Artix™

Artix Orchestration Administration Console

Version 4.2, March 2007

Making Software Work Together™

Updated: April 23, 2007

Contents

List of Tables

List of Figures

CHAPTER 1

Administration Console Basics

This chapter includes the following topics:

Introducing the Administration Console

Overview

The Artix Orchestration Administration Console allows you to manage and configure the Artix Orchestration engine and the artifacts that are deployed into it.

The Administration Console provides ways to deploy, select, inspect, and correct processes and related endpoint references.

Prerequisites

Before running the Administration Console in your browser, be sure to complete configuration and database setup by following the instructions in *Artix 4.2 Orchestration Installation Guide*.

Accessing the Admin Console

To access the Administration Console:

1. Start the Artix Orchestration server by opening a command prompt and running the following script:

   ```
   ArtixInstallDir\artix\VersionNumber\bin\start_bpel
   ```

2. Launch your web browser and type the following into the **Address** field:

   ```
   http://localhost:8080/BpelAdmin
   ```

   Alternatively, in Windows, select **Artix 4.2 Orchestration | Artix Orchestration Admin Console** from your Artix Orchestration group under **Start |(All) Programs**.

Using the Console Home Page

Home page options

The Home page of the Administration Console provides an overview of the engine that executes BPEL processes. It contains the following items.

Table 1: *Home Page Options*

Item	Description
Date Started	Engine start date
Deployed Processes	Number of business processes (BPEL files) currently stored in the database
Description	Engine configuration. This is the application server platform supported for this engine.
Status	Statuses for the Artix Orchestration engine are Running and Stopped. Additional database messages are included. Select **Storage** to see more detailed information regarding the database.
Version	Engine version number

Stopping and starting the engine

The engine starts when you run the `start_bpel` script and stops when you run the `stop_bpel` script.

Clicking **Stop Engine** on the console home page stops all running processes.

Start Engine changes the engine status to Running.

Administration Console Tasks

Tasks overview	This section provides a summary of the different tasks you can perform using the Administration Console with cross references to the relevant sections of the guide.
Configuring the Orchestration engine	For details on how to update engine configuration and tune engine performance, see "Configuration" on page 14.
Deploying and undeploying processes	If you are running the persistent version of Artix Orchestration server, you can deploy and undeploy processes from within the Administration Console. To deploy processes, see "Deploy BPR" on page 32. To undeploy a process: 1. Expire the version of the process from the Deployed Process Version Detail page. See "Expiring a process version" on page 37. 2. Remove completed and faulted instances of the process from the database. See "Deleting completed and faulted processes" on page 27. 3. Delete inactive plans from the database. See "Deleting inactive plans" on page 27.
Filtering active processes	You can analyze execution steps and diagnose problems from the Active Process page. See "Filtering active processes" on page 48 for details.
Enabling logging	You can enable logging and download a copy of the execution log for a running or completed process. See "Using the Process Details Page" on page 56.
Managing process exceptions	For details on managing process exceptions, see "Process Exception Management" on page 70.

Engine

The Engine section of the Administration Console menu contains the following options:

Configuration

Overview

The Configuration page contains the following tabbed pages:

- Engine Properties
- URN Mappings
- Function Contexts
- Alerts

Engine Properties

Overview

On the **Engine Properties** tab of the Engine **Configuration** page, you can make configuration changes without stopping and restarting the engine.

When you make a change and select **Update**, the changes take effect immediately. If you are using a database for persistence, the changes are also persisted to the database.

Engine configuration settings as shown in Table 2.

Note: Some of these properties are the same as the Artix Orchestration Designer Simulation preferences.

Table 2: *Engine Properties*

Property Name	Description
Auto create target path for Copy/To	Applies to processes that are validated against the BPEL4WS 1.1 specification only. For WS-BPEL 2.0 processes, this property can be added as an extension on a per process basis. See the *Extensions* help topic in the *Artix Orchestration Designer Online Help*. Determines if the Artix Orchestration server is allowed to create a location path for a non-existent node in a complex variable in a process instance document. When an assignment refers to a non-existent node (or to more than one node), the standard BPEL fault, `bpws:selectionFailure`, must be thrown, according to the BPEL specification. Enabling this option allows selections to be created on-the-fly. This means an assign copy TO operation can refer to a non-existent node and assign a value to it. This option is disabled by default.

Table 2: *Engine Properties (Continued)*

Property Name	Description
Disable bpws:selectionFailure fault	Applies to processes that are validated against the BPEL4WS 1.1 specification only. For WS-BPEL 2.0 processes, this property can be added as an extension on a per process basis. See the Extensions help topic in the *Artix Orchestration Designer Online Help*. Enabling this option allows a null value to be returned from a function or assignment that contains an XPath query string. You can enable this to override XPath behavior, for cases that handle data samples with optional elements. By default, this option is not enabled, and if the query string returns an empty selection from an assign copy FROM, the process throws a `bpws:selectionFailure` fault, which is the standard response described in the BPEL4WS specification.
Logging enabled	By default Artix Orchestration does not generate an execution log for running processes. Logging is turned off to enhance engine performance. You can enable this setting, and then view or download an execution log for a running or completed process. An execution log provides start/end times for activity execution and helps you troubleshoot faulting or faulted processes.

Table 2: *Engine Properties (Continued)*

Property Name	Description
Replace existing resources on deployment	Overwrites the current WSDL definition and other resources such as schema files. By default Artix Orchestration allows you to replace a resource file currently in cache without restarting the server. You can deploy a new version of a BPR file containing updated resources. BPEL developers who are testing and modifying processes and WSDL definitions may find this option useful.
Suspend process on uncaught fault	According to the WS-BPEL 2.0 specification, a process with an uncaught fault terminates. Enable this option to suspend all processes on an uncaught fault to put them in a suspended-faulting state. You can then perform process exception management on the faulting process followed by retrying or completing the faulting activity or scope. An individual process can override this setting with an entry in the PDD file. See "Process Suspension on Uncaught Fault" on page 88.
Validate input/output messages against schema	Validates the data loaded into process variables against the WSDL schema. Enable this option to validate data before execution starts. Disable this option for faster execution. This option is enabled by default.

Table 2: *Engine Properties (Continued)*

Property Name	Description
Deployment Plan Cache	A deployment plan corresponds to each deployed version of a process, including associated disposition of running processes. Process versions that are active can be cached for better engine performance. The default number of plans that are cached is 100. For details regarding versions, see "Process Versions" on page 83.
Process Count	The maximum number of processes in memory. The default number is 50. Specifying 0 indicates no limit, but is not recommended.
Process Idle Timeout	Number of seconds to wait until process state information is written to the database during idle processing times, such as waiting for a reply from an invoked service. Increasing the timeout value enhances engine performance. Decreasing the value ensures that the full process state is always in the database and so avoids potential process recovery time in the event of a server failure. The default is 10 seconds.
Resource Cache	The number of WSDL files and other resources in stored cache. The default is 100. Modifying the cache size may improve engine performance. A value of -1 means unlimited caching, but is not recommended.

Table 2: *Engine Properties (Continued)*

Property Name	Description
Unmatched Correlated Receive Timeout	The amount of time to wait (in seconds) for a correlated message to be matched to a receive, in the case that the message arrives before the receive becomes active. If this value is exceeded, a message is discarded so that the process can complete normally. The default is 300. Specifying 0 indicates that unmatched correlated messages are immediately discarded.
Web Service Timeout	The interval in seconds after which a process times out due to a reply or synchronous invoke activity not executing. The default is 600 seconds.
Work Manager Threads Per Process Max	The maximum number of execution threads the engine can spawn simultaneously for an individual process. The default is 10.
Work Manager Threads Min	The minimum number of execution threads the engine allocates for its Work Manager. The default is 10. This property does not appear in the Administration Console if Artix Orchestration server is configured to use an application server Work Manager.

