Oracle® BPEL Process Manager

Quick Start Guide

10*g* (10.1.3.1.0)

B28983-01

September 2006

Oracle BPEL Process Manager Quick Start Guide, 10*g* (10.1.3.1.0)

B28983-01

Primary Author: Mark Kennedy

Contributor: Guru Balse, Dave Shaffer, Raj Venkatesan

Contents

3 Reviewing a Loan Procurement BPEL Process

Index

Preface

This guide is the primary source of introduction and usage information for Oracle BPEL Process Manager.

Audience

This manual is intended for users who want to install and use Oracle BPEL Process Manager.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

For more information, see the following Oracle resources:

- *Oracle BPEL Process Manager Order Booking Tutorial*
- *Oracle BPEL Process Manager Developer's Guide*

Printed documentation is available for sale in the Oracle Store at

`http://oraclestore.oracle.com/`

To download free release notes, installation documentation, white papers, or other collateral, visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

`http://www.oracle.com/technology/membership/`

To download Oracle BPEL Process Manager documentation, technical notes, or other collateral, visit the Oracle BPEL Process Manager site at Oracle Technology Network (OTN):

`http://www.oracle.com/technology/bpel/`

If you already have a username and password for OTN, then you can go directly to the documentation section of the OTN Web site at

`http://www.oracle.com/technology/documentation/`

See the *Business Process Execution Language for Web Services Specification*, available at the following URL:

`http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbizspec/html/bpel1-1.asp`

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Getting Started with Oracle BPEL Process Manager

This chapter provides an overview of how the business process execution language (BPEL) and Oracle BPEL Process Manager enable you to design service-oriented architecture (SOA)-based applications by developing synchronous and asynchronous services into end-to-end BPEL process flows. This chapter also describes how to install and start the Oracle BPEL Process Manager, Oracle BPEL Server, and other key components.

This chapter contains the following topics:

- BPEL: The Cornerstone of SOA
- Oracle BPEL Process Manager
- Installing Oracle Application Server SOA Suite and Oracle JDeveloper
- Starting and Stopping Oracle BPEL Process Manager Components
- Verifying the Hostname in Your Web Browser Preferences
- Understanding Namespaces and Namespace URIs

BPEL: The Cornerstone of SOA

Many companies are looking at Web services and SOA as a method for addressing the integration requirements involved in building connected applications. While SOA has existed for over a decade, there was confusion about which interfaces to adopt. BPEL and Web service standards solve this dilemma by addressing common application requirements in an open, portable, and standard way. SOA optimizes business performance by using existing resources and minimizing the cost of deploying new applications. Enterprises adopting these standards and architectural approaches achieve a significant return on investment (ROI) from using the same standards-based approach to building connected applications that they used for building Web applications with Java/Java 2 Platform, Enterprise Edition (J2EE).

Making Web services work is a two-step process:

1. Publish the services.

   Publishing a service involves taking a function within an existing application or system and making it available in a standard way.

2. Compose, or orchestrate, the services into business flows.

   Orchestration involves composing multiple services into an end-to-end business process.

Web services standards, including web services description language (WSDL), extensible markup language (XML), and simple object access protocol (SOAP), have emerged as an effective and highly interoperable platform for publishing services. In addition, high performance binding frameworks enable enterprises to access legacy systems and native Java code without having to wrap them in a SOAP interface.

BPEL Concepts

BPEL is emerging as the clear standard for composing multiple synchronous and asynchronous services into collaborative and transactional process flows. BPEL benefits from over 15 years of research that improves upon its predecessor languages of XLANG and WSFL. BPEL provides the following features:

- Web services/WSDL as component model
- XML as data model (data loose-coupling)
- Synchronous and asynchronous message exchange patterns
- Deterministic and nondeterministic flow coordination
- Hierarchical exception management
- Long-running unit of work/compensation

Since the BPEL specification was submitted to the Organization for the Advancement of Structured Information Standards (OASIS) in March 2003, it has gained the support of nearly every major industry vendor. This provides a great benefit to enterprises that can now implement their business processes in a standard and portable way, avoiding vendor-specific rules to a degree not previously possible.

Oracle BPEL Process Manager

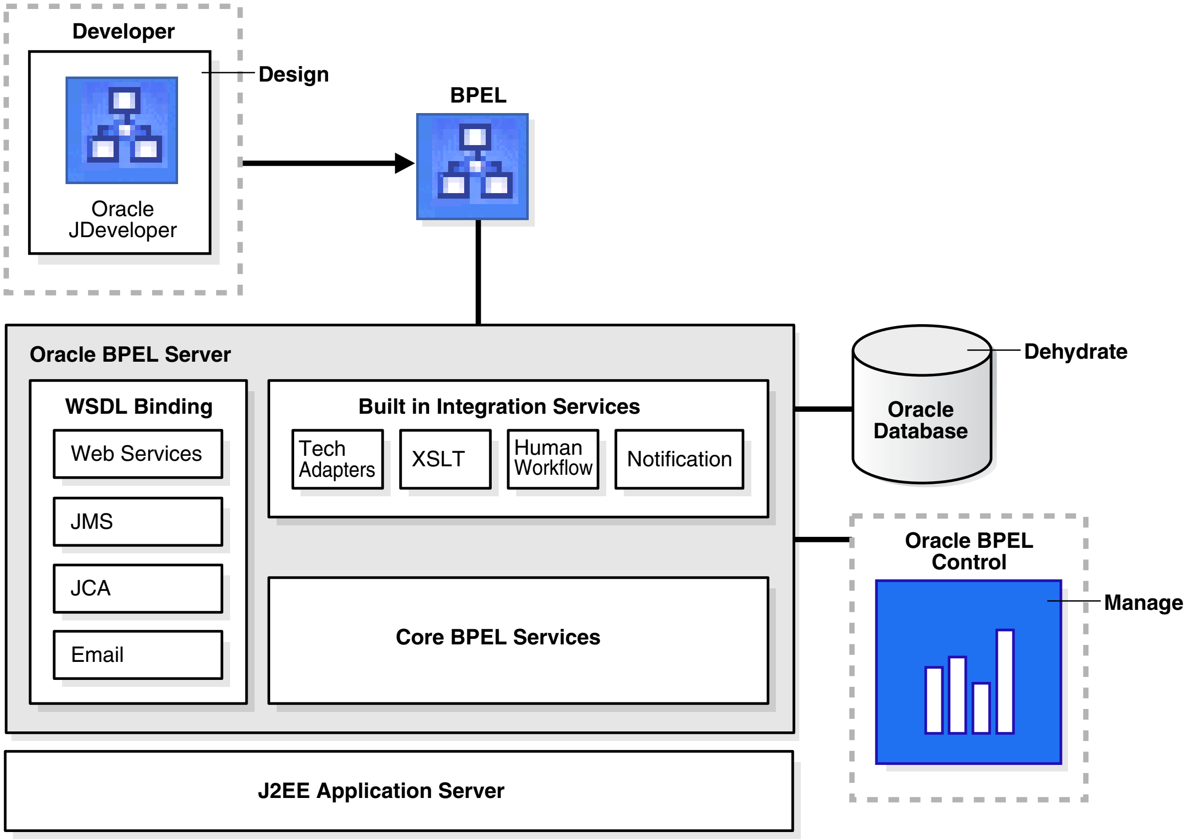
This section contains these topics:

- Introduction to Oracle BPEL Process Manager
- How Is Oracle BPEL Process Manager Different?
- Oracle BPEL Process Manager Feature Summary

Introduction to Oracle BPEL Process Manager

Oracle BPEL Process Manager provides a user-friendly and reliable solution for designing, deploying, and managing BPEL business processes. Oracle BPEL Process Manager consists of the key components shown in Figure 1–1.

Figure 1–1 Oracle BPEL Process Manager



Oracle BPEL Process Manager is loaded into Oracle JDeveloper. This extends the functionality of Oracle JDeveloper to enable you to model, edit, and design business processes using BPEL. Oracle JDeveloper provides a graphical and user-friendly way to build BPEL processes. What is unique about Oracle JDeveloper is that it uses BPEL as its native format. This means that processes built with Oracle JDeveloper are 100% portable. Oracle JDeveloper also enables you to view and modify the BPEL source without decreasing the usefulness of the tool. The core BPEL engine provides the most mature, scalable, and robust implementation of a BPEL server available today. Oracle BPEL Process Manager executes standard BPEL processes. Oracle BPEL Process Manager also provides a dehydration capability that enables the states of long-running flows to be automatically maintained in a database, thus enabling clustering for both fail over and scalability. Oracle BPEL Server uses an underlying J2EE application server, with support for most major commercial application servers and a bundled version available.

The built-in integration services enable you to use advanced connectivity and transformation capabilities of standard BPEL processes. These capabilities include support for XSLT and XQuery transformation, and bindings to hundreds of legacy systems through Java connector architecture (JCA) adapters and native protocols. A human task service is provided as a built-in BPEL service to enable the integration of people and manual tasks into BPEL flows.

The extensible WSDL binding framework enables connectivity to protocols and message formats other than SOAP. Bindings are available for JMS, e-mail, JCA, HTTP GET, HTTP POST, and many other protocols enabling simple connectivity to hundreds of back-end systems.

Oracle Enterprise Manager 10g BPEL Control provides a mature Web-based interface for management, administration, and debugging of processes deployed to Oracle BPEL Server. Audit trails and process history and reporting information are automatically maintained and available through both Oracle Enterprise Manager 10g BPEL Control and a Java API.

How Is Oracle BPEL Process Manager Different?

Oracle BPEL Process Manager provides the following key differences:

- The power of an open standard

  By capturing your business processes in BPEL, you protect your intellectual property and investments while avoiding vendor lock-in. BPEL is to business process management what SQL is to data management.

- Unparalleled visibility and administration

  Oracle Enterprise Manager 10g BPEL Control reduces the cost and complexity of deploying and managing your business processes. Visually monitor the execution of each BPEL process, drill down into the audit trail and view the details of each conversation, or debug a running flow against its BPEL implementation.

- Open and flexible binding framework

  You can orchestrate XML Web services, and also Java/J2EE components, portals, JCA interfaces, and Java message service (JMS) destinations. You can tie into back-end systems. You can use your Java skills and application server investments.

Oracle BPEL Process Manager Feature Summary

Oracle BPEL Process Manager includes the following features:

- Oracle JDeveloper
  - Native BPEL support
  - Drag-and-drop process modeler
  - Universal description, discover, and integration (UDDI) and Web services inspection language (WSIL) service browser
  - Visual XPath editor
  - One-click build and deploy
- Oracle Enterprise Manager 10g BPEL Control
  - Visual monitoring
  - Auditing
  - BPEL debugging
  - Process versioning
  - In-flight administration
  - Process performance monitoring statistics
  - Partitioning/domains
- Built-in integration services
  - Java embedding
  - E-mail and JMS messaging services

  - Extensible style sheet language transformation (XSLT) and XQuery transformation services
  - Workflow tasks and portal integration
  - Extensible WSIF binding framework
- Oracle BPEL Server
  - Comprehensive BPEL version 1.1
  - Synchronous and asynchronous messaging
  - Context dehydration
  - Advanced exception management
  - Side-by-side versioning
  - Large XML documents

Installing Oracle Application Server SOA Suite and Oracle JDeveloper

This section describes how to install Oracle Application Server SOA Suite and Oracle JDeveloper and configure Oracle BPEL Process Manager to work with Oracle JDeveloper.

This section contains the following topics:

- Installing Oracle Application Server SOA Suite
- Identifying Ports
- Installing Oracle JDeveloper
- Creating Oracle Application Server and Integration Server Connections

> **Note:** If you install the Oracle BPEL Process Manager for Developers installation type and attempt to use this tutorial, note that the ports (`9700`), Oracle BPEL Server startup commands, and Oracle Enterprise Manager 10g BPEL Control user name (`bpeladmin`) may be different from those listed in this tutorial.

Installing Oracle Application Server SOA Suite

1. See the *Oracle Application Server Installation Guide 10g Release 3 (10.1.3.1)* for your operating system for instructions on installing Oracle Application Server SOA Suite.
2. Select **Basic Install** on the Oracle Application Server SOA Suite 10.1.3.1.0 Installation window.

> **Note:** If you want to use Oracle BPEL Process Manager samples, enter **welcome1** in the **AS Administrator Password** field of the Oracle Application Server SOA Suite 10.1.3.1.0 Installation window for the **oc4jadmin** user. If you enter a different password, some samples require modification in order to work.

3. Click **Install**.
4. Click **Exit** when installation completes.

Identifying Ports

Identify several port numbers that were automatically configured during installation. You use these port numbers when creating application server and integration server connections to Oracle BPEL Server.

1. Use a text editor to open the `*SOA_Oracle_Home*\bpel\utilities\ant-orabpel.properties` file.
2. Make note of the ports being used by the following properties:
   - `http.port`
   - `opmn.requestport`

   These port values are required later when you create connections in "Creating Oracle Application Server and Integration Server Connections" on page 1-6.

Installing Oracle JDeveloper

You must separately install Oracle JDeveloper 10.1.3.1 Studio.

1. Unzip Oracle JDeveloper into a location on your host.
2. Start Oracle JDeveloper by clicking `*JDev_Oracle_Home*\JDev\bin\jdev.exe` or creating a shortcut.

> **Note:** Do *not* install Oracle JDeveloper into a directory path that includes a space (for example, `C:\Program Files\JDev`). If you do, you receive an error when you compile BPEL processes.

Creating Oracle Application Server and Integration Server Connections

1. Select **Connection Navigator** from the **View** main menu in Oracle JDeveloper.
2. Right-click **Application Server**.
3. Select **New Application Server Connection**.
4. Click **Next** on the **Welcome** page.
5. Provide a meaningful name for connecting to the application server. Remember this name, as you select it when creating an integration server connection in Step 19 on page 1-7.
6. Select **Oracle Application Server 10g 10.1.3** from the **Connection Type** list.
7. Click **Next**.
8. Enter the following details:

Field	Value
Username	**oc4jadmin**
Password	Enter the same administrator password you entered in "Installing Oracle Application Server SOA Suite" on page 1-5.
Deploy Password	Select this check box; otherwise, you must authenticate each time you deploy a process.

9. Click **Next**.
10. Enter the following details:

Field	Value
Host Name	**localhost**
OPMN Port	Enter the same port value used by the `opmn.requestport` property that you identified in Step 2 of "Identifying Ports" on page 1-6. This is likely port number `6003`.
Group/OC4J Instance Name	**home**
Single Instance	Select this radio button.
Group	Leave this radio button unselected.

11. Click **Next**.
12. Test the connection by clicking **Test Connection**. If the connection is successful, the following message appears:

    ```
    Success!
    ```

13. Click **Finish**.
14. Right-click **Integration Server** in the **Connection Navigator**.
15. Select **New Integration Server Connection**.

    You use this integration server connection to deploy all your BPEL processes.
16. Click **Next** on the **Welcome** page.
17. Provide a meaningful name for connecting to the server.
18. