needed to start the process.

Process Output Message

If the process has finished, its output message is shown here.

Activities

Information about the activities that make up the process:

Name The name of the activity. Each name is a link that you can use to select an activity to work with.

State The current state of the activity.

Activated

The date and time when the activity was started.

Completed

The date and time when the activity was completed.

Process Instance Monitor page

Use this page to view the status of activities in a process instance. You can also view more detailed information on activities from this page.

To work on an activity, click its activity name in the **To Do Name** list. This displays the Activity page, which provides details about the activity and, if appropriate, provides buttons for you to act on the activity.

This page is displayed if you click Monitor on the Administered By Me page, the Created By Me page, the Process Instance page, or the User-Defined Process Instance List page.

This page contains the following sections:

Process Description

Information about the process instance, including the starter of the process, the process readers, and the process administrators.

Activities

The activities that the process instance contains with information about each activity. Click an activity to see more information about it.

Process Output Message page

Use this page to view the results of a business process that you started. This page is displayed when a non-interruptible process ends.

The information displayed by the output message depends on the process template that defines the model for the process.

Process Template page

Use this page to view information about a process template, which provides the design for business processes. If you are authorized to do so, you can click **Start** to start processes based on the template from this page.

5.0.2 This page is displayed if you click a process template on the My Templates page.

5.0+ This page is displayed if you select a process template from the Templates drop-down list in the navigation pane and click **View**.

Available Actions

Buttons that you can use to work on selected process templates. An action is available only if you have the authority to carry out the action on the selected templates in their current state.

View Select **View** to display further information about the process template.

Start Select **Start** to start a process instance from the selected process template. This displays the Process Input Message page, where you can change the properties of the input message before you start the process instance.

Process Template Description

Provides information about the process template. This includes when the template was put into use, whether process instances based on the template are interruptible, and if they are automatically deleted when they complete.

User-Defined Process Instance List page

Use this page to work with process instances found by a query. If you are authorized as the Process Choreographer system administrator, you can also save the query so that it appears under Process Instance Lists. To save the query, type a unique name for the query in the **Save List As** field and click **Save**.

This page is displayed if you click **Show List** in the Define Process Instance List page.

This page has the following sections:

Available Actions

Buttons that you can use to work with the process. An action is available only if you have authority to perform the action on the process in its current state.

Terminate

Select **Terminate** to discontinue the navigation of the process instance and stop any activities that are running.

Delete Select **Delete** to explicitly delete a finished process instance that is not automatically deleted when it completes, or a process that was terminated by a user.

Monitor

Select **Monitor** to view the progress of the activities in the process instance.

Process Name

The identifier of the process.

Template Name

The name of the business process template from which the process was started.

Process Starter

The user ID that started the process.

State The current state of the process.

Started

The date and time when the process was started.

User-Defined Process Template List page

Use this page to work with process templates found by a query. If you are authorized as the Process Choreographer system administrator, you can also save the query so that it appears under Process Template Lists. To save the query, type a unique name for the list in the **Save List As** field, then click **Save**.

This page is displayed if you click **Show List** in the Define Process Template List page.

Available Actions

Button that you can use to work with the process.

Start If you are authorized to start a process from this template, the **Start** button is available. To start a process, click **Start**.

Process Template Name

The name of the process template.

Valid-From Date

The date and time when the template was officially put into use.

Interruptible

Whether processes derived from the template are interruptible.

Auto Delete

Whether processes derived from the template are automatically deleted on completion.

State The state of the process template.

Description

A short description of the process template.

User-Defined Work Item List page

Use this page to work with work items found by a query. If you are authorized as the Process Choreographer system administrator, you can also save the query so that it appears under Work Item Lists.

From this page you can:

- Work with a work item. Click the name of the work item in the **To Do Name** list. This displays the Activity page, which provides details about the activity associated with the work item and, if appropriate, provides buttons for you to act on the work item.
- Work with a process. Click the process name under **Process** in the list. This displays the Process Instance page, which provides details about the process and, if appropriate, provides buttons for you to act on the process instance.
- Save the work item list. Type a unique name for the list in the **Save List As** field, then click **Save**. The work item list then appears under My Lists.

This page is displayed if you click **Show List** in the Define Work Item List page.

This page has the following sections:

Available Actions

Buttons that you can use to work on selected work items. An action is available only if you have the authority to carry out the action on the selected work items in their current state.

View Click **View** to display further information about the work item.

Claim Available if the work item is in a Ready state and you are a potential owner of the work item. If you click **Claim**, you become the work item owner (its state is changed to Claimed) and are responsible for its completion.

Complete

Available if the work item is in a Claimed state and you are the owner of the work item. If you have already specified the required user input, you can click **Complete** to complete the work item.

Cancel

Available if the work item is in a Claimed state and you are the owner of the work item. You can click **Cancel** to leave the work item without saving any changes you might have made to it.

Restart

Available if the work item is in a Stopped state and you are the process administrator for the process instance.

Force Complete

Available if the work item is in a Stopped state and you are the process administrator for the process instance.

To Do Name

The name of the work item. This is a link that you can click to work with the work item.

Process

The name of the process instance the work item belongs to. This is a link that you can click to work with the process instance.

Template Name

The name of the template that contains the process.

Activated

The date and time that the work item was activated.

Started

The date and time that the work item was started.

Activity Owner

The owner of the work item.

State The current state of the work item.

Process Choreographer Web client roles and actions

The actions you can take through the Web client depend on the role that you have been assigned; there are roles for processes and roles for work items.

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

no unique ID for: <user ID>

Actions for process roles

Action	Business process administrator	Process administrator	Starter	Reader
Start process	Yes	-	Yes	-
Terminate process	Yes	Yes	-	-
Monitor process (5.0.2 and later)	Yes	Yes	-	-
Delete process	Yes	Yes	-	-
Display process	Yes	Yes	Yes	Yes
Save user-defined lists (5.0.2 and later)	Yes	-	-	-
Force complete stopped activity	Yes	Yes	-	-
Force restart activity	Yes	Yes	-	-
Claim work item	Yes	Yes	-	-
Save work item	Yes	Yes	-	-
Display work item	Yes	Yes	-	-
Complete work item	Yes	Yes	-	-

Actions for work-item roles

Action	Business process administrator	Potential owner	Owner	Reader	Editor
Claim work item	Yes	Yes	See note	-	-
Save work item	Yes	Yes	Yes	-	Yes
Display work item	Yes	Yes	Yes	Yes	Yes
Complete work item	Yes	Yes	Yes	-	-
Force restart work item	Yes	-	-	-	-
Force complete stopped work item	Yes	-	-	-	-
Note: If a potential owner claims a work item, the potential owner becomes the owner of the work item.					

Chapter 7. Developing applications using the Process Choreographer API

The Process Choreographer API provides the following renderings for developing applications:

- An EJB rendering that allows the API to be called remotely using IIOP. A stateless session bean, `BusinessProcess`, exposes the functions that can be called by an application program.
- A JMS rendering that allows a subset of the API functions to be called remotely using JMS.

You can use the API to develop the following types of applications:

- Business-process applications for non-interruptible processes
- Business-process applications for interruptible processes
- Administration applications for interruptible processes

For more information, see the Javadoc for Process Choreographer in the `com/ibm/bpe/api` package package.

Accessing the Process Choreographer EJB interface

An application accesses the `BusinessProcess` session bean through the bean's home and remote interfaces.

Steps for this task

1. Add a reference to the `BusinessProcess` session bean to your application deployment descriptor.

Add the reference to:

- `application-client.xml`, for a J2EE client application
- `web.xml`, for a Web application
- `ejb-jar.xml`, for an EJB application

Add the reference as follows:

```
<ejb-ref>
  <ejb-ref-name>ejb/BusinessProcessHome</ejb-ref-name>
  <ejb-ref-type>Session</ejb-ref-type>
  <home>com.ibm.bpe.api.BusinessProcess</home>
  <remote>com.ibm.bpe.api.BusinessProcess</remote>
</ejb-ref>
```

2. Make the `BusinessProcess` session bean's home interface available to the application.

You can do this using JNDI-lookup mechanisms:

```
// Obtain the default initial JNDI context
Context initialContext = new InitialContext();

// Lookup the home interface of the BusinessProcess bean
Object result = initialContext.lookup("java:comp/env/ejb/BusinessProcessHome");

// Convert the lookup result to the proper type
BusinessProcessHome processHome =
    (BusinessProcessHome)javax.rmi.PortableRemoteObject.narrow(result,BusinessProcessHome.class);
```

The BusinessProcess session bean's home interface contains a create method for EJB objects. The method returns the session bean's remote interface.

3. Access the BusinessProcess session bean's remote interface:

```
BusinessProcess process = processHome.create();
```

4. Call the business functions exposed by the BusinessProcessService interface, for example:

```
process.initiate("MyProcessModel",input);
```

Calls from applications are executed as transactions. A transaction is established and ended in one of the following ways:

- Automatically by WebSphere (the deployment descriptor specifies TX_REQUIRED).
- Explicitly by the application. You can bundle application calls into one transaction:

```
// Obtain user transaction interface
UserTransaction transaction= (UserTransaction)initialContext.lookup("jta/usertransaction");

// Begin a transaction
transaction.begin();

// Applications calls ...

// On successful return, commit the transaction
transaction.commit();
```

Accessing the Process Choreographer JMS interface

Process Choreographer accepts JMS messages that follow the point-to-point paradigm. An application that sends or receives JMS messages must:

Steps for this task

1. Create a connection to Process Choreographer.

Use JNDI lookup to retrieve the connection factory. The JNDI-lookup name must be the same as the name specified when the Process Choreographer external request queue is configured.

```
//Obtain the default initial JNDI context
Context initialContext = new InitialContext();

// Look up the connection factory
QueueConnectionFactory queueConnectionFactory =
    (QueueConnectionFactory) initialContext.lookup("jms/BPECF");

// Create connection
QueueConnection queueConnection =
    queueConnectionFactory.createQueueConnection();
```

2. Create a session so that message producers and consumers can be created.

```
//Create a nontransacted session using autoacknowledgement
QueueSession queueSession =
    queueConnection.createQueueSession( false, Session.AUTO_ACKNOWLEDGE );
```

3. Create a message producer to send messages.

The JNDI-lookup name must be the same as the name specified when the Process Choreographer external request queue is configured.

```
// Look up the destination of the Process Choreographer queue to send messages to
Queue bpeQueue = (Queue) initialContext.lookup("jms/BPEApiQueue");
// Create message producer
QueueSender queueSender = queueSession.createSender(bpeQueue);
```

4. Create a message consumer to receive replies.

The JNDI-lookup name must specify a user-defined destination to send replies to.


```

// Look up the destination of the reply to queue
Queue replyToQueue = (Queue) initialContext.lookup("MyReplyToQueue");
// Create message consumer
QueueReceiver queueReceiver = queueSession.createReceiver(replyToQueue);
5. Send requests and receive replies.
// Start sending and receiving messages
queueConnection.start();

// Create message - see the task descriptions for examples - and send
ObjectMessage message = queueSession.createObjectMessage();
// message.SetStringProperty(..);
// message.setObject(..);

queueSender.send(message);

// Receive message and analyze reply.
// See the detailed task descriptions for examples
Message reply = queueReceiver.receive();
6. Close the connection to the free resources.
// Final housekeeping: free resources
queueConnection.close();

```

Developing applications for non-interruptible processes

You can develop the following applications for non-interruptible processes:

- Execute a non-interruptible process using the EJB interface
- Execute a non-interruptible process using the JMS interface

Also see the Javadoc for Process Choreographer in the `com/ibm/bpe/api` package.

Executing a non-interruptible process using the EJB interface

Steps for this task

1. **(Optional)** List the process templates to find the name of the non-interruptible process you want to execute.

This step is optional if you already know the name of the process.

```

ProcessTemplateData[] processTemplates = process.queryProcessTemplates(
    "PROCESS_TEMPLATE.CAN_RUN_SYNC=TRUE",
    "PROCESS_TEMPLATE_NAME",
    new Integer(50),
    null);

```

The results are sorted by name. The query returns an array containing the first 50 sorted templates that can be started as non-interruptible processes.

2. Start the process with an input message.

In the following, `Customer` and `OrderNo` are message types known to the system.

```

Customer input = new Customer("Smith");
...
ClientObjectWrapper output = process.call("CustomerTemplate", new ClientObjectWrapper(input));
OrderNo order = (OrderNo) output.getObject();

```

This creates an instance of the process template, `CustomerTemplate`, and passes some customer data. The operation returns only when the process is complete. The result of the process, `OrderNo`, is returned to the caller.

Executing a non-interruptible process using JMS

Steps for this task

1. Create a message, for example, an `ObjectMessage`.
`ObjectMessage message = queueSession.createObjectMessage();`
2. **(Optional)** Set the `JMSReplyToQueue`.
If you do not want to receive a reply, this step is optional.
`//Specify the destination object replies are to be sent to
message.setJMSReplyTo(replyToQueue);`
3. **(Optional)** Specify the JMS properties.
If you do not specify any properties, the process template name **Dispatch** is assumed. If `JMSReplyToQueue` is set, **call** is issued. If `JMSReplyToQueue` is not set, **initiate** is issued.

```
message.setStringProperty("wf$verb", "call");  
message.setStringProperty("wf$processTemplateName", "CustomerTemplate");
```

4. Start the process with an input message.
Specify the input message as the body, the payload, of the message. In the example, `Customer` is a message type known to the system.

```
//Create Customer input message  
Customer input = new Customer();  
input.setLastName("Smith");  
  
message.setObject(new ClientObjectWrapper(input));  
  
//Send message  
queueSender.send(message);
```

This creates an instance of the process template, `CustomerTemplate`, and passes some customer data. The operation returns only when the flow is complete and when a `JMSReplyToQueue` is specified. The result of the process, `OrderNo`, is returned as the payload of the reply message. Because an `ObjectMessage` was passed, an object message is returned.

5. **(Optional)** Get the result of the process.
You can get the results of the process only if you specified a queue in step 2. In the example, `OrderNo` is a message type known to the system.

```
Message m = queueReceiver.receive();  
if (m instanceof ObjectMessage)  
{  
    ClientObjectWrapper wrapper = (ClientObjectWrapper)m.getObject();  
    OrderNo output = (OrderNo)wrapper.getObject();  
}
```

Characteristics of non-interruptible business processes

A non-interruptible business process has the following characteristics:

- Runs as one transaction.
- Consists of only synchronous services, EJBs, Java snippets, empty activities, and non-interruptible subprocesses.
- Usually started using the **call** method so that an output message is returned when the process is complete.
- Normally short running.
- Is not visible during execution because run-time values are not stored in the database.
- Cannot contain interruptible processes.

Developing applications for interruptible processes

You can use the EJB interface to develop the following services for interruptible processes:

- Start an interruptible process
- Process a person activity
- Send an event to a process instance
- Analyze the result of a process
- Use work lists to query information

You can use the JMS interface to develop the following services for interruptible processes:

- Start an interruptible process
- Send an event to a process instance
- Analyze the result of a process

For more information, see the Javadoc for Process Choreographer in the `com.ibm/bpe/api` package.

Starting an interruptible process using the EJB interface

Steps for this task

1. **(Optional)** List the process templates to find the name of the interruptible process you want to start.

This step is optional if you already know the name of the process.

```
ProcessTemplateData[] processTemplates = process.queryProcessTemplates
("PROCESS_TEMPLATE.CAN_RUN_INTERRUPTIBLE=true"
 "PROCESS_TEMPLATE.NAME",
 new Integer(50),
 null);
```

The results are sorted by name. The query returns an array containing the first 50 sorted templates that can be started as interruptible processes.

2. Start the process with an input message of the appropriate type.

Note: If you specify a process-instance name, it must not start with an underscore. If a process-instance name is not specified, the PIID in String format is used as the name.

```
Customer input = new Customer("Smith");
PIID piid =
    process.initiate("CustomerTemplate", "CustomerOrder", new ClientObjectWrapper(input));
```

This creates an instance, `CustomerOrder`, of the process template, `CustomerTemplate`, and passes some customer data. When the process starts, the operation returns the object ID of the new process instance to the caller.

The starter of the process instance is set to the caller of the request and receives a work item for the process instance. The process administrators, readers, and editors of the process instance are determined and receive work items for the process instance. The starting activity instances are determined and either started automatically or, if they are person activities or receive events, work items are created for the potential owners.

Processing person activities using the EJB interface

Steps for this task

1. List the activities belonging to a logged-on person that are ready to be worked on:

```
QueryResultSet result = process.query("ACTIVITY.AIID",
    "ACTIVITY.STATE = ACTIVITY.STATE.STATE_READY
    AND WORK_ITEM.REASON = WORK_ITEM.REASON.REASON_POTENTIAL_OWNER",
    null,
    null,
    null);
```

This returns a query result set that contains the activities that can be started by the logged-on person.

2. Claim the activity to be worked on:

```
if (result.size() > 0)
{
    result.first();
    AIID aaid = (AIID) result.getOID(1);
    {
        ClientObjectWrapper input = process.claim(aaid);
        Order activityInput = (Order) input.getObject();
    }
}
```

When the activity is claimed, the input message of the activity is returned.

3. When work on the activity is complete, complete the activity.

```
OrderNo output = new OrderNo(4711);
process.complete (aaid, new ClientObjectWrapper(output));
```

Sending an event to a process instance using the EJB interface

Steps for this task

1. **(Optional)** List the processes that are waiting for a specific event from the logged-on user.

```
QueryResultSet result = query("DISTINCT EVENT.PIID",
    "EVENT.NAME = 'OrderEvent'
    AND WORK_ITEM.REASON = WORK_ITEM.REASON.REASON_POTENTIAL_OWNER",
    null,
    null,
    null);
```

2. Send an event.

The caller must be a potential owner of the awaited OrderEvent or an administrator of the process instance, CustomerOrder.

```
if (result.size() > 0)
{
    result.first();
    Order input = new Order("Chocolate");
    process.sendEvent ((PIID)result.getOID(1), "OrderEvent", new ClientObjectWrapper(input));
}
```

Sends the specified OrderEvent to the waiting process instance and passes some order data.

Analyzing results of a process using the EJB interface

An interruptible process runs asynchronously. Its output message is not automatically returned when the process completes. The message must be retrieved explicitly.

Note: The results of the process are stored in the database only if the process template from which the process instance was derived does not specify automatic deletion of the output message.

Steps for this task

1. Analyze the results of the process:

```
QueryResultSet result = process.query("PROCESS_INSTANCE.PIID",
    "PROCESS_INSTANCE.NAME = 'CustomerOrder' AND
    PROCESS_INSTANCE.STATE = PROCESS_INSTANCE.STATE.STATE_FINISHED",
    null,
    null,
    null);

if (result.size() > 0)
{
    result.first();
    PIID piid = (PIID) result.getOID(1);
    ClientObjectWrapper output = process.getOutputMessage(piid);
    OrderNo order = (OrderNo) output.getObject();
}
```

Using worklists to query information

A worklist is a query that is persistently stored in the database. It represents a set of items which have the same characteristics. Although worklist definitions are stored persistently, items contained in the worklist are assembled dynamically when they are queried. All worklists are publicly accessible.

Steps for this task

1. **(Optional)** List the available worklists:

```
String[] worklists = getWorklistNames();
```

2. **(Optional)** Check the query defined by a specific worklist:

```
WorkListData worklist = process.getWorklist("CustomerOrdersStartingWithA");
String selectClause = worklist.getSelectClause();
String whereClause = worklist.getWhereClause();
String orderByClause = worklist.getOrderByClause();
Integer threshold = worklist.getThreshold();
```

3. Run the query defined by the worklist:

```
QueryResultSet result =
process.executeWorklist("CustomerOrdersStartingWithA");
```

Starting an interruptible process using the JMS interface

Steps for this task

1. Create a message, for example, an ObjectMessage.

```
ObjectMessage message= queueSession.createObjectMessage();
```

2. **(Optional)** Set the JMSReplyToQueue.

If you do not want to receive a reply, this step is optional.

Note: You cannot specify a temporary queue as a reply-to queue.

```
// Specify the destination object replies are to be sent to
message.SetJMSReplyTo(replyToQueue);
```

3. Set the JMS properties.

If you do not specify any properties, the process-template name **Dispatch** is assumed. If JMSReplyToQueue is set, **call** is issued. If JMSReplyToQueue is not set, **initiate** is issued. If the process-instance name is set, it must not start with an underscore. If the process-instance name is not set, the PIID in String format is used as the name.

```
message.SetStringProperty("wf$verb", "initiate");
message.SetStringProperty("wf$processTemplateName", "CustomerTemplate");
message.SetStringProperty("wf$processInstanceName", "CustomerOrder");
```

4. Start the process with an input message.

Specify the input message as the body, the payload, of the message. In the following, Customer is a message type known to the system.

```
// Create Customer input message
Customer input= new Customer();
input.setLastName("Smith");

message.setObject(new ClientObjectWrapper(input));

// Send message
queueSender.send(message);
```

This creates an instance, `CustomerOrder`, of the process template, `CustomerTemplate`, and passes some customer data. The operation returns the object ID of the newly created instance as the value of the JMS property `wf$piid` if a `JMSReplyToQueue` is specified. The reply message does not contain a payload.

5. Get the result of the process initiation.

```
Message m = queueReceiver.receive();
String fiid = m.getStringProperty("wf$piid");
```

Sending an event to a process instance using the JMS interface

Steps for this task

1. Create a message, for example, an `ObjectMessage`.

```
ObjectMessage message= queueSession.createObjectMessage();
```

2. **(Optional)** Set the `JMSReplyToQueue`.

If you do not want to receive a reply, this step is optional.

```
// Specify the destination object replies are to be sent to
message.setJMSReplyTo(replyToQueue);
```

3. Set the JMS properties.

You can specify the PIID of the process instance instead of the process-instance name. If you specify both properties, the `processInstanceName` is used.

```
message.setStringProperty("wf$verb", "sendEvent");
message.setStringProperty("wf$processInstanceName", "CustomerOrder");
message.setStringProperty("wf$event", "OrderEvent");
```

4. Send the specified `OrderEvent` to the process instance, `CustomerOrder`.

```
// Create event input message
Order input = new Order("Chocolate");

message.setObject(new ClientObjectWrapper(input));

// Send message
queueSender.send(message);
```

If `JMSReplyToQueue` is set, this returns an empty reply message if the event was sent successfully. The `JMSCorrelationID` is set to the `JMSMessageID` of the `sendEvent` request. Neither properties nor payload are set on the reply message.

Analyzing results of a process using the JMS interface

An interruptible process runs asynchronously. Its output message is not automatically returned when the process completes. The message must be retrieved explicitly. You can get the results of the process only if you specified a `JMSReplyToQueue` and used the `call` verb to instantiate the process.

Steps for this task

1. Analyze the results of the process:

In the example, OrderNo is a message type known to the system. When an ObjectMessage is passed in the request, an object message is returned.

```
Message m = queueReceiver.receive();
if (m instanceof ObjectMessage)
{
    ClientObjectWrapper wrapper = (ClientObjectWrapper)m.getObject();
    OrderNo output = (OrderNo)wrapper.getObject();
}
```

Characteristics of interruptible processes

An interruptible process has the following characteristics:

- Runs as several transactions.
- Can consist of services, EJBs, Java snippets, event, person, empty, and process activities.
- Usually started using the **initiate** method because the output message cannot be retrieved synchronously.
- Normally long running.
- Is visible during execution because run-time values are stored persistently.

Event activities

An event is an asynchronous notification that can be sent to a process instance. It is used to synchronize the execution of a process instance with the systems external to it. An event activity waits for the occurrence of an external event, several event activities might be waiting for the same event. All these activities receive the external event when it is sent. The event is consumed; subsequent event activities waiting for the same external event require a new event with the same name to be sent.

Person activities

When a person activity is activated, the process engine creates work items and distributes them to the potential owners of the activity. One of these people claims the associated activity, works with the associated data and components, and completes the activity. The completion of the activity triggers the next transaction and continues the process. You can only use the EJB interface to process person activities.

Developing administration applications for interruptible processes

You can develop the following administration applications for interruptible processes. You can use the EJB interface to render all of these applications. The JMS interface can be used only to terminate a process instance.

- Cancel a claimed activity
- Force the completion of an activity
- Retry the execution of a stopped activity
- Delete a process instance
- Terminate a process instance using the EJB interface
- Terminate a process instance using the JMS interface
- Manage work lists

For more information, see the Javadoc for Process Choreographer in the `com/ibm/bpe/api` package.

Canceling a claimed activity

Sometimes it is necessary for someone with process administrator rights to cancel an activity that has been claimed by someone else. This might happen, for example, when an activity must be completed but the owner of the activity is absent.

Steps for this task

1. List the claimed activities owned by a specific person to find the ID of the activity in question.

```
QueryResultSet result = process.query("DISTINCT ACTIVITY.AIID",
    "ACTIVITY.STATE = ACTIVITY.STATE.STATE_CLAIMED AND
    ACTIVITY.OWNER = 'Smith'
    AND ACTIVITY.TEMPLATE_NAME = 'CustomerTemplate'",
    null,
    null,
    null);
```

This returns a query result set that lists the activities claimed by the specified person, Smith.

2. Cancel the claimed activity.

```
if (result.size() > 0)
{
    result.first();
    AIID aaid = (AIID) result.getOID(1);
    process.cancelClaim(aaid);
}
```

This returns the activity to the ready state so that it can be claimed by one of the other potential owners.

Forcing the completion of an activity

If an activity in an interruptible process encounters a system exception or an unconnected fault terminal and the associated activity template specifies that the activity should stop when an error occurs, the activity is put into the stopped state so that it can be repaired. You can force completion of the activity. You can also pass parameters in the force-complete call, such as the message that should have been computed or the fault that should have been raised.

Steps for this task

1. List the stopped activities.

```
QueryResultSet result = process.query("DISTINCT ACTIVITY.AIID",
    "ACTIVITY.STATE = ACTIVITY.STATE.STATE_STOPPED AND
    PROCESS_INSTANCE.NAME='CustomerOrder'",
    null,
    null,
    null);
```

This returns the stopped activities for the process instance, CustomerOrder.

2. Complete the activity.

In this example an output message is passed.

```
if (result.size() > 0)
{
    result.first();
    AIID aaid = (AIID) result.getOID(1);
    OrderNo output = new OrderNo(4711);
    process.forceComplete(aaid, new ClientObjectWrapper(output), true);
}
```


For more information, see the Javadoc for Process Choreographer in the `com/ibm/bpe/api` package.

This completes the activity. If a system error occurs, the `continueOnError` parameter determines whether the activity stays in the stopped state. In the example, `continueOnError=true`. This means that if an error occurs during processing of the `forceComplete` request, the activity is put into the failed execution state. Navigation continues and the process is put into the failing state.

Retrying the execution of a stopped activity

If an activity in an interruptible process encounters a system exception or an unconnected fault terminal and the associated activity template specifies that the activity should stop when an error occurs, the activity is put into the stopped state so that it can be repaired. You can retry the execution of the activity by passing a new input message.

Steps for this task

1. List the stopped activities.

```
QueryResultSet result = process.query("DISTINCT ACTIVITY.AIID",
    "ACTIVITY.STATE = ACTIVITY.STATE.STATE_STOPPED AND
    PROCESS_INSTANCE.NAME='CustomerOrder'",
    null,
    null,
    null);
```

This returns the stopped activities for the process instance `CustomerOrder`.

2. Retry the execution of the activity.

```
if (result.size() > 0)
{
    result.first();
    AIID aaid = (AIID) result.getOID(1);
    Order input = new Order("Chocolate");
    process.forceRetry(aaid, new ClientObjectWrapper(input), true);
}
```

For more information, see the Javadoc for Process Choreographer in the `com/ibm/bpe/api` package.

This retries the activity. If a system error occurs, the `continueOnError` parameter determines whether the activity stays in the stopped state. In the example, `continueOnError=true`. If an error occurs during processing of the `forceRetry` request, the activity is put into the failed execution state. Navigation continues and the process is put into the failing state.

Deleting a process instance

Processes instances are only automatically deleted when they complete if this is specified in the process template from which the process instances are derived. To delete all finished process instances:

Steps for this task

1. List the process instances that are finished.

```
QueryResultSet result = process.query
("DISTINCT PROCESS_INSTANCE.PIID",
"PROCESS_INSTANCE.STATE = PROCESS_INSTANCE.STATE.STATE_FINISHED",
null,
null,
null);
```

This returns a query result set that lists finished process instances.

2. Delete the finished process instances.

```
PIID piid;
while (result.next() )
{
    piid = (PIID) result.getOID(1);
    process.delete(piid);
}
```

Terminating a process instance using the EJB interface

Sometimes it is necessary for someone with process administrator rights to terminate a process instance that is known to be in an unrecoverable state. For example, when an application is invoked and fails and does not return to a dormant state.

It is recommended that you terminate a process instance only in exceptional situations. The process instance is terminated immediately without waiting for any outstanding activities. Process instances that are terminated are not compensated.

Steps for this task

1. Retrieve the process instance to be terminated.

```
ProcessInstanceData processInstance =
process.getProcessInstance("CustomerOrder");
```

2. Terminate the process instance.

```
PIID piid = processInstance.getID();
process.forceTerminate(piid);
```

The process instance is terminated immediately without waiting for any outstanding activities.

Terminating a process instance using the JMS interface

Sometimes it is necessary for someone with process administrator rights to terminate a process instance that is known to be in an unrecoverable state. For example, when an application is invoked and fails and does not return to a dormant state.

It is recommended that you terminate a process instance only in exceptional situations. The process instance is terminated immediately without waiting for any outstanding activities. Process instances that are terminated are not compensated.

Steps for this task

1. Create a message, for example, an ObjectMessage.

```
ObjectMessage message= queueSession.createObjectMessage();
```

2. **(Optional)** Set the JMSReplyToQueue.

If you do not want to receive a reply, this step is optional.

```
// Specify the destination object replies are to be sent to
message.SetJMSReplyTo(replyToQueue);
```

3. Set the JMS properties.

```
message.SetStringProperty("wf$verb", "forceTerminate");
message.SetStringProperty("wf$processInstanceName", "CustomerOrder");
```

4. Terminate the process instance, CustomerOrder.

```
// Send message
queueSender.send(message);
```

The process instance is terminated immediately without waiting for any outstanding activities. If `JMSReplyToQueue` is set, this returns an empty reply message if the process instance was terminated successfully. The `JMSCorrelationID` is set to the `JMSMessageID` of the `forceTerminate` request. Neither properties nor payload are set on the reply message.

Managing worklists

A worklist is a query that is stored persistently in the database. It represents a set of items which have the same characteristics. Although worklist definitions are stored persistently, items contained in the worklist are assembled dynamically when they are queried. All worklists are publicly accessible.

Steps for this task

1. Create a worklist

To create a worklist, save a query with a specific name:

```
process.newWorklist("CustomerOrdersStartingWithA",
    "PROCESS_INSTANCE.NAME, DISTINCT PROCESS_INSTANCE.PIID",
    "PROCESS_INSTANCE.NAME LIKE 'A%'",
    "PROCESS_INSTANCE.NAME",
    null,
    null);
```

This query returns a sorted list of all the process-instance names that begin with the letter A with their associated PIIIDs.

2. Delete a worklist.

```
process.deleteWorklist("CustomerOrdersStartingWithA");
```

Authorization for EJB renderings

Security must be enabled in WebSphere. When an instance of the `BusinessProcess` session bean is created, WebSphere associates a session context with the instance. The session context contains the caller's principal. This is used by both the container and the process engine to check the caller's authorization for each call.

The following work-item assignment reasons are used:

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

These assignment reasons are mapped to authorization authorities:

- Activity reader authority: allowed to see properties of the associated activity instance, and its input and output messages
- Activity editor authority: allowed everything the activity reader allows, and write access to messages and other data associated with the activity
- Potential activity owner authority: allowed everything the activity editor allows, and the right to claim the activity
- Activity owner authority: allowed everything the potential activity owner allows, and the right to complete the activity
- Process starter authority: allowed to see properties of the associated process instance, and its input and output messages, and write other data associated with the process

- Process reader authority: allowed to see properties of the associated process instance, and its input and output messages, and everything the activity reader allows for all contained activities, including those in blocks but not those of the independent subprocesses
- Process administrator authority: allowed everything the process reader and starter allow, and the right to influence the execution of a process, for example, terminating it

Special authority is granted to a person with the role of business process administrator. A business process administrator is a special role; it is different from the process administrator of a process instance. A business process administrator has all privileges.

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

no unique ID for: <user ID>

Required authorizations for process requests

Access to the remote BusinessProcess interface does not guarantee that all functions can be executed; the caller must also be authorized to perform the request. The following minimum authorization authorities are needed for process requests:

Request	Required authorization
createMessage	reader
getActivityInstance	reader
getAllActivities	reader
getAllWorkItems	reader
getCustomAttribute	reader
getEventNames	reader
getFaultMessage	reader
getFaultTerminalNames	reader
getInputMessage	reader
getOutputMessage	reader
getProcessInstance	reader
getVariable	reader
getUISettings	reader
setCustomAttribute	starter
delete	process administrator
forceTerminate	process administrator

Required authorizations for activity requests

Access to the remote BusinessProcess interface does not guarantee that all functions can be executed; the caller must also be authorized to perform the request. The following minimum authorization authorities are needed for activity requests:

Request	Required authorization
createMessage	activity reader or process reader
getActivityInstance	activity reader or process reader
getCustomAttribute	activity reader or process reader
getFaultMessage	activity reader or process reader
getFaultTerminalNames	activity reader or process reader
getInputMessage	activity reader or process reader
getOutputMessage	activity reader or process reader
getOutputTerminalNames	activity reader or process reader
getUserInput	activity reader or process reader
getUISettings	activity reader or process reader
setCustomAttribute	activity editor or process administrator
setOutputMessage	activity editor or process administrator
setFaultMessage	activity editor or process administrator
setUserInput	activity editor or process administrator
claim	potential activity owner or process administrator
sendEvent	potential activity owner or process administrator
cancelClaim	activity owner or process administrator
complete	activity owner or process administrator
forceRetry	process administrator
forceComplete	process administrator

Authorization for JMS renderings

Security must be enabled in WebSphere. When an instance of the BusinessProcess MDB is deployed, the role 'MDBUser' must be mapped to a specific user ID. This user ID is used by both the business process container and the process engine to check the caller's authorization for each request.

The following authorization authorities are needed:

Request	Required authorization
forceTerminate	process administrator
sendEvent	potential activity owner or process administrator

Special authority is granted to a person with the role of business process administrator. A business process administrator is a special role; it is different from the process administrator of a process instance. A business process administrator has all privileges.

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

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

Structure of a Process Choreographer JMS message

A JMS message consists of:

- A message header for message identification and routing information.
- Properties for JMS-specific data, application-specific data, and provider-specific data (optional).
- The body (payload) of the message that holds the content (optional).

Process Choreographer supports text, object, and bytes-message formats.

Message header

The following fields can be set by a Process Choreographer client application:

- **JMSReplyTo**
The destination where a reply to the request should be sent. If this is not specified, a reply is not returned.
- **JMSMessageID**
Uniquely identifies a message. This is set by Process Choreographer when the message sent returns. This is used as the **JMSCorrelationID** in the reply message.
- **JMSCorrelationID**
Links messages. Do not set this field. A Process Choreographer reply message contains the **JMSMessageID** of the request message.

Data properties

This data is passed as name-value pairs. Process Choreographer adds the following properties:

- **wf\$verb**
The function to be called. Possible values are: **initiate**, **call**, **forceTerminate**, **sendEvent**.
If **wf\$verb** is not set and **JMSReplyTo** is set, then **call** is issued. If neither **wf\$verb** nor **JMSReplyTo** are set, **initiate** is issued.
- **wf\$processTemplateName**
The name of the process template to be instantiated. This must be set for **call** and **initiate**. If the template name is not set, **Dispatch** is assumed.
- **wf\$piid**
The object ID of the process instance. It can be used with **forceTerminate** and **sendEvent** to identify the process instance. It is set as the result of **initiate** if **JMSReplyTo** is set. If both **wf\$piid** and **wf\$processInstanceName** are set, the value for **wf\$processInstanceName** is used.
- **wf\$processInstanceName**
The name of the process instance. It can be used with **forceTerminate** and **sendEvent** to identify the process instance to be processed. It can be used with **call** and **initiate** to specify the process instance name of the process instance to be created.
- **wf\$eventName**
The name of the event to be sent. This must be set for **sendEvent**.
- **wf\$processState**
The state of the process instance. This is set in reply messages.
- **wf\$exceptionText**

Specifies the message text of the exception that ended the process.

Message payload

Used to specify input, output, and fault messages.

- Request messages
 - `TextMessage` must contain an object of the type `String`. This object contains the input message to be passed. It requires that the defined input message is also of type `String`.
 - `ObjectMessage` must contain an object of the type `ClientObjectWrapper`. This object contains the input message to be passed.
 - `BytesMessage` must contain a byte array that represents the streamed version of the input message to be passed.
- Reply messages
 - If possible, the reply message is the same format as the request message. That is, if the request message is a `TextMessage`, then the reply message is also a `TextMessage`. If a `TextMessage` cannot be used, for example, because the type of the process instance output message is not a `String`, then an `ObjectMessage` is returned.
 - If an `ObjectMessage` is returned, it contains an object of the type `ClientObjectWrapper`. This object contains the output message or the message of the fault terminal.
 - If a `BytesMessage` is returned, it contains the streamed output message or the streamed message of the fault terminal.
 - If a process can be navigated until its output or fault terminal is reached, the properties `wf$processInstanceName` and `wf$processState` are set.
 - If an exception occurs during the processing of a request or if an exception occurs during the execution of a process instance and the fault is not connected to a fault terminal of the process, the reply message does not contain a payload. The properties `wf$processInstanceName`, `wf$processState`, and `wf$exceptionText` are set. `wf$exceptionText` contains the message text of the exception that ended the process. Nested exceptions follow the message text. These are separated by new-line characters.

Queries on business-process objects

You can use the process query interface to retrieve business-process information that is stored persistently. You use SQL-like syntax to query the following objects:

- Process templates
- Process instances
- Activity instances
- Work items

The query function is provided by the `BusinessProcess` session bean's remote interface. For process templates, the query function has the following syntax:

```
queryProcessTemplates (java.lang.String whereClause,  
                       java.lang.String orderByClause,  
                       java.lang.Integer threshold,  
                       java.util.Timezone timezone);
```

For the other business-process objects, the query function has the following syntax:

```
QueryResultSet query (java.lang.String selectClause,
                    java.lang.String whereClause,
                    java.lang.String orderByClause,
                    java.lang.Integer threshold,
                    java.util.Timezone timezone);
```

The query is made up of:

- Select clause
- Where clause
- Order-by clause
- Threshold parameter
- Timezone parameter

For example, a list of work items accessible to the caller of the function is retrieved by:

```
QueryResultSet result = process.query("WORK_ITEM.WIID",
                                    null, null,
                                    null, null);
```

The query function returns objects according to the caller's authorization. The query result set contains only those objects that the caller is authorized to see.

For more information, see the Javadoc for Process Choreographer in the `com/ibm/bpe/api` package.

Predefined views for queries on business process objects

Process Choreographer provides the following predefined views for queries on business process objects:

- PROCESS_TEMPLATE
- PROCESS_INSTANCE
- PROCESS_ATTRIBUTE
- ACTIVITY
- ACTIVITY_ATTRIBUTE
- EVENT
- WORK_ITEM

PROCESS_TEMPLATE view

Column name	Type	Comments
PTID	ID	Process template ID.
NAME	String	Name of the process template.
APPLICATION_NAME	String	Name of the enterprise application to which the process template belongs.
VALID_FROM	Timestamp	The time from when the process template can be instantiated.
VERSION	String	User-defined version.

Column name	Type	Comments
CREATED	Timestamp	The time the process template is created in the database.
STATE	Integer	Specifies whether the process template is available for process instances to be created. Possible values: STATE_STARTED STATE_STOPPED
DESCRIPTION	String	Description of the process template.
CATEGORY	String	The category to which the process template belongs.
CAN_RUN_SYNC	Boolean	Specifies if the process can run non-interrupted.
CAN_RUN_INTERRUPT	Boolean	Specifies if the process can run interrupted.

PROCESS_ATTRIBUTE view

Column name	Type	Comments
PIID	ID	The ID of the process instance that has a customer attribute.
NAME	String	Name of the customer attribute.
VALUE	String	Value of the customer attribute.

PROCESS_INSTANCE view

Column name	Type	Comments
PTID	ID	Process template ID.
PIID	ID	Process instance ID.
NAME	String	Name of the process instance.
STATE	Integer	State of the process instance. Possible values: STATE_READY STATE_RUNNING STATE_FINISHED STATE_COMPENSATING STATE_FAILED STATE_TERMINATED STATE_COMPENSATED STATE_TERMINATING STATE_FAILING
CREATED	Timestamp	The time the process instance is created.
STARTED	Timestamp	The time the process instance is started.

Column name	Type	Comments
COMPLETED	Timestamp	The time the process instance is completed.
PARENT_NAME	String	The name of the parent process instance.
TOP_LEVEL_NAME	String	The name of the top-level process instance. This is the current process instance if there is no top level.
STARTER	String	Principal ID of the starter.
TEMPLATE_NAME	String	The name of the associated process template.
TEMPLATE_DESCR	String	Description of the associated process template.
TEMPLATE_CATEGORY	String	The category to which the associated process template belongs.

ACTIVITY view

Column name	Type	Comments
PIID	ID	Process instance ID.
AIID	ID	Activity instance ID.
PTID	ID	Process template ID.
ATID	ID	Activity template ID.
KIND	Integer	Kind of activity. Possible values: KIND_PROCESS_SUBFLOW KIND_PROCESS_BLOCK KIND_EMPTY KIND_SINK KIND_SOURCE KIND_ELEMENTAL KIND_FAULT KIND_PERSON KIND_EVENT
RUN_MODE	Integer	Possible values: RUN_MODE_SYNCHRONOUS RUN_MODE_INTERRUPTIBLE RUN_MODE_ATOMIC_SPHERE RUN_MODE_CHAINED
ACTIVATED	Timestamp	The time the activity is activated.
STARTED	Timestamp	The time the activity is started.
COMPLETED	Timestamp	The time the activity is completed.

Column name	Type	Comments
STATE	String	State of the activity. Possible values: STATE_INACTIVE STATE_READY STATE_RUNNING STATE_SKIPPED STATE_FINISHED STATE_FAILED STATE_TERMINATED STATE_CLAIMED STATE_TERMINATING STATE_FAILING STATE_WAITING STATE_EXPIRED STATE_STOPPED
OWNER	String	Principal ID of the owner.
TEMPLATE_NAME	String	Name of the associated activity template.
TEMPLATE_DESCR	String	Description of the associated activity template.

ACTIVITY_ATTRIBUTE view

Column name	Type	Comments
AIID	ID	The ID of the activity instance that has a customer attribute.
NAME	String	Name of the customer attribute.
VALUE	String	Value of the customer attribute.

EVENT view

Column name	Type	Comments
EIID	ID	The ID of the awaited event.
AIID	ID	The ID of activity waiting for the event.
PIID	ID	The ID of the process instance that contains the event.
NAME	String	Name of the event.

WORK_ITEM view

Column name	Type	Comments
WIID	ID	Work item ID.
OWNER_ID	String	Principal ID of the owner.
EVERYBODY	Boolean	Flag indicating whether everybody owns this work item.

Column name	Type	Comments
OBJECT_TYPE	Integer	Type of the associated object. Possible values: OBJECT_TYPE_ACTIVITY OBJECT_TYPE_PROCESS_INSTANCE OBJECT_TYPE_EVENT
OBJECT_ID	ID	ID of the associated object, for example, the associated process or activity.
ASSOC_OBJECT_TYPE	Integer	Type of the object associated with, or containing, the work item's associated object. Possible values: OBJECT_TYPE_ACTIVITY OBJECT_TYPE_PROCESS_INSTANCE OBJECT_TYPE_EVENT
ASSOC_OID	ID	ID of the object associated to or containing the work item's associated object. For example, the PIIID of the process instance containing the activity instance for which this work item has been created.
REASON	Integer	The reason the work item was assigned. Possible values: REASON_POTENTIAL_OWNER REASON_EDITOR REASON_READER REASON_OWNER REASON_POTENTIAL_STARTER REASON_STARTER REASON_ADMINISTRATOR

Select clause

The select clause describes the query result. It specifies a list of names that identify the object properties (columns of the result) to be returned. Its syntax is the same as an SQL select clause; use commas to separate parts of the clause. Each part of the clause must specify a property from one of the predefined views. The columns returned in the QueryResultSet appear in the same order as the properties specified in the select clause.

Note: The select clause does not support SQL aggregation functions, such as AVG(), SUM(), MIN(), MAX(), or COUNT().

Example select clauses

- "WORK_ITEM.OBJECT_TYPE, WORK_ITEM.REASON"
Gets the object types of the associated objects and the assignment reasons for the work items.
- "DISTINCT WORK_ITEM.OBJECT_ID"
Gets all object IDs of objects for which the caller has a work item without duplicates.

- "ACTIVITY.TEMPLATE_NAME, WORK_ITEM.REASON"
Gets the names of the activities the caller has work items for and their assignment reason.
- "ACTIVITY.STATE, PROCESS_INSTANCE.STARTER"
Gets the states of the activities and the starters of their associated process instances.

If an error occurs during the processing of the select clause, a QueryUnknownTable or QueryUnknownColumn exception is thrown with the name of the property that is not recognized as a table or column name.

Where clause

The where clause describes the filter criteria that are to be applied to the query domain. Its syntax is the same as an SQL where-clause. If you do not want to filter a query, you must specify **null** for the where clause.

The where-clause syntax supports:

- Keywords: AND, OR, NOT
- Comparison operators: =, <=, <, <>, >, >=, LIKE
- Set operation: IN

The LIKE operation supports the wildcard characters defined for the queried database.

The following rules also apply:

- Specify object ID constants as ID('string-rep-of-oid').
- Specify timestamp constants as TS('yyyy-mm-ddThh:mm:ss'). To refer to the current date, specify CURRENT_DATE as the timestamp.
You must specify at least a date or a time value in the timestamp:
 - If you specify a date only, the time value is set to zero.
 - If you specify a time only, the date is set to the current date.
 - If you specify a date, the year must consist of four digits; the month and day values are optional. Missing month and day values are set to 01. For example, TS('2003') is the same as TS('2003-01-01T00:00:00').
 - If you specify a time, time values are expressed in the 24-hour system. For example, if the current date is 1 January 2003, TS('T16:04') or TS('16:04') is the same as TS('2003-01-01T16:04:00').
- Specify binary constants as BIN('UTF-8 string')
- It is recommended that you use symbolic constants instead of integer enumerations. For example, instead of specifying an activity state expression "ACTIVITY.STATE=2", specify "ACTIVITY.STATE=ACTIVITY.STATE.STATE_READY".

Examples of where clauses

- Comparing an object ID with an existing ID
`"WORK_ITEM.WIID = ID('_WI:800c00ed.df8d7e7c.feffff80.38')"`

This type of where clause is usually created dynamically with an existing object ID from a previous call. If this object ID is stored in a variable `wiid1`, the clause could be constructed as:

```
"WORK_ITEM.WIID = ID('" + wiid1.toString() + "')
```

- Using timestamps

```
"ACTIVITY.STARTED >= TS('2002-06-1T16.00.00')"
```

- Using symbolic constants
"WORK_ITEM.REASON = WORK_ITEM.REASON.REASON_OWNER"
- Using boolean values true and false
"PROCESS_TEMPLATE.CAN_RUN_SYNC = TRUE"

Order-by clause

Use the order-by clause to specify the sort criteria for the query result set. The order-by clause syntax is the same as an SQL order-by clause; use commas to separate each part of the clause. Each part of the clause must specify a property from one of the predefined views.

If you identify more than one property, the query result set is ordered by the values of the first property, then by the values of the second property, and so on.

If you do not want to sort the query result set, you must specify **null** for the order-by clause.

Examples of order-by clauses

- "PROCESS_TEMPLATE.NAME"
Sorts the query result alphabetically by the process-template name.
- "PROCESS_INSTANCE.CREATED, PROCESS_INSTANCE.NAME DESC"
Sorts the query result by the creation date and, for a specific date, sorts the results alphabetically by the process-instance name in reverse order.
- "ACTIVITY.OWNER, ACTIVITY_TEMPLATE.NAME, ACTIVITY.STATE"
Sorts the query result by the activity owner, then the activity-template name, and then the state of the activity.

Threshold parameter

The threshold parameter in the query function restricts the number of objects returned in the query result set. This can be useful, for example, in a GUI where only a small number of items should be displayed. If you set the threshold parameter accordingly, it improves the performance; the database query is faster and less data needs to be transferred from the server to the client.

If the parameter is set to null, a threshold is not applied and all the qualifying objects are returned.

Example of a threshold parameter

- `new Integer(50)`
Specifies that only 50 qualifying objects are to be returned.

Timezone parameter

Timezones can differ between the client that starts the query and the process engine that processes the query. Use the timezone parameter to specify the timezone of the timestamp constants used in the where clause, for example, to specify local times. The dates returned in the query result set have the same timezone as that specified in the query.

If the parameter is set to null, the timestamp constants are assumed to be UTC times.

Examples of timezone parameters

- ```
process.query("ACTIVITY.AIID",
 "ACTIVITY.STARTED > TS ('2002-01-01T17:40')",
 null,
 null,
 java.util.Timezone.getDefault());
```

Specifies that object IDs are to be returned for activities that have been started later than 17:40 local time on 1 January 2002.

- ```
process.query("ACTIVITY.AIID",  
             "ACTIVITY.STARTED > TS ('2002-01-01T17:40')",  
             null,  
             null,  
             null);
```

Specifies that object IDs are to be returned for activities that have been started later than 17:40 UTC on 1 January 2002. This is, for example, 6 hours earlier in eastern standard time.

Query results

A query result set contains the results of a query. The elements of the set are objects that the caller is authorized to see. Elements can be read in a relative fashion using the **next()** method or in an absolute fashion using the **first()** and **last()** methods. Because the implicit cursor of a query result set is initially positioned before the first element, you must call either **first()** or **next()** before reading an element. You can use the **size()** method to determine the number of elements in the set.

An element of the query result set comprises the selected attributes of work items and their associated referenced objects, such as activity instances and process instances. The first attribute (column) of a `QueryResultSet` element specifies the value of the first attribute specified in the select clause of the query request. The second attribute (column) of a `QueryResultSet` element specifies the value of the second attribute specified in the select clause of the query request, and so on.

You can retrieve the values of the attributes by calling a method that is compatible with the attribute type and by specifying the appropriate column index.

Note: The numbering of the column indexes starts with 1.

Attribute type	Method
String	<code>getString</code>
ID	<code>getOID</code>
Timestamp	<code>getTimestamp</code> <code>getString</code>
Integer	<code>getInteger</code> <code>getShort</code> <code>getLong</code> <code>getString</code> <code>getBoolean</code>
Boolean	<code>getBoolean</code> <code>getShort</code> <code>getInteger</code> <code>getLong</code> <code>getString</code>

Example

The following query is run:

```
QueryResultSet resultSet = process.query("ACTIVITY.STARTED, ACTIVITY.TEMPLATE_NAME AS NAME,  
    WORK_ITEM.WIID, WORK_ITEM.REASON",  
    null,  
    null,  
    null,  
    null);
```

The returned query result set has four columns:

- Column 1 is a timestamp
- Column 2 is a string
- Column 3 is an object ID
- Column 4 is an integer

You can use the following methods to retrieve the attribute values:

```
while (resultSet.next())  
{  
    java.util.Calendar activityStarted = resultSet.getTimestamp(1);  
    String templateName = resultSet.getString(2);  
    WIID wiid = (WIID) resultSet.getOID(3);  
    Integer reason = resultSet.getInteger(4);  
}
```

You can use the display names of the result set, for example, as headings for a printed table. These are the column names of the view or the name defined by the **AS** clause. You can use the following methods to retrieve the display names in the example:

```
resultSet.getColumnDisplayName(1) returns "STARTED"  
resultSet.getColumnDisplayName(2) returns "NAME"  
resultSet.getColumnDisplayName(3) returns "WIID"  
resultSet.getColumnDisplayName(4) returns "REASON"
```

Chapter 8. Configuring the staff service for Process Choreographer

Process Choreographer uses staff plug-ins to determine who can start a process or claim an activity. Your business processes can also use the staff plug-in services to resolve staff queries. Each type of directory service requires a different staff plug-in. You can register multiple staff plug-ins. The User Registry and System plug-ins are already installed and can be used without any configuration. To configure a staff plug-in provider:

Steps for this task

1. In the Administrative Console, click **Resources > Staff Plugin Provider**.

The System plug-in and the User Registry plug-in require no customization and are ready to use. The preconfigured LDAP staff plugin configuration assumes that the LDAP server is on the same host as the Enterprise Application Server.

2. To create a new LDAP configuration:

- a. Click on the name of the LDAP staff plug-in provider.

- b. Select **Staff Plugin Configuration**.

- c. Click **New**.

- d. Click **Browse**, and select the sample XSL transformation file to be used.

The standard XSL transformation for LDAP is located on UNIX in `$WAS_HOME/ProcessChoreographer/Staff/LDAPTransformation.xsl` and on Windows, in

`%WAS_HOME%\ProcessChoreographer\Staff\LDAPTransformation.xsl`.

Note: Do not modify this transformation file. If you need to customize the transformations to match your organization's LDAP schema, modify a copy that has a different file name.

- e. Click **Next**.

- f. Enter an administrative name for the staff plug-in provider.

- g. Enter a description.

- h. Enter the JNDI name that will be used by business processes to reference this plug-in, for example, `bpe/staff/ldapsrvr1`

- i. Click **Apply**.

- j. Click **Custom Properties**.

- k. For each of the required properties and for any optional properties that you want to set, click on the property's name, enter a value, and click **OK**.

This table describes each property for the LDAP plug-in.

LDAP plug-in property	Required or optional	Comments
-----------------------	----------------------	----------

AuthenticationAlias	Optional	The authentication alias used to connect to LDAP, for example, mycomputer/My LDAP Alias. You must have defined this alias in the Administrative console via Security > JAAS Configuration > J2C Authentication Data. If this is not set, anonymous logon to the LDAP server is used.
AuthenticationType	Optional	If AuthenticationType is not set, the default logon is anonymous authentication. In all other cases, the default is simple authentication.
BaseDN	Required	The base distinguished name for all LDAP search operations, for example, "o=mycompany, c=us"
ContextFactory	Required	Sets the JNDI context factory, for example, "com.sun.jndi.ldap.LdapCtxFactory"
ProviderURL	Required	This URL must point to the LDAP/JNDI directory server and port. The format must be in normal JNDI syntax, for example, "ldap://localhost:389"
SearchScope	Required	The default search scope for all search operations. Determines how deep to search beneath the baseDN. It must be one of: "objectScope", "oneLevelScope", or "subtreeScope"
additionalParameterName1-5 & additionalParameterValue1-5	Optional	These Name-Value pairs can be used to set up to five arbitrary JNDI properties for the connection to the LDAP server.

- l. To apply the changes you have made, click **Save** in the Message(s) box.
3. To activate the plug-in, stop and start the server.
4. In case of problems, refer to Troubleshooting the staff service and staff plug-ins.

Results

Processes can now use the staff support services to resolve staff queries, and to determine which activities can be performed by which people.

What to do next

Depending on the queries you want to create and your directory structure, you might need to create your own transformations. For more information about this, see "About the staff service in Process Choreographer" on page 139.

About the staff service in Process Choreographer

Process Choreographer allows you to separate your business process logic from the staff resolution. Staff queries are resolved using a plug-in that is specific to the directory service. Some examples of using the staff service are described here:

- Staff queries using the staff support service
- Staff query verb set
- Explicit staff assignments using the System staff plug-in
- Staff queries transformed for the User Registry staff plug-in
- Staff queries transformed for the LDAP staff plug-in

For detailed information on the staff resolution plug-ins, see the staff resolution white papers in WebSphere Developer Domain at <http://www7b.software.ibm.com/wsdd/zones/was/wpc.html>.

Staff queries using the staff support service

You can use WebSphere Studio Application Developer Integration Edition to define staff queries for the staff support service. Staff queries are templates that define how to retrieve the list of users authorized for a certain work item. You can use the abstract query templates (staff verbs) in the WebSphere Studio Application Developer Integration Edition Process Editor when you model a business process. These staff verbs are transformed during modeling and deployment into a set of queries that can be executed at run time against a staff repository.

Before the initial staff query verbs in the Process Editor and the parameterized verbs in the process models can be executed as queries against a specific staff repository, they must be translated into executable queries using an XSL transformation. The result of a transformation (mapping) can be executed by staff resolution plug-ins that provide access to specific directories, such as LDAP or the User Registry. At run time, the plug-in executes the query by invoking the staff repository's APIs and creating the list of authorized user IDs for the corresponding work item.

During process deployment, the staff support service is invoked to deploy the staff query. It retrieves the staff plug-in provider configuration with the corresponding JNDI name, and on behalf of the XSLT verb mapping file and the staff resolution plug-in defined in the configuration, the staff support service converts the parameterized staff query verb into a query that can be executed by a specific staff resolution plug-in. All staff query verbs belonging to a process template must use the same provider configuration.

The following example illustrates an FDML snippet as generated by the Process Editor to retrieve the manager of the employee that started the process:

```
<wf:staff type="staffSupportService">
  <staff:verb>
    <staff:id>Manager of Employee by user ID</staff:id>
    <staff:name>Manager of Employee by user ID</staff:name>
    <staff:parameter id="EmployeeUserID">
      %wf:process.starter%
    </staff:parameter>
  </staff:verb>
</wf:staff>
```

Staff query verb set

The staff support service accepts queries in an abstract form that is independent of the directory infrastructure used. The Process Editor has a set of predefined staff query verbs delivered that you can use as-is. The individual staff resolution plug-ins and the XSLT mapping files do not support all the verbs. The *Manager of Employee* verb, for example, is not available if you use the User Registry or the System plug-in.

The staff query verbs are contained in the file VerbSet.xml. This file is located:

- On UNIX in \$WAS_HOME/ProcessChoreographer/Staff/
- On Windows in %WAS_HOME%\ProcessChoreographer\Staff\

You can modify this set of staff query verbs. Make your changes to a copy that has a different file name.

The following predefined set of verbs is available. For information on the parameters that you can use with each of the verbs, see “Predefined staff verbs and their parameters” on page 142.

Department Members

This verb allows you to define a query to retrieve the members of a department. The retrieved users will belong to any of the specified departments (DepartmentName, AlternativeDepartmentName1, or AlternativeDepartmentName2). This verb is supported by the LDAP plug-in.

Everybody

This verb allows you to assign a work item to every user authenticated by the WebSphere Application Server. This verb is supported by the System, User Registry, and LDAP plug-ins.

Group Members

This verb allows you to define a query to retrieve the members of a group. The retrieved users will belong to any of the specified groups (GroupName, AlternativeGroupName1, or AlternativeGroupName2). This verb is supported by the User Registry and LDAP plug-ins.

Group Search

This verb allows you to search for a group based on an attribute match and to retrieve the members of the group. This verb is supported by the User Registry and LDAP plug-ins.

Manager of Employee

Retrieves the manager of a person using the person’s name. This verb is supported by the LDAP plug-in. You might need to customize the default mapping XSLT file to match your organization’s LDAP schema.

Manager of Employee by user ID

Retrieves the manager of a person using the person’s user ID. It is useful in combination with context queries. This verb is supported by the LDAP plug-in. You might need to customize the default mapping XSLT file to match your organization’s LDAP schema.

Native Query

This verb allows you to define a native query based on directory-specific parameters. This verb is supported by the User Registry and LDAP plug-ins. You might need to customize the default mapping XSLT file to match your organization’s LDAP schema.

Nobody

This verb denies normal users access to the work item; only the process administrator and the Process Choreographer system administrator have access to it. This verb is supported by the System, User Registry, and LDAP plug-ins.

Person Search

Allows you to search for a person based on an attribute match. This verb is supported by the User Registry and LDAP plug-ins. You might need to customize the default mapping XSLT file to match your organization's LDAP schema.

Role Members

Retrieves the users associated with a business process role. The retrieved users will belong to any of the specified roles (RoleName, AlternativeRoleName1, or AlternativeRoleName2). This verb is supported by the LDAP plug-in. You might need to customize the default mapping XSLT file to match your organization's LDAP schema.

Users This verb allows you to define a staff query for a user who is known by name. It is not recommended that you hard code user names in process templates. This verb is useful for testing processes. This verb is supported by the System, User Registry, and LDAP plug-ins. You might need to customize the default mapping XSLT file to match your organization's LDAP schema.

Users by user ID

This verb allows you to define a staff query for a user whose user ID is known. Even though it is not recommended that you hard code user IDs in process templates, this verb is useful in combination with context queries, for example:

```
User [username='%wf:process.starter%']
```

This verb is useful for testing processes. This verb is supported by the System, User Registry, and LDAP plug-ins. You might need to customize the default mapping XSLT file to match your organization's LDAP schema.

Explicit staff assignments using the System staff plug-in

This plug-in requires no configuration parameters, and comes pre-installed and ready-to-use. It allows you to hard code user names in your business process. This is not normally recommended, but it can be useful for testing if you are using context variables, or special queries, as illustrated by the following examples:

```
<sur:staffQueries>
  <staff:userID name="%wf:process.starter%">
  <staff:userID name="%wf:process.administrators%">
</sur:staffQueries>
<sur:staffQueries>
  <staff:everybody>
</sur:staffQueries>
<sur:staffQueries>
  <staff:nobody>
</sur:staffQueries>
```

The following XML snippet explicitly retrieves the WebSphere user ID smith:

```
<staff:staffQueries>
  <staff:userID name="smith">
</staff:staffQueries>
```

Staff queries transformed for the User Registry staff plug-in

This plug-in allows staff queries to refer to users and groups that are known to the WebSphere Application Server User Registry. This plug-in does not require any configuration parameters, and comes pre-installed and ready-to-use.

The following example XML snippet retrieves the user IDs of all members of the Administrators group, and all users whose name begins with 'Mi':

```
<sur:staffQueries>
  <sur:usersOfGroup groupName="Administrators"/>
  <sur:search type="user" name="Mi*"/>
</sur:staffQueries>
```

Note: This XML snippet is an example of the output of the transformation.

Staff queries transformed for the LDAP staff plug-in

If you want to use the LDAP plug-in, you will probably have to create a customized version of the LDAP XSL transformation to match your organization's LDAP schema. The standard XSLT provided for LDAP is located:

- On UNIX in \$WAS_HOME/ProcessChoreographer/Staff/LDAPTransformation.xml
- On Windows in %WAS_HOME%\ProcessChoreographer\Staff\LDAPTransformation.xml

Make your changes to a copy that has a different file name.

The following XML snippet illustrates the results of the LDAP transformation for a search.

```
<sldap:staffQueries>
  <sldap:search baseDN="ou=mydivision, o=mycompany, c=us" filter="cn=*"
    searchScope="onelevelScope" recursive="no">
    <sldap:attribute name="uid" objectclass="person" usage="simple"/>
  </sldap:search>
</sldap:staffQueries>
```

Predefined staff verbs and their parameters

You can use staff verbs in the WebSphere Studio Application Developer Integration Edition Process Editor to model staff assignments in a business process. These staff verbs are transformed during modeling and deployment into a set of queries that can be executed at run time against a staff repository. The parameters for the following predefined staff verbs are listed here:

- Department Members
- Everybody
- Group Members
- Group Search
- Manager of Employee
- Manager of Employee by user ID
- Native Query
- Nobody
- Person Search
- Role Members
- Users
- Users by user ID

Department Members

This verb allows you to define a query to retrieve the members of a department.

Parameter	Use	Type	Supported by	Description
DepartmentName	Mandatory	string	LDAP	Department name of the users to be retrieved.
IncludeNestedDepartments	Mandatory	boolean	LDAP	Specifies whether nested departments are considered in the query.
Domain	Optional	string	LDAP	The domain to which the department belongs. Use this parameter to limit the query to a subset of the directory.
AlternativeDepartmentName1	Optional	string	LDAP	An alternative department to which the users can belong.
AlternativeDepartmentName2	Optional	string	LDAP	An alternative department to which the users can belong.

Everybody

This verb allows you to assign a work item to every user authenticated by the WebSphere Application Server. It has no parameters. It is supported by the System, User Registry, and LDAP plug-ins.

Group Members

This verb allows you to define a query to retrieve the members of a group.

Parameter	Use	Type	Supported by	Description
GroupName	Mandatory	string	User Registry, LDAP	Group name of the users to be retrieved.
IncludeSubgroups	Mandatory	boolean	LDAP	Specifies whether nested subgroups are considered in the query.
Domain	Optional	string		The domain to which the group belongs. Use this parameter to limit the query to a subset of the directory.
AlternativeGroupName1	Optional	string	User Registry, LDAP	An alternative group to which the users can belong.
AlternativeGroupName2	Optional	string	User Registry, LDAP	An alternative group to which the users can belong.

Group Search

This verb allows you to search for a group based on an attribute match and to retrieve the members of the group.

Parameter	Use	Type	Supported by	Description
GroupID	Optional	string	User Registry, LDAP	The group ID of the users to be retrieved.
Type	Optional	string	LDAP	The group type of the users to be retrieved.
IndustryType	Optional	string	LDAP	The industry type of the group to which the users belong.
BusinessType	Optional	string	LDAP	The business type of the group to which the users belong.
GeographicLocation	Optional	string	LDAP	Where the users are located.
Affiliates	Optional	string	LDAP	The affiliates of the users.
DisplayName	Optional	string	LDAP	The display name of the group.
Secretary	Optional	string	LDAP	The secretary of the users.
Assistant	Optional	string	LDAP	The assistant of the users.
Manager	Optional	string	LDAP	The manager of the users.
BusinessCategory	Optional	string	LDAP	The business category of the group to which the users belong.
ParentCompany	Optional	string	LDAP	The parent company of the users.

Manager of Employee

Retrieves the manager of a person using the person's name.

Parameter	Use	Type	Supported by	Description
EmployeeName	Mandatory	string	LDAP	The name of the employee whose manager is to be retrieved.
Domain	Optional	string		The domain to which the employee belongs. Use this parameter to limit the query to a subset of the directory.

Manager of Employee by user ID

Retrieves the manager of a person using the person's user ID.

Parameter	Use	Type	Supported by	Description
EmployeeUserID	Mandatory	string	LDAP	The user ID of the employee whose manager is to be retrieved. Allows context variables, such as %wf:process.starter%
Domain	Optional	string		The domain to which the employee belongs. Use this parameter to limit the query to a subset of the directory.

Native Query

This verb allows you to define a native query based on directory-specific parameters.

Parameter	Use	Type	Supported by	Description
QueryTemplate	Mandatory	string	User Registry, LDAP	The query template to be used for the query. The default mapping files for the User Registry and LDAP plug-ins support the templates <code>search</code> , <code>user</code> , and <code>usersOfGroup</code> .
Query	Mandatory	string	User Registry, LDAP	Specifies the query. You can use context variables, such as <code>%wf:process.starter%</code> . The type of query depends on the plug-in and the query template. User Registry <ul style="list-style-type: none"> • search template: search pattern • user template: user name • usersOfGroup: group name LDAP <ul style="list-style-type: none"> • search template: search filter • user template: user dn • usersOfGroup: group dn
AdditionalParameter1	Optional	string	User Registry, LDAP	Specifies the query. You can use context variables, such as <code>%wf:process.starter%</code> . The type of parameter depends on the plug-in and the query template. User Registry <ul style="list-style-type: none"> • search template: used for the search type. Allowed values are <i>group</i> and <i>user</i>. • user template: not supported • usersOfGroup: not supported LDAP <ul style="list-style-type: none"> • search template: used to specify whether recursive search is done. Allowed values are <i>yes</i> and <i>no</i> • user template: not supported • usersOfGroup: used to specify whether recursive search is done. Allowed values are <i>yes</i> and <i>no</i>

Parameter	Use	Type	Supported by	Description
AdditionalParameter2	Optional	string	User Registry, LDAP	Allows you to specify an additional parameter.
AdditionalParameter3	Optional	string	User Registry, LDAP	Allows you to specify an additional parameter. If you use the default mapping XSLT files, this parameter is not supported.
AdditionalParameter4	Optional	string	User Registry, LDAP	Allows you to specify an additional parameter. If you use the default mapping XSLT files, this parameter is not supported.
AdditionalParameter5	Optional	string	User Registry, LDAP	Allows you to specify an additional parameter. If you use the default mapping XSLT files, this parameter is not supported.

Nobody

This verb denies normal users access to the work item; only the process administrator and the Process Choreographer system administrator have access to it. It has no parameters. It is supported by the System, User Registry, and LDAP plug-ins.

Person Search

Allows you to search for people based on an attribute match.

Parameter	Use	Type	Supported by	Description
UserID	Optional	string	User Registry, LDAP	The user ID of the users to be retrieved.
Profile	Optional	string	LDAP	The profile of the users to be retrieved.
LastName	Optional	string	LDAP	The last name of the users to be retrieved.
FirstName	Optional	string	LDAP	The first name of the users to be retrieved.
MiddleName	Optional	string	LDAP	The middle name of the users to be retrieved.
Email	Optional	string	LDAP	The e-mail address of the users.
Company	Optional	string	LDAP	The company to which the users belong.
DisplayName	Optional	string	LDAP	The display name of the users.
Secretary	Optional	string	LDAP	The secretary of the users.

Parameter	Use	Type	Supported by	Description
Assistant	Optional	string	LDAP	The assistant of the users.
Manager	Optional	string	LDAP	The manager of the users.
Department	Optional	string	LDAP	The department to which the users belong.
Phone	Optional	string	LDAP	The telephone number of the users.
Fax	Optional	string	LDAP	The fax number of the users.
Gender	Optional	string	LDAP	Whether the user is male or female.
Timezone	Optional	string	LDAP	The timezone in which the users are located.
PreferredLanguage	Optional	string	LDAP	The preferred language of the user.

Role Members

Retrieves the users associated with a business process role.

Parameter	Use	Type	Supported by	Description
RoleName	Mandatory	string	LDAP	Role name of the users to be retrieved.
IncludeNestedRoles	Mandatory	boolean	LDAP	Specifies whether nested roles are considered in the query.
Domain	Optional	string		The domain to which the role belongs. Use this parameter to limit the query to a subset of the directory.
AlternativeRoleName1	Optional	string	LDAP	An alternative role name for the user.
AlternativeRoleName2	Optional	string	LDAP	An alternative role name for the user.

Users

This verb allows you to define a staff query for a user who is known by name.

Parameter	Use	Type	Supported by	Description
Name	Mandatory	string	System, User Registry, LDAP	The name of the user to be retrieved.
AlternativeName1	Optional	string	System, User Registry, LDAP	An alternative user name. Use this parameter to retrieve more than one user.
AlternativeName2	Optional	string	System, User Registry, LDAP	An alternative user name. Use this parameter to retrieve more than one user.

Users by user ID

This verb allows you to define a staff query for a user whose user ID is known.

Parameter	Use	Type	Supported by	Description
UserID	Mandatory	string	System, User Registry, LDAP	The user ID of the user to be retrieved.
AlternativeID1	Optional	string	System, User Registry, LDAP	An alternative user ID. Use this parameter to retrieve more than one user.
AlternativeID2	Optional	string	System, User Registry, LDAP	An alternative user ID. Use this parameter to retrieve more than one user.

Troubleshooting the staff service and the staff plug-ins

If you encounter one of the following situations, it might be due to a problem with the staff service or a staff plug-in:

- Stopped staff activities
- Changes to the staff repository are not immediately reflected in work-item assignments

Use this overview task to help resolve the problem. You can also go to the Technical support search page, to look for additional information.

Stopped staff activities

You have encountered one or more of the following problems:

- Work items resulting from staff activities cannot be claimed although the business process started navigating successfully.
- The SystemOut.log file contains the following message: BPEE0057I: Activity 'MyStaffActivity' of processes 'MyProcess' has been stopped because of an unhandled failure...

This indicates that the WebSphere Application Server security might not be enabled. Staff activities **require** that security is enabled and the User Registry is configured. Do the following:

1. Check that WebSphere security is enabled. In the administrative console, go to **Security > Global Security**.
2. Check that the User Registry is configured. In the administrative console, go to **Security > User Registries**.

Changes to the staff repository are not immediately reflected in work-item assignments

For example, you have added the user, Frank, to the staff repository but Frank has not received any work items although he is eligible for them.

This problem can occur when the staff query results for a process template in the cache have expired. To optimize the staff query resolution performance, the retrieved query results are cached and shared for all process instances of a process template if the content of the context variables used in the query does not differ from one query instance to

another. The cache content is checked for currency when a new process instance is created or the corresponding staff activity gets scheduled. By default, the time after which the shared staff query results expire is one hour.

To change the default value, modify the variable `StaffQueryResultValidTimeSeconds` in the `bpe.properties` file. This file is located:

- On UNIX in `$WAS_HOME/properties`
- On Windows in `%WAS_HOME%\properties`

Note:

Note: You might need to create the `bpe.properties` file. For example, to set the expiry time to one minute, change the variable to:

```
StaffQueryResultValidTimeSeconds=60
```

Staff service settings

Use this page to enable or disable the staff service, which manages staff plugin resources used by the server.

To view this administrative console page, click **Servers > Application Servers > *server_name* > Staff Service** .

Startup

Specifies whether the server will attempt to start the staff service.

Default	Selected
Range	Selected When the application server starts, it attempts to start the staff service automatically.
	Cleared The server does not try to start the staff service. If staff plugin resources are to be used on this server, the system administrator must start the staff service manually or select this startup property then restart the server.

Staff plugin provider collection

Use this page to manage staff plugin providers.

A staff plugin is responsible for the retrieval of user information.

To view this administrative console page, click **Resources > Staff Plugin Providers** .

Name

The name by which the staff plugin provider is known for administrative purposes.

Data type	String
-----------	--------

Description

A description of the staff plugin provider.

Data type String

Jar File

The file name, including the absolute path, of the Jar file containing the plugin.

Data type String

Staff plugin provider settings

Use this page to modify staff plugin provider settings.

Staff Plugins are responsible for the retrieval of user information. Each staff plugin provider is registered with the run-time environment by specifying a name and a Jar file containing the plugin. A configuration file in the Jar file defines the class name, which represents the plugin as well as the properties for the plugin.

To view this administrative console page, click **Resources > Staff Plugin Providers > *staffpluginprovider_name***.

Name

The name by which the staff plugin provider is known for administrative purposes.

Data type String

Description

A description of the staff plugin provider.

Data type String

Jar File

The file name, including the absolute path, of the Jar file containing the plugin.

Data type String

Staff plugin configuration collection

Use this page to manage staff plugin configurations.

A staff plugin configuration is defined for a staff plugin provider. The staff plugin configuration can define any custom properties specified by the staff plugin provider. Each staff plugin provider can have multiple staff plugin configurations.

To view this administrative console page, click **Resources > Staff Plugin Providers > *staffpluginprovider_name* > Staff Plugin Configuration**.

Name

The name by which the staff plugin configuration is known for administrative purposes.

Data type String

Description

A description of the staff plugin configuration.

Data type String

JNDI Name

The JNDI name used to look up the staff plugin configuration in the namespace.

Data type String

XSL Transform File

The file name, including absolute path, of the XSL transformation file.

Data type String

Staff plugin configuration settings

Use this page to modify staff plugin configuration settings.

To view this administrative console page, click **Resources > Staff Plugin Providers > *staffpluginprovider_name* > Staff Plugin Configuration > *staffpluginconfiguration_name***.

Name

The name by which the staff plugin configuration is known for administrative purposes.

Data type String

Description

A description of the staff plugin configuration.

Data type String

JNDI Name

The JNDI name used to look up the staff plugin configuration in the namespace.

Data type String

XSL Transform File

The file name, including absolute path, of the XSL transformation file.

Data type String

Chapter 9. Using compensation in service choreography

You can use *compensation* in business processes to enable updates to be committed in several related transactions before the process has completed. If the process does not complete successfully, compensation is used to automatically perform actions that *compensate for* updates that have been committed. The compensation actions can be to back out committed updates or to take alternative actions.

For compensation, application developers can create a type of component service called a *compensation pair*. Each compensation pair contains a *primary operation* that performs the normal actions of the service and a *compensation operation* that can be used for the compensation actions.

You can use WebSphere Studio to define compensation properties of interruptible business processes.

Note: You cannot use compensation in non-interruptible processes.

When a process is started, it starts a new *compensation sphere*. Each compensation pair called as a node of the process registers with the compensation sphere, runs its primary operation, and stores the information needed to run its compensation operation if the process needs to be compensated. When the process ends it closes the compensation sphere, which checks if there is a need for compensation. The process decides whether to accept all of the updates made by primary operations or to run the compensation operation of each compensation pair. The compensation operation can access parts of both the input message and the primary operation's output message.

Chapter 10. Troubleshooting Process Choreographer

Process Choreographer uses the WebSphere framework JRas for traces, messages, and audit logs. The following topics contain troubleshooting information that applies for Process Choreographer only.

Steps for this task

1. Using process-related trace information
2. Using process-related audit trail information
3. Using process-related messages
4. Troubleshooting the business process container

Using process-related trace information

Process Choreographer uses the WebSphere framework JRas for traces. How to use traces in WebSphere is described in Working with trace.

The following is process-specific information that is relevant for Process Choreographer:

- The Java package for Process Choreographer is **com.ibm.bpe**. Use `com.ibm.bpe` to trace process-related events only.
- The trace information is stored in the trace file that corresponds to the Java package you are using.

Using process-related audit trail information

When a process instance is executed and audit trailing is enabled, Process Choreographer writes information about each significant event into an audit log, which is located in the corresponding database table. Process Choreographer provides a plug-in for the audit trail database to be used. In addition, you can clean up the audit log table according to your needs by using the cleanup utility.

- “BPEAuditLogDelete utility for Process Choreographer” describes how to start the utility and lists the options that you can use.
- “Process Choreographer - structure of the audit trail database table” on page 156 describes the structure of the audit trail database table.
- “Process Choreographer - audit event types” on page 158 contains a list and description of all available event type codes that can occur during the processing of processes.

BPEAuditLogDelete utility for Process Choreographer

Syntax

You can use this utility to delete audit log entries for Process Choreographer from the database.

To run the utility, change to the `\bin` subdirectory of the WebSphere application server, and enter the command, using the following syntax:

```
BPEAuditLogDelete {-processtime processtime | -time time | -all [slice]}
  -dbs database_system -dbn database_name
  [-u userId] [-p password]
  [-host host] [-port port]
```

Parameters

Supported arguments include:

-processtime

Deletes all audit log entries that belong to a process that has finished before the time specified in *processtime*. Use the same time format as for the -time option.

-time Deletes all audit log entries that are older than the time you specify in *time*. The time format must be as follows: YYYY-MM-DD[‘T’HH:MM:SS]. If you only specify year, month, and day, the hour, minutes, and seconds are set to 00:00:00.

-all Deletes all audit log entries in the database. This is done in multiple transactions where each transaction deletes the specified amount of entries as specified in *slice*. The default size for *slice* is 250.

-dbs Specifies the database system. Allowed database systems and keywords are DB2, Cloudscape, Oracle, and Sybase.

-dbn Specifies the name of the database that contains the audit log table.

-u Specifies the user that is used to connect to the database. This option is not required for Cloudscape and DB2.

-p Specifies the password for the user you specified with the -u option. You can also enter the password after starting the utility.

-host Specifies the name of the database host. This option is only required for Oracle and Sybase, it is not supported for DB2 and Cloudscape.

-port Specifies the port of the database listener on the host. This option is only required for Oracle and Sybase, it is not supported for DB2 and Cloudscape.

Process Choreographer - structure of the audit trail database table

The following table describes the structure of the audit trail database table AUDIT_LOG_T. It lists the names of the columns, the audit event types, and gives you a short description for the table column.

To read the content of the audit trail, use SQL or any other administration tool that allows you to read database tables.

There are the following audit event types:

- Process instance events (PIE)
- Activity instance events (AIE)
- Events related to variables (VAR)
- Control link events (CLE)
- Process template events (PTE)

For a list of the possible audit event type codes, refer to “Process Choreographer - audit event types” on page 158.

**Structure of the Process Choreographer audit trail database table
AUDIT_LOG_T**

Name	Audit event type	Description
ALID	all	Internal ID and primary key for a row.
EVENT_TIME	all	Timestamp of when the event occurred (in UTC format).
AUDIT_EVENT	all	For a list of audit event codes, refer to "Using process-related audit trail information" on page 155.
PTID	all	Process template ID of the process that is related to the current event.
PIID	PIE, AIE, VAR, CLE	Process Instance ID of the process instance that is related to the current event.
PROCESS_TEMPL_NAME	PIE, AIE, VAR, CLE	Process template name of the process template that is related to the current event.
PROCESS_INST_NAME	PIE, AIE, VAR, CLE	Process instance name of the process template that is related to the current event.
TOP_LEVEL_PI_NAME	PIE, AIE, VAR, CLE	Name of the top-level process that is related to the current event.
TOP_LEVEL_PIID	PIE, AIE, VAR, CLE	Identifier of the top-level process that is related to the current event.
PARENT_PI_NAME	PIE, AIE, VAR, CLE	Name of the parent process instance.
PARENT_PIID	PIE, AIE, VAR, CLE	Process instance ID of the parent process, or null if no parent exists.
VALID_FROM	PIE, AIE, VAR, CLE	Valid-from date of the process template that is related to the current event.
ACTIVITY_NAME	AIE	Name of the activity on which the event occurred.
ACTIVITY_KIND	AIE	Kind of the activity on which the activity occurred. It can have the following values: KIND_PROCESS_SUBPROCESS KIND_PROCESS_BLOCK KIND_EMPTY KIND_ELEMENTAL KIND_PERSON KIND_EVENT
ACTIVITY_STATE	AIE	State of the activity that is related to the event.
CONTROL_LINK_NAME	CLE	Name of the control link that is related to the current control link event.

IMPL_NAME	AIE	Name of the activity implementation. This is only applicable for elemental activities.
PRINCIPAL	AIE, PIE	Name of the principal that requested the API-call related to the event, applicable for these events (but only if directly called over the API): ACTIVITY_CLAIMED ACTIVITY_COMPLETED ACTIVITY_FAILED ACTIVITY_FAULT_MESSAGE_SET ACTIVITY_OUTPUT_MESSAGE_SET ACTIVITY_USERINPUT_SET PROCESS_DELETED PROCESS_STARTED PROCESS_STARTED PROCESS_TERMINATED
TERMINAL_NAME	AIE	Name of the fault terminal that has been set, applicable for SET_FAULT_MESSAGE.
VARIABLE_DATA	VAR	Data for variables for VARIABLE_UPDATED events.
EXCEPTION_TEXT	PIE, AIE	Exception message that caused an activity or process to fail. Applicable for: - PROCESS_FAILED - ACTIVITY_FAILED.
DESCRIPTION	PIE, AIE	Description of activity or process, containing potentially resolved replacement variables.

Process Choreographer - audit event types

The following tables list the types of audit events that can occur while processes are running, together with their corresponding codes. The ones in brackets are not implemented in this version.

Process instance events (PIE)

PROCESS_STARTED	21000
(PROCESS_SUSPENDED)	21001
(PROCESS_RESUMED)	21002
PROCESS_COMPLETED	21004
PROCESS_TERMINATED	21005
PROCESS_DELETED	21020
PROCESS_FAILED	42001
(PROCESS_COMPENSATING)	42003
PROCESS_COMPENSATED	42004
PROCESS_TERMINATING	42009
PROCESS_FAILING	42010

Activity events (AIE)

ACTIVITY_READY	21006
ACTIVITY_STARTED	21007
ACTIVITY_COMPLETED	21011
ACTIVITY_CLAIM_CANCELED	21021
ACTIVITY_CLAIMED	21022
ACTIVITY_TERMINATED	21027
(ACTIVITY_RESTARTED_EXIT_CONDITION_FALSE)	21041
ACTIVITY_FAILED	21080
ACTIVITY_EXPIRED	21081
ACTIVITY_LOOPED	42002
(ACTIVITY_SKIPPED)	42005
ACTIVITY_TERMINATING	42008
ACTIVITY_FAILING	42011
ACTIVITY_OUTPUT_MESSAGE_SET	42012
ACTIVITY_FAULT_MESSAGE_SET	42013
(ACTIVITY_USERINPUT_SET)	42014
ACTIVITY_STOPPED	42015

Events related to variables (VAR)

VARIABLE_UPDATED	21090
------------------	-------

Control link events (CLE)

CONTROL_LINK_EVALUATED_TO_TRUE	21034
--------------------------------	-------

Process template events (PTE)

PROCESS_INSTALLED	42006
PROCESS_UNINSTALLED	42007

Using process-related messages

Process Choreographer uses the WebSphere framework JRas for handling messages. All messages that belong to Process Choreographer are prefixed with *BPE*. The format of Process Choreographer messages is *BPE<Component><Number><TypeCode>*. The type code can be:

- I** Information message
- W** Warning message
- E** Error message

Process Choreographer messages can be found in the SystemOut.log, the SystemError.log, 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 Process Choreographer messages, check the activity.log file by using the WebSphere Log Analyzer.

Steps for this task




1. Click **Start > Programs > IBM WebSphere > Application Server v5.0 > Log Analyzer**.
2. **(Optional)** Click **File > Update database > WebSphere Application Server Symptom Database** to check for the newest version of the symptom database.
3. **(Optional)** Load Activity log.
Load the log from:
 - WebSphere_install_root\logs\activity.log on Windows systems
 - WebSphere_install_root/logs/activity.log on UNIX systems
4. Select the activity log file and click **Open**.

Chapter 11. Process Choreographer: Resources for learning

Use the following links to find relevant supplemental information about Process Choreographer. The information resides on IBM and non-IBM Internet sites, whose sponsors control the technical accuracy of the information.

These links are provided for convenience. Often, the information is not specific to the Process Choreographer, but is useful all or in part for understanding the product. When possible, links are provided to technical papers and Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

Planning, business scenarios, and IT architecture

-  **Process Choreographer Concepts and Architecture** at <http://www7b.software.ibm.com/wsdd/zones/was/wpc.html>
This paper uses scenarios to show how you can benefit from using Process Choreographer in today's business environment. It explains some basic business-process concepts that are used by Process Choreographer and describes how to develop, use, and administer business processes. An overview of the architecture is also given.
-  **Process Choreographer Staff Resolution Architecture** at <http://www7b.software.ibm.com/wsdd/zones/was/wpc.html>
This paper describes the architecture of the components involved in staff resolution. It explains the interaction between the Process Choreographer Web client, business process engine, work item manager, staff resolution plug-ins, and staff repositories, then focuses on the role of staff resolution.
-  **Web services and business process management** at <http://researchweb.watson.ibm.com/journal/sj/412/leymann.html>
This paper presents the relationship between Web services and the management of business processes in a tutorial-like manner.

Programming instructions and examples (not in this document)

- Process Choreographer sample
Follow the instructions in the Samples Gallery for accessing the samples on your local machine.

See also the set of links given in Enterprise beans: Resources for learning.

Chapter 12. Process Choreographer glossary

This glossary defines terms and abbreviations used by Process Choreographer.

Activity

An activity is one of the nodes that make up a process. An activity can be one of the following types:

- Process activity
- Elemental activity
- Empty activity
- Person activity
- Event activity

Activity instance

An activity instance is an instance of an activity template. It is created as part of a process instance during the execution of a process. At any given time, there can be more than one instance of an activity template.

Activity template

An activity template contains the specifications for an activity.

Activity types

An activity type describes alternative forms of implementations of an activity template. There are the following activity types:

- Process activity
- Elemental activity
- Empty activity
- Person activity
- Event activity

Adapter

A software system that invokes or is invoked by programming interfaces in other software systems and presents a standard interface, for example, one based on XML messages.

Audit trail

The audit trail is a log that contains an entry for predefined events that occur during the execution of a process instance. Audit logs are written by plug-ins. Process Choreographer provides a default plug-in that writes the log entries into a relational database.

Business process container

The business process container is the part of the Web application server that is responsible for the execution of business processes. Business processes that are installed on a Web application server are executed in their corresponding business process containers.

Block A process that is defined within its parent business process. Also known as an *inline subprocess*.

Claim If a user takes over the responsibility for working on an activity, this is referred to as claiming. If an activity is claimed, it is reserved for exclusive use.

Complete

When an activity is completed, the results of the activity are returned to the business process container for further processing. You can only complete activities that you have claimed.

Control link

Defines the possible flow of control between two nodes in a process. The actual flow of control is determined at run time based on the value of the transition conditions associated with the control link.

Custom attribute

An attribute that can be added to a process or activity by process modelers according to their needs.

Dead-path elimination

To guarantee the completion of a process, the business process container removes run-time paths that can no longer be reached during the execution of a process instance.

Deployment

To make a business process executable, it must be deployed. During the deployment of a process, the run-time artifacts required to run the process are generated, for example, Java classes specified in the FDML. The process is translated to a deployed form, which can be stored in a database. The result of deploying a process is a deployed process module (FAR file) that can be installed in a business process container.

See also *process module*.

EAR file

Enterprise application archive.

Early binding

The binding of activity implementations, such as services or processes, to elemental or process activities at process modeling or deployment time.

Editor (authorization level)

A list of users that provides edit access to the messages associated with an activity.

EJB Enterprise Java bean.**Elemental activity**

An elemental activity is an activity in a process that is implemented by a service.

Empty activity

An empty activity is an activity without an implementation, for example:

- Explicit synchronization points for parallel branches
- Explicit split or join nodes
- Explicit branch nodes

Event An external, asynchronous notification that is sent to a process. For example, an e-mail from a customer, or an EDI message.

Event activity

An event activity is an activity in a process that waits for an event.

Facade EJB

An Enterprise Java bean that provides a synchronous interface to a specific process with direct and type-safe access.

Facade MDB

A message-driven bean that provides an asynchronous interface to a specific process using JMS messages.

FAR file

Process archive.

Fault (also known as exception)

Faults indicate exception conditions. If a fault occurs, a fault message is sent to the corresponding fault terminal.

Fault message

The message that is associated with a fault.

Fault terminal

Activities and processes can have fault terminals. Fault terminals are used to model the error case. A fault terminal becomes active if the activity or process detect the corresponding error.

FDML Flow Definition Markup Language.

Input message

The input message contains all IN-parameters of an activity or a process.

Interruptible process

A long-running process that includes several transactions for execution. Individual activities can be executed in parallel, possibly even on different servers. A running instance of an interruptible process is associated with a persistent state in the process database so that the process can be recovered at any time. An interruptible process can contain all kinds of activities, including asynchronous invocations and activities that require human interaction.

JCA J2EE Connector Architecture.

JMS Java Message Service.

Late binding

The binding of activity implementations, such as processes, to activities, such as process activities, at run time.

MDB Message-driven bean.

Message

A data instance that is manipulated by a process is called a message. This includes data, which is:

- Passed when the process is started
- Passed to the input of an activity
- Produced by an activity
- Stored in a variable
- The result of the process
- The result of exceptions

Non-interruptible process

A short-running, non-interruptible process that is run as part of a single call or invocation to or from the process engine. A running instance of a non-interruptible process is associated with a thread of the Web application server executing on behalf of the non-interruptible process for its entire duration. A non-interruptible process can contain only activities that are implemented by synchronous operations.

Output message

An output message contains all OUT-parameters of an activity or a process.

Owner of an activity

The person who has claimed an activity and is therefore responsible for its completion.

Owner of a work item

The person who has received a work item. Ownership of a work item implies the responsibility to complete the associated activity.

Parent process

Is a process that contains an activity that is implemented by another process.

Person activity

An activity in a process that represents a human interaction. A person activity has associated staff queries that specify the set of eligible users to perform it, and additional properties to specify its behavior, such as GUI parameters and required user inputs.

Potential owner

A person who is eligible to claim the activity.

Process

A process consists of activities, where each activity represents a task, such as the invocation of a service, another process, or a human interaction. The sequence of execution of the activities is defined by using control links. Processes allow you to compose a service out of other services.

Process activity

An activity that is implemented by a process. The activity becomes a subprocess of the current process.

Process administrator (authorization level)

A list of users that provides administrative access to a business process and its activities. The process administration access right includes all other authorizations.

Process instance

A process instance is an instance of a process template. Multiple process instances of the same process template can be executed in parallel.

Process model

Is a synonym for business process and contains all definitions of a process.

Process module

A process module is an archive file that contains one or more processes, that is, FDML files that have been generated by the WebSphere Studio Tools. Process modules are deployed and installed as part of EAR files. When a process module is deployed, it can contain additional run-time artifacts, such as generated Java classes that are needed for the execution of the contained processes. Process modules are manageable entities that appear in the WebSphere Application Server Administrative Console.

Also known as a *FAR file*.

Process navigation

The process of execution of processes by the business process container.

Process template

A process template is a deployed and installed process that is used as a

template for process instances at run time. It contains all information of a process model. The process templates can be managed in the WebSphere Application Server Administrative Console.

Reader (authorization level)

A list of users that provides read-only access to an activity or a process.

Services

Operations that are made available by providing a service description (WSDL). Services are used to implement elemental activities of a process. There are Web services and local services. An example for a Web service is a SOAP service. Examples for local services are:

- Java methods
- EJB methods
- CICS transactions
- SAP remote function calls

Subprocess

A subprocess is a process that implements a process activity in a parent process.

Top-level process

A parent process at the highest level.

Transition condition

Determines the truth value of the associated control connector at run time, determining which paths of the process are taken at run time, and which paths are eliminated. The start condition of an activity refers to the truth values of the control links.

Variable

The schema of a message is described as a variable. The schema is described in the form of a WSDL message.

WAR file

Web application archive.

Web service

A Web service is a software application identified by a URL, whose interfaces and binding are capable of being defined, described, and discovered by XML artifacts. It supports direct interactions with other software applications using XML-based messages via Internet-based protocols.

Work item

Representation of work to be done in the context of an activity in a process instance.

Worklist

A worklist contains a set of work items obtained by a query using certain filter and sort criteria.

WSDL

Web Services Description language.

WSIF Web Services Invocation Framework.