Table 2: *Engine Properties (Continued)*

Property Name	Description
Work Manager Threads Max	Set the maximum number of execution threads the engine can spawn simultaneously. The default is 50. A value of -1 means that there is no maximum number of threads. This property does not appear in the Administration Console if Artix Orchestration server is configured to use an application server Work Manager.
Work Manager Threads Per Process Max	Set the maximum number of execution threads the engine can spawn simultaneously for an individual process. The default is 10.

URN Mappings

Overview

The Administration Console allows you to assign a physical address to a universal resource name (URN).

URN mappings provide a flexible and dynamic way to define target endpoint references. Use URN mappings to specify the physical address of a partner link endpoint reference instead of using the address specified in a process deployment descriptor (PDD) or WSDL file.

By mapping a URN to a URL, you do not have to rely on invoking a statically defined endpoint address. URN mappings give you flexibility, for example, to deploy the same BPR files for testing and production environments.

Instead of using the default invocation, you can specify a logical or physical address for a static endpoint reference in the PDD file. If you specify a logical address, or URN, you can then map the URN to the physical address in the URN Mappings page. If you specify a URL, you can replace the URL by mapping it to a different URL.

Examples

The following example illustrates one type of URN to URL mapping:

```
urn:localhost= http://localhost:8080/artix-bpel/services/{urn.3}
```

This mapping might be used when a process is deployed with the following partner link address information:

```
<partnerLink name="assessor">
   <partnerRole endpointReference="static"
     invokeHandler="default:Address">
      <wsa:EndpointReference xmlns:assessor="http://
       tempuri.org/services/loanassessor">
      <wsa:Address>urn:localhost:AssessRisk</wsa:Address>
      <wsa:ServiceName PortName=
        "SOAPPort">assessor:LoanAssessor</wsa:ServiceName>
   </partnerRole>
</partnerLink>
```

The Artix Orchestration invocation framework resolves the URN as follows:

```
urn:localhost:AssessRisk=http://localhost:8080/artix-bpel/services
   /AssessRisk
```

Mapping URNs to URLs

Table 3 shows different ways of mapping URNs to URLs. Note that each segment of the URN is separated by a colon. This means you can use a variable, such as `{urn.3}` shown in the second example, to indicate a replacable token in the third segment.

Table 3: *URN Mappings*

URN	URL
`urnSegment1:urnSegment2`	`http://localhost:8080/artix-bpel/services/MyService`
`urnSegment1:urnSegment2:urnSegment3`	`http://localhost:8080/artix-bpel/services/{urn.3}`
`http://ServerA:8080/artix-bpel/services/MyService`	`http://ServerB:8081/artix-bpel/services/MyService`
`urn:localhost:service`	`http://localhost:{$AE-NODE1-PORT}/artix-bpel/services/{$urn.4}`

The last example shows how to use variable substitution in a URL.

The URL values can optionally contain variables. The variables can be environment variables accessible through `java.lang.System.getProperties()` or a segment from the URN itself.

The Apache Ant style variable declaration of `${property}` is used to identify a property within the URL. Segments from the input URN value can be referenced by using a special property naming convention of `${urn.offset}` where `offset` is a one-based offset identifying the segment from the input URN value to use for substitution.

The URL in the mapping above contains two variables. The `{$AE-NODE1-PORT}` variable pulls the port number from an environment variable. This variable would need to be set as a `-D` parameter on the Java runtime environment (for example, `java -D{$AE-NODE1-PORT}=8080 ...`) or populated externally to the Artix Orchestration server.

The `{$urn.4}` variable in the above mapping references the fourth segment from the input URN value. Notice that the URN contains only three segments. The URN in the PDD file should contain at least one other segment. A sample URN might be:

```
urn:localhost:service:StoreService
```

The value of the fourth segment of this URN is `StoreService`. The resulting URL is:

```
http://localhost:8080/artix-bpel/services/StoreService
```

Updating or Deleting an URN Mapping

To update a URN mapping, select the URN. The URN and URL values appear in the text boxes where you can edit them and select **Update**. Editing the URN results in a new URN mapping. It does not update the existing one. Only the URL can be updated.

To delete a mapping, select the check box next to the mapping and click **Delete**.

Function Contexts

Using custom functions

On the **Function Contexts** tab of the Engine **Configuration** page, you can add custom function information.

Note: This option is available in the persistent version of Artix Orchestration only.

BPEL processes may contain custom functions that are used within XPath or other expression languages. Artix Orchestration provides a `FunctionContext` interface for implementation of custom functions. By using the `FunctionContext` interface, new or different functions may be installed and made available to the Artix Orchestration XPath (or another) expression writer.

If you already have custom functions implemented with a different interface, such as the jaxen `FunctionContext` interface, you can use them in your BPEL process.

Implementing the FunctionContext interface

To implement the `FunctionContext` interface, do the following:

1. Locate the following folder on your machine:

   ```
   ArtixInstallDir\lib\bpel\runtime_engine\1.0
   ```

2. Locate `ae_rtbpel.jar`.
3. The class file in `ae_rtbpel.jar` you need in order to implement the `FunctionContext` interface is:

   ```
   org.activebpel.rt.bpel.xpath.IAeXPathFunctionContext
   ```

Adding custom functions to Artix Orchestration Server

You add custom function details to make the functions known to the engine.

You can specify an absolute classpath location for the function or use a system property to indicate the location.

To add a custom function:

1. From the Engine Configuration page, select **Function Contexts**.
2. In the **Add Function Context Details** section, Type in a **Name** for the custom function. The name appears in the Custom Function list.
3. In the **Namespace** field, type the namespace that identifies the function.
4. Type in the fully qualified **Class** name of the container file that implements the custom function.
5. Type in a **Classpath** location for the custom function folder, ZIP or JAR file. The classpath can be an absolute path, or can be a system property.
6. Select **Add Context**.

Artix Orchestration server validates the function details and ensures that a class loader can load the class files.

If an error is reported, ensure that you have a valid class name and classpath location.

For each successfully added context, the name, namespace, and class of the function is displayed in a list. You can delete any function that you no longer need, if you delete the associated processes.

Alerts

Overview

On the **Alerts** tab of the Engine **Configuration** page, you can add the name of the service you want to run when processes are faulting.

Note: This option is available in the persistent version of Artix Orchestration only.

You can add the service name of a BPEL process that is designed to send out an alert when a certain process state is encountered, currently *suspended* or *faulting*. When the state occurs, the Artix Orchestration server instantiates the alert service, which can then invoke some action, such as notifying an administrator that a processing is faulting.

Adding an alert service

To add a service, type a name in the **Service** field and click **Update**. The service name is the My Role partner link service, identified in the PDD file deployed with the BPEL process to be used as the alert service. You can find this name by looking on the Deployed Process Version Detail page.

After you add the service, the Alert Service details are displayed: Process name, namespace, and partner link. Select the Process Name to view process version details.

For details on how to create an alert service, see the topic *BPEL processes as engine services* in the Artix Orchestration Designer help.

Storage

Overview

The Artix Orchestration server engine includes persistent storage based on the database settings you configured during installation.

You must configure one database before running the engine. See the *Artix 4.2 Orchestration Installation Guide* for details.

The Storage page displays database configuration properties and allows you to delete completed processes and deployment logs.

Database properties

The following relational database properties are displayed:

Table 4: *Database Properties*

JNDI Location	The Java Naming and Directory Interface (JNDI) context that specifies where to look for the database. For example, `java:comp/env/jdbc/ArtixOrchestrationBPEL`
Database Type	The database type, such as `mysql`

Deleting completed and faulted processes

To delete a completed or faulted process:

1. Enter a date in the **Completed/Faulted before** field.
2. Select **Delete**. The number of matching processes is displayed.
3. Click **OK**.

Deleting inactive plans

A *plan* consists of a process version plus the associated disposition of running processes. An *inactive plan* refers to a process version and associated processes that either have reached their expiration date or have been manually expired. If you have deleted completed processes from the database, you can delete the plan associated with those processes.

This means that the process version associated with the plan will no longer be displayed on the Deployed Processes page. The associated WSDL is not deleted, nor is any Partner Definition, since these files may be associated with other plans.

If a plan is associated with a subprocess, you cannot delete it until the main process is deleted. A *subprocess* is a BPEL process that is invoked by another process.

Deleting deployment logs

To delete a deployment log:

1. Select the type of log in **Log Contents**.
2. If desired, enter dates in the **Deployed between** fields.
3. Click **Delete**. The number of matching logs is displayed.
4. Click **OK**.

Version Detail

Overview

The Version Detail page shows the version number and build date of the Artix Orchestration engine libraries. This information may be useful for troubleshooting purposes.

Deployment

The Deployment section of the Administration Console menu contains the following options:

Deploy BPR

Overview

The Deploy BPR page allows you to add new business process archives (BPR) to the server.

Note: This option is available in the persistent version of Artix Orchestration only.

You can deploy one BPR at a time, but the archive can include as many BPEL files, deployment descriptors, partner definition files, and WSDL and schema files as you wish.

Deploying a BPR

To deploy a BPR:

1. Browse to a folder containing a business process archive.
2. Select a BPR file.
3. Click **Deploy**.

The engine validates the files contained in the BPR and stores the files in the database. The Deployment Log page appears showing errors, warnings, and information about the deployed process files.

After you deploy a BPR file, you can view details for deployment descriptors, partner definition files, BPEL files, indexed properties, WSDL definitions, and schema files by making selections in the Administration Console navigation bar.

Deployment Log

Overview

The Deployment Log page shows a list of logs generated when new and modified BPR files are deployed. The number of errors and warnings generated, if any, are shown.

On this page you can:

- Change the display of the logs list by using the **Selection Filter**
- Select a BPR file to view its deployment log

Selecting a deployment log

To select deployment logs:

1. Select the **Log Contents** type, if desired.
2. Select **Deployed between** dates, if desired.
3. Type in the exact **Name** of a BPR file, if desired.
4. Click **Submit**. The Deployment Logs list rebuilds based on your selection filters.

Deployment Log details

The Deployment Log page shows the name, date, and log for the selected BPR file.

During deployment, the engine validates the deployment descriptor of the BPEL process, ensuring that the associated WSDL file is available and valid for the current version of the process. If any validation errors or warnings occur, make corrections and redeploy the BPR file or create a new BPR file for any invalid processes.

Deployed Processes

Overview

The Deployed Processes page lists all processes that have been deployed to the server.

Select a process to display the Deployed Process Detail page.

Persistent version details

In the persistent version of Artix Orchestration, the following process version information is displayed.

Table 5: *Deployed Processes Details (Persistent Version)*

Item	Description
Name	Local part of the process qualified name (qname)
Active Ver.	Version that process instances can attach to or can run to completion. Normally the active version is the current version. However, if the current version has reached its expiration date, active processes can run to completion based on the expired version.
Versions	Number of deployed versions stored in the database
Future Ver.	Yes|no field indicating whether a process version has an effective date set to a future date

Filtering processes

You can select the following filters to view a subset of processes:

- Process **Status**, as described below
- **Process Name**. You must type in the exact name and select **Submit**.

Process statuses

Process versions can have one of the following statuses:

- **Current**. By default, when no version information is specified in a deployment descriptor, a deployed process is the current version with an immediate effective date. It is ready to receive requests.
- **Future**. If an effective date is specified in a process' deployment descriptor, a process has a future version.

- **Expired**. A version is expired if reaches the expiration date specified in a process' deployment descriptor, you manually expire the version, or a newer version is deployed.
- **Inactive**. When all process instances of an expired version complete, a process version is inactive.

See "Process Versions" on page 83 for details.

Deployed Process Detail

Non-persistent details

In the non-persistent version of Artix Orchestration, the Deployed Process Detail page displays the following information:

- Process name
- Namespace
- Process deployment descriptor (PDD) source
- BPEL source

Persistent details

In the persistent version, the Deployed Process Detail page shows the following information:

Table 6: *Deployed Process Details (Persistent Version)*

Item	Description
Version	The version number increments automatically when a new process version is deployed, unless a version number is specified in the deployment descriptor. The format of the number is N.nn, where N is major and n is minor.
Plan Id	The ID assigned to this version and associated disposition of running processes
Effective Date	If an effective date was not specified in the deployment descriptor, the effective date is the same as the Deployed Date. A process is effective immediately when deployed, unless an effective date is specified.
Expiration Date	A process version does not have an expiration date unless one is specified in the deployment descriptor. By default, a process version automatically expires when a newer version becomes effective.
Deployed Date	Date the process is added to the engine database
Processes	Number of active process instances
Migrated To	Process version that this version migrates to

Table 6: *Deployed Process Details (Persistent Version) (Continued)*

Item	Description
Status	Current, Future, Expired, Inactive

Expiring a process version

To expire the current version and inactivate future versions of a process, click **Expire**.

The result of expiring all versions is that all running processes attached to the current version will complete, but no new processes can be started. Future versions will not become active when their effective date arrives.

Note: To terminate a running process, navigate to the Active Processes page and select a process.

Restoring an expired version

To restore an expired version, see "Deployed Process Version Detail" on page 38.

Viewing version details

To view details for a single version click the version number for that version to go to the Deployed Process Version Detail page.

Deployed Process Version Detail

Overview

In the persistent version of Artix Orchestration, the Deployed Process Version Detail page displays all the details from the process deployment descriptor (PDD) as well as the process definition.

Details

The Deployed Process Version Detail page shows the following details:

- **My Role** partner link endpoint reference details are generated from information in the deployment descriptor. Hover over the partner link **Type** to view the associated namespace. Select the **Service Name** link to view the WSDL file for the Web service exposed by the partner link. For details see "Deployed Services" on page 40.
- **Partner Role** partner link endpoint reference details are generated from information in the deployment descriptor. Rest your mouse on the partner link **Type** to view the associated namespace. Select a static endpoint type from the **Linkage** column to view the endpoint definition.
- **Indexed Properties**, if any are displayed. For details, see "Indexed Properties" on page 42.
- The **Resource Usage** shows WSDL, schema, and other files and their target namespace referenced in this process. Click **Namespace** to view the resource definition.
- The **BPEL** source for the process

Viewing process graphs

Click the **View Process Graph** link to see the process in Outline view and Graph view. These views also are available for running processes. For explanations of these views, see "Using the Process Details Page" on page 56.

Updating a process

Depending on the version status (Current, Future, Expired, Inactive), you may be able to update the effective date, expiration date, and running process disposition

Expiring a process version

To inactivate a process version immediately, click **Expire**. No new process instances can attach to this version.

Restoring a process version

If you had previously expired a process version, you may be able to restore it. To restore the version to the current or future version, click the available option, **Restore to Current** or **Restore to Future**.

Deployed Services

Overview

A deployed process contains at least one My Role partner link, and this partner link is assigned a service name in the PDD.

The service name identifies the WSDL that the Artix Orchestration engine generates during deployment and adds to the Services page. The WSDL includes the messages, operations, service, and binding details for the Web Service exposed by the process' My Role partner link.

The process receives messages at the Web Service address, which is shown in the following example:

```
http://localhost:8080/artix-bpel/services/[servicename]?wsdl
```

Note: Some services are deployed as external, indicating they are not exposed as Web Services. For example, a Retry Policy service is deployed to tell the engine how many times to retry a non-communicating service. This type of process is not intended for outside consumption. External services are listed, but are not linked, on the Deployed Services page.

Details

The Deployed Services page shows the following details:

Table 7: *Deployed Service options*

Item	Description
Name	Service name assigned to a My Role partner link in the PDD file. Select the name to link to the WSDL generated for this partner link. The WSDL is the Web Service that receives inbound messages.
Process Name	Process containing the My Role partner link associated with this service

Table 7: *Deployed Service options (Continued)*

Item	Description
Binding	Standard SOAP binding styles indicating how to format inbound messages for the service. Can be one of: • Document Literal • RPC Literal • RPC Encoded • External • Policy
Partner Link	Name assigned to a My Role partner link that is exposing the service. Select the name to link to the Deployed Process Version Detail page.

Filtering services

To display a service, you can type its name into the Service Name field and click **Submit**.

You can also use the asterisk (*) wildcard to search for names. For example `*par*` returns all service names containing the “par” characters.

To view all services, leave the **Service Name** field blank and click **Submit**.

Indexed Properties

Overview

An indexed property is a variable property that serves as a selection filter for active processes. It holds a piece of data, such as a customer ID, application date, or amount. Using an indexed property in a selection query provides a fast way to filter processes based on important data items.

Note: This option is available in the persistent version of Artix Orchestration only.

For example, you can retrieve a list of faulting processes that share the same indexed property, suspend one or more processes, fix bad data values, and continue process execution. For details, see "Filtering active processes" on page 48.

Details

Indexed properties are defined in the PDD. Deployment details are as follows:

Table 8: *Indexed Property Details*

Item	Description
Plan Id	The deployed process associated with the indexed property
Name	Indexed property name. This name appears in the Indexed Property list in the selection filters Expression Builder.
Type	Property type, such as string or double
Variable Path	Process variable name and declaration location in the process
Part	Process variable part for message type variables
Query	Process variable part detail (optional)

For details on how to define an indexed property, see the *Artix Orchestration Designer Help*.

Partner Definitions

Overview

A partner definition file contains the service information for a partner link that has been deployed designated as a *principal* endpoint reference in the PDD.

Select a principal, if any exists, to view details.

Details

The following details are displayed for the selected principal.

Table 9: *Partner Details*

Item	Description
Partner link type	The partner link type used in the partner definition for the principal
Role	Role defined for the partner link type

Select a partner link type to view the namespace and endpoint reference details for the partner definition.

Resource Catalog

Overview

The Resource Catalog is the centralized cross reference for all WSDL, schema, XSL, and other resource files referenced in the PDD files deployed to the Artix Orchestration server engine.

Any resource in the catalog can be accessed by any deployed BPEL process, and only one copy is maintained. There are no restrictions based on the deployment context.

Catalog properties

The Resource Catalog page displays the following details:

Table 10: *Resource Catalog Properties*

Item	Description
Total Reads	The number of reads to retrieve resource information during process execution (in cache or not)
Disk Reads (%)	The number of reads made to resource files not in the cache expressed as an absolute number and percentage of Total Reads
Cache Size	The number of resource files in stored cache. The default is 100. You can set cache size on the Configuration page. Modifying the cache size may improve engine performance. See "Engine Properties" on page 15.

The Deployed Resources list shows the name and namespace for the WSDL. Hover over the **Type** or **Resource** name to view the physical location where the resource was loaded from.

Select a resource file to view details.

Viewing resource details

The Resource Detail page shows the same information that is on the Resource Catalog for each resource and also displays the XML source code.

Table 11: *Resource Details*

Item	Description
Type	Type of resource, such as WSDL, XSD, or XSL
Location	The actual physical location where the resource is loaded from. This helps to uniquely define the location when the deployment descriptor was created and can be used to have multiple resource files of the same name deployed to the engine. The WSDL location is referenced in the PDD file.
Namespace	Target namespace in the resource
Referenced By	The process versions referencing this resource

Updating and deleting resources

You can make minor changes to a resource, such as correcting a referenced URL, and then save your changes by clicking the **Update** button.

You can remove any resource from the database that is not directly referenced by any process. Click the **Delete** button, if available.

CHAPTER 4

Process Status

The Process Status section of the Administration Console menu contains the following options:

Active Processes

Overview

The Active Processes page shows a list of process instances that have been executed or are executing in the Artix Orchestration engine. The version that this instance is attached to is also shown. States can be *running*, *suspended*, *completed*, *compensatable* (for a subprocess), or *faulted*.

Select a Process ID or Name to view details of the process instance. For more information, see "Using the Process Details Page" on page 56.

Filtering active processes

You can use the comprehensive **Selection Filter** settings for advanced filtering and selecting of processes. For details, see "Filtering active processes" on page 48.

Note: If an active process is a subprocess, that is, it is invoked by another process, you may see additional state information for it. The additional state for a subprocess is *compensatable*. See the *Artix Orchestration Designer Online Help* for more information on creating a BPEL process to be used as a subprocess.

You can filter the active processes list by using a wide range of properties and functions. Select the filters to apply and then click **Submit**.

The selection filters include:

- **State**. Select one of the following process states:
  - **All**. All states of process instances.
  - **Running**. Normally running processes.
  - **Completed**. Normal completions.
  - **Compensatable**. A sub-process is complete and eligible for compensation.
  - **Faulted**. Processes completed with a fault.
  - **Suspended**. A process suspended for any reason.
  - **Suspended (Faulting)**. Suspended on a faulting activity. You can update variables on a faulting process prior to resuming it. For details, see "Process Exception Management" on page 70.
  - **Suspended (Activity)**. Suspended on a BPEL suspend activity.

- **Suspended (Manual)**. Manually suspended process.

- **Created between**. Date and time range for process starts.
- **Completed between**. Date and time range for process completions.
- **Name**. Process name.
- **Additional query**. Use the Expression Builder to create a query based on an extensive set of criteria. Click the Dialog button at the end of the row to open the Expression Builder.

Using the Expression Builder

You can create and submit a query for retrieving processes for display. In the Expression Builder, double-click the properties and functions to build the query, and Click **OK**. The expression appears in the **Additional query** text box. You can edit the expression and can use it in conjunction with the other criteria in the Selection Filters. Select **Submit** to retrieve processes that meet the criteria selected.

The following table describes the functions, variables, and properties you can use for filtering the active processes list.

Table 12: *Process Properties*

Criterion	Example Expression
End Date	getProcessProperty("EndDate") >= "2007/02/17 10:03 AM" Use the **Date** selector to enter a correctly formatted date.
Id	getProcessProperty("Id") = '102'
Name	getProcessProperty("Name") = 'LoanApproval'
Namespace	getProcessProperty("Namespace") = 'http://services.acme.com'
Start Date	getProcessProperty("EndDate") <= "2007/02/17 10:03 AM" Use the **Date** selector to enter a correctly formatted date.

Table 12: *Process Properties (Continued)*

Criterion	Example Expression
State	Property Codes for process states: 1 - Running 2 - Suspended 3 - Completed 4 - Faulted 5 - Compensatable Example: getProcessProperty("State") = '1'
State Reason	Property Codes indicating the reason why a process is suspended: 2 - Suspended (Activity). Suspended at a BPEL suspend activity 1- Suspended (Faulting). Suspended as a result of an uncaught fault 0 - Suspended (Manual). Suspended manually. Example: getProcessProperty("State'Reason') = '1'
Version	getProcessProperty("Version") = '2'

Table 13: *Indexed Properties*

Functions	Example
getParentId()	getParentId() = '101' (Returns all subprocesses whose parent process Id is 101)
getProcessProperty("...")	See Process Property examples above
getIndexedPropertyValue("...")	getProcessIndexedProperty('amount') >= 50000
hasWaitingAlarm()	hasWaitingAlarm()
hasWaitingReceive()	hasWaitingReceive() (For processes with a waiting receive or OnMessage)
hasWaitingReceive("partnerLinkName"[,op"])	hasWaitingReceive('{http://services.acme.com}AR', 'Invoice')

Table 13: *Indexed Properties (Continued)*

Functions	Example
isParentProcess()	isParentProcess() (Returns all processes that invoke another process) isParentProcess() OR isSubProcess() (Returns all processes involved either as parent or subprocess)
isSubProcess()	isSubProcess() (returns all processes that are subprocesses of parent processes)
nextAlarmTime()	nextAlarmTime() > '2006-12-31 14:30' (Returns all processes which have an alarm scheduled for any time after 2:30 pm on December 31, 2006)
not(...)	(getProcessProperty("Name") = 'LoanApproval' not(getProcessProperty("EndDate") >= '2005-02-01 4 am'))

Alarm Queue

Overview

The Alarm Queue page allows you to view a list of active On Alarm process activities.

Select one or more options from the **Selection Filter** option list to narrow your view of active alarms.

Table 14: *Alarm Queue Selection Filters*

Item	Description
Deadline Between	Beginning and Ending date and time for alarm
Process Id	Process instance Id. You can find this Id on the Active Processes page
Process Name	Local part of the process qualified name (qname)

Receive Queue

Overview

In the Receive Queue page, you can view a list of active receive and onMessage activities. These activities are queued for incoming messages.

Tasks

You can do the following from this page:

- Select a receive and then select a partner link to view details. A window opens where you can see the BPEL process location in which the receive activity executes. You can also see the correlation property alias and data, if any, associated with this receive activity.
- Select one or more options from the **Selection Filter** option list to view a selection of active receives. You can find this information on the Deployed Process Version Detail page, which shows the BPEL source code.

You do not need to enter the fully qualified name for the operation.

CHAPTER 5

Process ID and Process Details

This chapter contains the following topics:

Using the Process Details Page

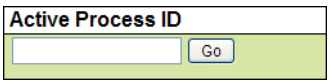
Viewing details for a process

The Active Process Detail page helps you analyze the execution state of a process instance.

To quickly view details for a particular process, type the instance ID for the process in the **Active Process ID** text box and click **Go**.

Note: To find a the process's instance ID, go to the Active Processes page.

Figure 1: *The Active Process ID Text Box*



The Active Process Detail page launches in a new browser window, providing a comprehensive snapshot of a running, suspended, faulting, faulted, or completed process instance.

Tasks

You can do the following from the Active Process Details page:

- View process and activity-level properties and values
- View the execution state of each activity
- Inspect the current value of variables, activity links, partner links, correlation sets, fault, compensation, and event handlers
- **Refresh** a running process to view an updated snapshot of the execution state
- **Suspend**, **Resume**, or **Terminate** a running process. These buttons appear at the top of the outline view.
- View and download the execution log to your computer from the process log. Select the **View Process Log** button at the top of the outline view. Copy the execution log to your computer from the **Process Log** text box.

Perform process exception management by correcting, resuming, retrying or completing a faulting activity. For details see

Note: If the Process Log box displays “Log file not available,” it means logging was not enabled when this process instance ran. For future process instances, you can enable logging from the Configuration page of the Administration Console.

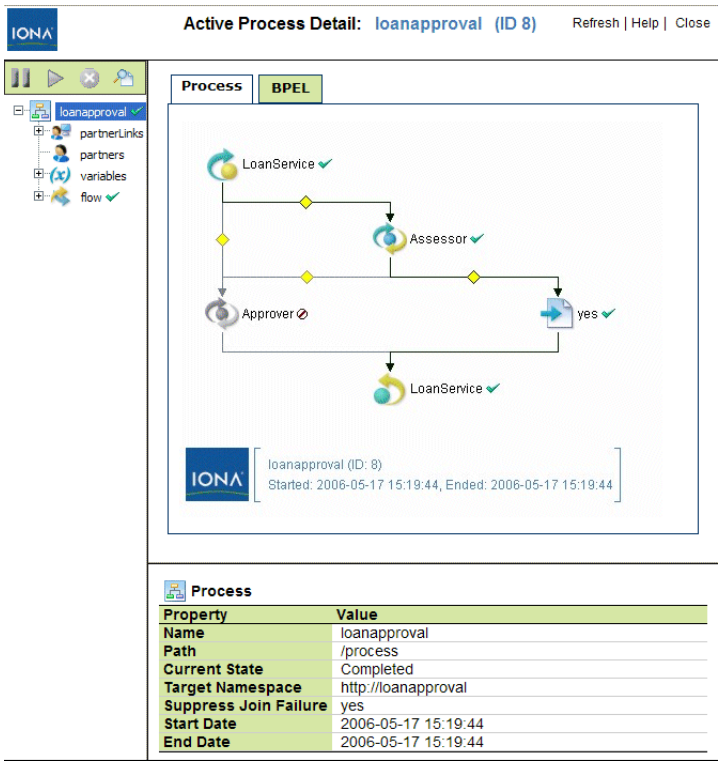
Views

The Active Process Details page presents many details about a process instance:

- An **Outline** view shows the structural elements of a BPEL process and the current process execution state of each activity. You can select an element to view its properties and values.
- A **Graphic** view shows the main process flow. If the process has event handlers, fault handlers, and compensation handlers, you can view them by selecting a tab, such as Fault Handlers shown below. You can also select an activity to view its properties.

- A **Properties** view appears for a selected element. For example, when the Process element is selected from the Outline view, you can see process properties and their current values.

Figure 2: *The Active Process Details Page*



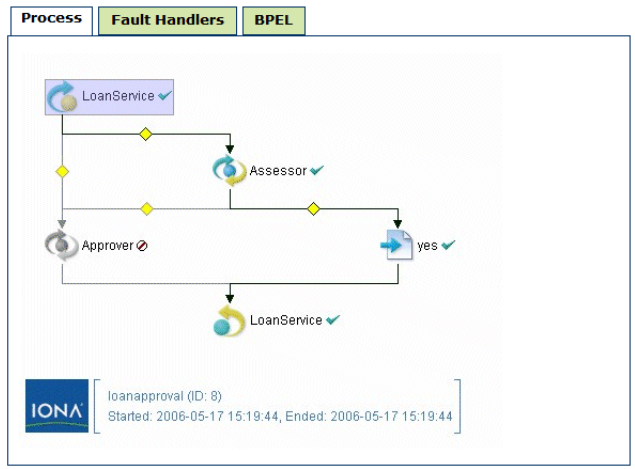
Using the Graphic View

Overview

The Graphic view of the Active Process Details shows the main process flow and the execution path through the process. You can also view the process fault, event, and compensation handlers, if the process definition includes these process-level handlers. The handlers have their own tabs in the view.

In the upper-right panel of the page, you see the main flow of a BPEL process. The process diagram reflects the layout rendering that is part of Artix Orchestration Designer.

Figure 3: *Active Process Details Graphic View*



Receive

Property	Value
Name	LoanService
Path	/process/flow/receive[@name='LoanService']
Current State	Finished
Operation	request
Partner Link	customer
Port Type	loanServicePT
Create Instance	yes
Variable	creditInformationMessage

Activity icons

Each process activity has an icon, a label, and an execution state indicator, as shown in the following illustration.

Figure 4: *A Process Activity*

1 Activity icon. Activity icons are the same as those supplied with Artix Orchestration Designer.

2 Activity label, which can be the activity type, name, type:name, or custom text

3 Execution state indicator. For a description of each indicator, see "Using the Process Details Page" on page 56.

Activity colors

Activities may appear in different colors, to indicate different execution states, as the following table describes.

Table 15: *Graphic View Activity Color Key*

Activity Color	Execution State
full color	Executed
muted color	Ready to execute or inactive
gray	Dead path

Viewing activity details

To view details for a particular activity:

- Select an activity from the diagram to view its properties
- Select an activity from the diagram to put the activity in focus in the Outline

Printing a diagram

To print the diagram, select **Print Picture** from the right-mouse menu. The diagram prints with the same caption that appears in the graph view. The timestamp indicates when the Process Details page was opened or refreshed.

To print a large diagram, select appropriate scaling options in your Printer options dialog, such as Fit to Page or print as x% of Normal Size

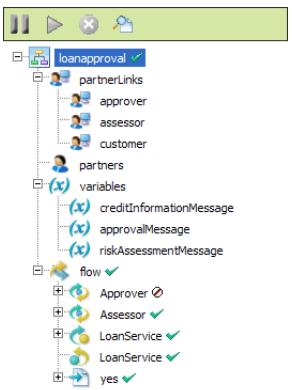
You can also save the diagram as an image file to print later.

Using the Outline View

Overview

The Outline view shows allows you to manipulate a BPEL process and view properties for the individual elements in the process.

Figure 5: *A BPEL Process in the Outline View*



Outline view menu bar

The **Suspend**, **Resume**, and **Terminate** buttons are enabled only if the process is currently running or suspended. If logging is enabled, you can click the **View Process Log** button to display the execution details for the process instance. See "Engine Properties" on page 15 to enable logging.

Figure 6: *The Suspend, Resume, and Terminate Buttons*

BPEL process elements

The Outline view displays the following BPEL process elements:

- **Process name** is the local part of the process qualified name
- **Partner links** represent the Web services that are invoked

- **Variables** contain the message or other data received, manipulated, and sent from the process
- **Correlation sets**, if present, contain the message properties that track different conversations carried on by the process
- **Fault handlers**, if present, catch faults thrown by process activities
- **Event handlers**, if present, run concurrently with a process scope and invoke an activity based on an alarm or message received
- **Termination handlers**, if present on a scope, handle process termination
- **Activities** carry out the processing steps

Details

To view details about a process element, select it. Table 16 describes each element.

Table 16: *Process Element Details*

Process Element in Outline	Details Displayed
Process name	Current state, Start/end time of process instance, and deployment details for the process. Fault details may also be displayed.
Partner links	The type(s) of partner links: partner role and/or my role. The endpoint reference of the partner link service. You can see the address information in the Properties view.
Variables	The variable type: message, schema type or schema element. The current value of the variable. For a running process, the value is current as of the time you opened or refreshed the Process Details window.
Correlation sets	The message property definition and current value. A correlation set contains a message property to ensure that each process conversation is uniquely identified.

Table 16: *Process Element Details (Continued)*

Process Element in Outline	Details Displayed
Fault handlers	Name, state, and details of fault handling activity. Scopes can have their own local fault handlers.
Event handlers	Name, state, and details of event handling activity. Scopes can have their own local fault handlers.
Activities	The activities section of the Outline begins with a flow activity that represents the main container for the whole process. Within the flow, there is a list of all process activities. The activities are in the same order as in the BPEL source. If the process was designed in Artix Orchestration Designer, the order matches the Outline view order. The activity list shown is not necessarily in execution order. For each activity, you can view the execution state and activity definition.
Links	If an activity is the source of a link, the link is displayed below the activity node. Link properties are displayed, including link status (whether or not the link executed), the transition condition, if it exists, and the link's target activity.

Activity states

You can determine the execution status of each activity by looking at the icon next to the activity.


Executing


Ready to Execute


Finished


Faulted. Occurs when a fault is thrown during the execution of an activity.

Faulting. Occurs when a fault is thrown during the execution of an activity and the fault is uncaught. If desired, you can make corrections or resume faulting processes. See "Process Exception Management" on page 70.

Terminated. Occurs when the process is manually terminated.

Dead Path

Suspended

(none) Inactive (the initial state of an activity)

For a running process, the icon next to an activity may change if you refresh the Process Details window.

Process states

The process can have the following execution states:

Table 17: *Process States*

Execution State	Description
Completed	Normal completion
Faulted	Completed with a fault or termination
Running	Snapshot of the executing process when you open the Process Details window. The process continues to run, but the Process Details window is not updated unless you select **Refresh**.
Suspended	The process stops running when you select Suspend from the Process Details window.
Faulting	Execution is stopped on a faulting activity. The activity has an uncaught fault and the process is configured for suspension on an uncaught fault.

Inspecting fault details

In the Process Details window, the Outline view shows a list of process activities. A red X appears next to an activity that faulted, or a red triangle next to a faulting activity.

You can select the process name to view details about the fault.

Figure 7: *A Faulted Process*

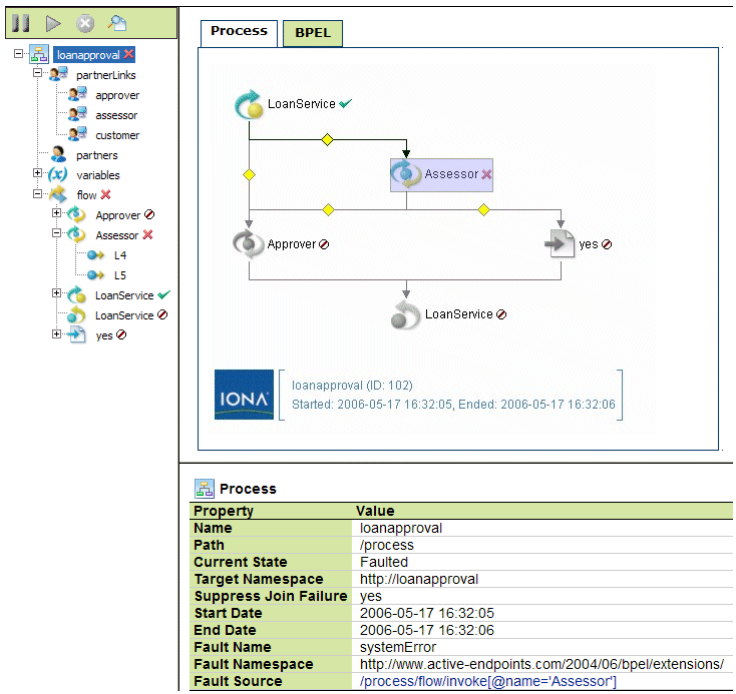


Table 18: *Fault Details*

Property	Description
Fault Name	Standard BPEL or engine fault name
Fault Namespace	Standard BPEL or engine fault namespace
Fault Source	Process activity that threw the fault
Fault Message Data	Data in the throw or catch fault variable

For more information

To get further information about faults:

- Select the faulted activity to view the Fault Name. See “Process Exception Management” on page 70 for details.
- Select **View Process Log** in the Outline to view the process log. You can see the execution path leading to the faulted activity.
- For a faulting activity, you can correct data, retry or complete the activity. See “Process Exception Management” on page 70.

Note: If the Process Log is not visible, you must enable logging on the Configuration page.

Fault Handling

This section covers the following topics

Process Exception Management

Overview

A process can receive an unexpected fault during its execution. In such an event, the process may not have the necessary fault handling logic, and the process may terminate. A more desirable result is to suspend the process, retry, step over, or make corrections and then complete the process normally.

You can configure Artix Orchestration server to suspend processes on uncaught faults. You can then use the following process exception management techniques available in the Administration Console:

- Find faulting processes quickly by using selection filters in the Active Processes page
- Update the value of variables, partner links, and correlation properties
- Retry an activity or scope
- Mark an activity or scope as Complete and continue to the next activity
- Set up alerts for faulting processes

Suspending a process on an uncaught fault

You can configure the engine to suspend processes on uncaught faults so that you can inspect a problem, make corrections, if desired, and continue the process.

For details, see "Engine Properties" on page 15. In addition, on a process-by-process basis, you can override the engine-wide setting with a setting in the PDD file. The global setting may not be desirable for all process types. For example, a "straight-through" process, which has someone waiting for a response, may not be a desirable candidate to suspend.

Correcting faults

Use the Selection Filters on the Active Processes page to view a list of faulting processes.

Select a process from the list and then from the Active Process Details page, you can do any of the following to correct a problem:

- Update the value of variables
- Update the endpoint reference of partner links, such as the service name or `ws:address`

- Update the value of correlation properties
- Retry, complete, or step activities

Selecting a faulting process

To select a faulting process:

1. From the Active Processes page, select a process with a Suspended-Faulting state. Use the Selection Filters to quickly locate the process you are looking for.
2. Click the **Plan Id** to display the Active Process Details page.

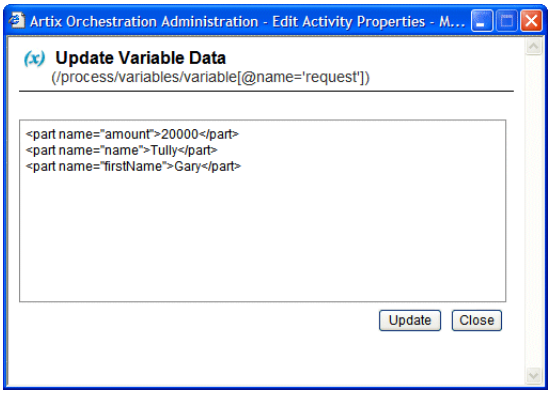
Updating the value of a variable on a faulting activity

A variable may cause an uncaught fault if it is invalid with regard to an activity's operation.

To update a variable value:

1. From the Outline view of the Active Process Details page, select a variable. The current value of the variable is shown in the Variable Data Instance box.
2. Click **Edit** at the bottom of the Variable Data Instance box.
3. In the Update Variable Data dialog, make the modifications necessary. Possible modifications include:
   - data value(s)
   - XML data structure

Figure 8: *The Update Variable Data Dialog*



4. Click **Update**.

Updating partner link data on a faulting activity

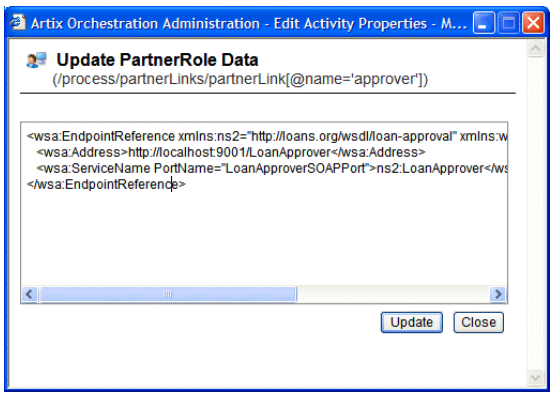
A partner link endpoint reference may cause an uncaught fault for a variety of reasons:

- The service is unavailable
- The address contains invalid information
- The address is missing required information, such as a header or credentials

If an endpoint is not available or the address is incorrect or incomplete, you can supply new `ws:address` information for a faulting partner link, as follows.

1. From the Outline view of the Active Process Details page, select a partner link. The current value of the endpoint reference is shown in the Partner Role Data box.
2. Click **Edit** at the bottom of the box.
3. In the Update Partner Role Data dialog, make the modifications necessary.

Figure 9: *The Update Partner Role Data Dialog*



4. Click **Update**.

Updating a correlation set on a faulting activity

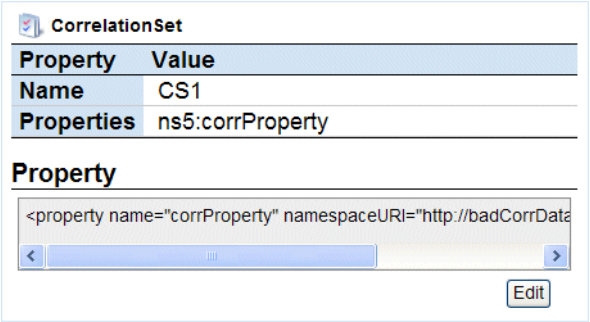
A *correlation set* is a set of properties shared by messages. It acts as a conversation identifier by keeping together all messages intended for the same process instance. If an activity is faulting due to bad message data, you may need to correct the correlation property data as well as the message variable data.

Correcting correlation data requires a comprehensive understanding of the message properties defined in the process's WSDL file and the expected value in the property alias associated with the input/output variables of receive, onMessage, invoke, and reply activities. Also, be aware that a correlation set can be declared at the process level or at a scope level. For details on defining and using correlation sets, see the *Artix Orchestration Designer Online Help*.

To update a correlation set:

1. From the Outline view of the Active Process Details page, select a correlation set. The current value of the correlation property is shown in the Correlation Set Property box.

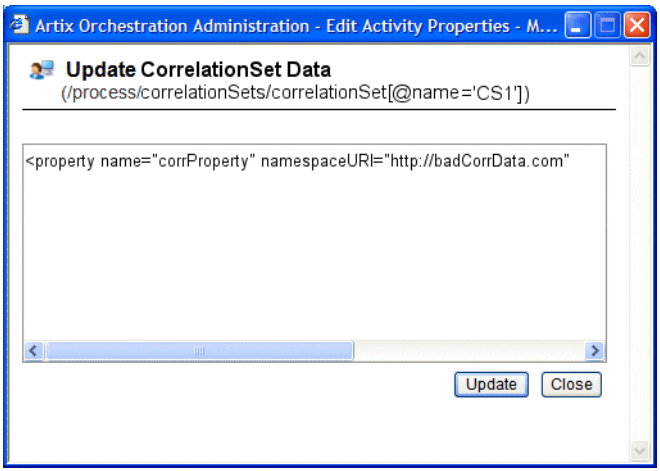
Figure 10: *The Correlation Set Property Box*



2. Click **Edit** at the bottom of the box.

3. In the Update Correlation Set Data dialog, make the modifications necessary.

Figure 11: *The Update Correlation Set Data Dialog*



4. Click **Update**.

Using Step, Retry and Complete

After correcting the variables, correlation properties, or partner links causing an uncaught fault (if desired), you can retry or complete (step over) the faulting activity or an enclosing scope.

On the Active Process Details page, select the faulting activity or scope from the Outline view or Graph view. The activity shows a Faulting state and includes an **Action** bar.

Table 19: *Action Bar Options*

Button	Action	Description
	Step	Steps the current activity. After stepping an activity, the next activity (if any) becomes ready for execution, and you can continue to step through the process or use the process level Resume button to continue. This button is disabled for a faulting activity since you can accomplish the same result simply by resuming the process from the Outline view menu bar.
	Retry	Retries the faulting activity or scope. If you corrected the cause of an uncaught fault, the next activity will become ready to execute and you can either resume the process or continue to step through the process one activity at a time. If you retry an executing scope, then the scope will first terminate any of its enclosed activities that are still running and also issue a compensate call to any eligible enclosed scopes prior to retrying.
	Complete	Marks the activity or scope as completed normally. The next activity will become ready to execute, and you can either resume the process or continue to step through the process one activity at a time. Keep in mind that you may have to take some additional steps to fix a process if you use the Complete button. Since the activity that you are marking as complete does not execute, there may be variables or correlation sets that are not properly initialized as a result of completing the activity.

Setting up alerts for faulting processes

When unexpected faults occur, you can manually check for the occurrence of suspended processes through the Active Process Page. Alternately, you may want to have the system notify you if there was a problem with one of your processes. To do so, you can designate a service to be notified of these faults. The service can then take steps to dispatch notifications of the problem (e.g., to an enterprise management system or pager) or in systemic cases automate recovery of the process.

Artix Orchestration allows you to designate a BPEL process as an alerting service. The service, which you can create in Artix Orchestration Designer, is based on a WSDL file that defines the schema, messages, port types, operations, and partner link types available for building an alerting system. Once the process is deployed, you can go to the **Alerts** tab of the **Configuration** page to set up the service to handle alerts.

For more information, see:

- "Alerts" on page 26
- The Process Exception Management topic in *Artix Orchestration Designer Online Help*
- The `alerts.wsdl` file in the *`ArtixInstallDir`*`\artix\`*`VersionNumber`*`\etc\bpel` directory, which is required for creating an alerting service.

Process exception management and endpoint policies

Artix Orchestration uses endpoint policies governed by the WS-Policy specification. A policy can describe when to avoid invoking a service, based on system downtime and how many times to retry a service that does not reply. The endpoint policy information is added to the PDD file of a process.

BPEL Standard Faults

List of faults

The following list specifies the standard faults defined within the WS-BPEL specification. All these faults are named within the WS-BPEL namespace standard prefix `bpel:` corresponding to URI:

```
urn:oasis:names:tc:wsbpel:2.0:process:executable.
```

Table 20: *BPEL Standard Faults*

Fault Name	Description
ambiguousReceive	Thrown when a business process instance simultaneously enables two or more IMAs for the same partnerLink, portType, operation but different correlation Sets, and the correlations of multiple of these activities match an incoming request message
completionConditionFailure	Thrown if upon completion of a directly enclosed `scope` activity within a `forEach` activity it can be determined that the completion condition can never be true
conflictingReceive	Thrown when more than one receive activity or equivalent (currently, onMessage branch in a pick activity) are enabled simultaneously for the same partner link, port type, operation and correlation set(s)
conflictingRequest	Thrown when more than one synchronous inbound request on the same partner link for a particular port type, operation and correlation set(s) are active

Table 20: *BPEL Standard Faults (Continued)*

Fault Name	Description
correlationViolation	Thrown when the contents of the messages that are processed in an `invoke`, `receive`, `reply` `onMessage`, or `onEvent` do not match specified correlation information
invalidBranchCondition	Thrown if the integer value used in the `branches` completion condition of `forEach` is larger than the number of directly enclosed `scope` activities
invalidExpressionValue	Thrown when an expression used within a WS-BPEL construct (except `assign`) returns an invalid value with respect to the expected XML Schema type
invalidVariables	Thrown when an XML Schema validation (implicit or explicit) of a variable value fails
joinFailure	Thrown when the join condition of an activity evaluates to false and the value of the `suppresJoinFailure` attribute is `yes`
mismatchedAssignmentFailure	Thrown when incompatible types are encountered in an assign activity
missingReply	Thrown when an inbound message activity has been executed, and the process instance or scope instance reaches the end of its execution without a corresponding `reply` activity having been executed

Table 20: *BPEL Standard Faults (Continued)*

Fault Name	Description
missingRequest	Thrown when a `reply` activity cannot be associated with an open inbound message activity by matching the partner link, operation and message exchange tuple
scopeInitializationFailure	Thrown if there is any problem creating any of the objects defined as part of scope initialization. This fault is always caught by the parent scope of the faulted scope
selectionFailure	Thrown when a selection operation performed either in a function such as `bpel:getVariableProperty`, or in an assignment, encounters an error
subLanguageExecutionFault	Thrown when the execution of an expression results in an unhandled fault in an expression language or query language
uninitializedPartnerRole	Thrown when an `invoke` or `assign` activity references a partner link whose partnerRole endpoint reference is not initialized
uninitializedVariable	Thrown when there is an attempt to access the value of an uninitialized variable, or in the case of a message type variable, one of its uninitialized parts
unsupportedReference	Thrown when a WS-BPEL implementation fails to interpret the combination of the reference-scheme attribute and the content element OR just the content element alone

Table 20: *BPEL Standard Faults (Continued)*

Fault Name	Description
xsltInvalidSource	Thrown when the transformation source provided in a `bpel:doXslTransform function` call was not legal
xsltStylesheetNotFound	Thrown when the named style sheet in a `bpel:doXslTransform` function call was not found

Artix Orchestration Custom Faults

List of faults

The following list specifies the custom faults defined for the Artix Orchestration engine. All these faults are in the namespace:

```
http://www.active-endpoints.com/2004/06/bpel/extensions/
```

Table 21: *Artix Orchestration Custom Faults*

Fault Name	Description
systemError	Unrecoverable system error
badProcess	Invalid BPEL
validationError	Error in message variable data. Validation errors are reported only if the configuration option “Validate input/output messages against schema” is enabled.
xpathFunctionError	Error in executing XPath function
invalidTransitionCondition	Non-Boolean return from an XPath evaluation of a transition condition
xpathDateParseError	Error in parsing an `xsd:date` or `xsd:datetime`
xpathDurationFormatError	Error in parsing an `xsd:duration`

CHAPTER 7

Process Versions

The persistent version of Artix Orchestration server manages process versions. This chapter includes the following sections:

Process Version Life Cycles

Overview

Process versioning allows different versions for a given process to exist in Artix Orchestration. Two deployments are considered to be different versions of the same process if they have the same target namespace and name in the BPEL file, but one deployment differs from the other in some way.

Process versioning allows you to control when processes become effective and for how long. You can also control what happens to processes created by older versions when a new version becomes effective. While multiple versions of a process can exist concurrently, only the latest effective version is capable of creating new process instances.

Note: Process versioning is available in the persistent version of Artix Orchestration only.

Version states

The latest effective version is in a *current* state. Other states include *future*, to describe versions that have an effective date in the future, *expired* to describe versions whose expiration date has arrived or has been set, and *inactive* to describe expired versions that no longer have running process instances.

Versioning in a PDD

The process deployment descriptor provides selections for describing how a deployment is to be versioned. These selections are all optional and have default values as described below.

The following example shows the syntax for version information in the PDD file.

```
<version effectiveDate="2005-12-12T00:00:00-05:00"
   expirationDate="2007-12-12T00:00:00-05:00" id="1.5"
   runningProcessDisposition="migrate"/>
```

where:

- **`effectiveDate`** is the date the new version becomes the current version and all new process instances run against it. Depending on the disposition selected for running processes, some may continue to run

until complete using the older version. The effective date is an XML schema date/time value. The time expression includes a time zone, indicated as the midnight hour plus or minus the number of hours ahead of or behind Coordinated Universal Time (UTC) for the computer's time zone. In the example above, the computer time zone is eastern standard time, which is five hours behind UTC. If you do not provide an effective date, it defaults to the date and time the process is deployed to the server.

- **`expirationDate`** is the date, beyond the effective date, the current version expires. An expired version is not capable of creating new process instances. Once all of the running processes tied to an expired version complete then the version becomes inactive. All process instances for the current version run to completion. The expiration date is an XML schema `datetime` value. (Same as effective date). If you do not provide an expiration date, the version does not expire until you manually expire it in the Administration Console or until a newer version is deployed.
- **`id`** is the process version number in major.minor format. You do not need to provide a version number. Artix Orchestration server auto-increments new versions. The server increments a version number by dropping the minor value and adding 1 to the max number. For example, version 1.5 increments to version 2.0.
- **`runningProcessDisposition`** is the action the server takes on any other versions of the same process that currently have processes executing once this version's effective date arrives.

---

runningProcessDisposition valid values

Valid values for `runningProcessDisposition` are:

- **`maintain`**—Indicates that all process instances for the previous versions should run to completion. This is the default value.
- **`terminate`**—Indicates that all process instances running under previous versions should terminate on the effective date of the new version, regardless of whether or not the process instances are complete.
- **`Migrate`**—All running process instances created by previous versions will have their state information migrated to use the newly deployed process definition once its effective date arrives. If there are

incompatible changes between the versions that would not permit them to be migrated, you receive an error message during deployment of the new process, and its deployment fails. Changes should be limited to expression language changes in the BPEL file.

Criteria for a new version

A new version of a deployed process must meet the following criteria:

- The fully qualified process name must not change
- A new version can include a change to either the BPEL XML or the process deployment descriptor file
- A version is not new if the autoincrement feature determines that the BPEL source file and the deployment descriptor are the same as the deployed version having the highest version number. If the BPEL and the descriptor file do not contain any differences, then the version on the server is up-to-date and no deployment occurs.
- If you need to modify WSDL-related properties, such as partner links or correlation properties, then you should create a new process, not a new version

Process Version Persistence Type

Overview

Persistence refers to storage of active processes. When a process runs on the server, by default all state information is stored in the server database. However, this setting can be changed in the PDD file.

Persistence setting selections are as follows:

Table 22: *Persistence Settings*

Item	Description
System Default	The current engine setting for all processes. The default engine setting is Full persistence
Full	The default setting. For each process instance, all state information is stored for a running, faulted, and completed process.
None	No process information is stored in the server database when a process terminates

Process Suspension on Uncaught Fault

Overview

According to the WS-BPEL 2.0 specification, an uncaught fault forces a process to exit.

On the Engine Properties tab of the Configuration page, you can enable an option to suspend all processes on an uncaught fault to put them in a suspended-faulting state. You can then perform process exception management on the faulting process followed by retrying or completing the faulting activity or scope.

An individual process can override the engine setting with an entry in the PDD file. The settings are:

Table 23: *Process Suspension Settings*

Item	Description
System Default	The current engine setting for all processes. The default engine setting is to disable suspension on uncaught fault; however the current setting may be different.
False	Do not allow this process to suspend on an uncaught fault. The process will terminate abnormally. This setting overrides the engine setting.
True	Suspend this process on an uncaught fault to put it in a suspended-faulting state. You can then perform process exception management on the faulting process, followed by retrying or completing the faulting activity or scope. This setting overrides the engine setting.

See "Process Exception Management" on page 70 for details.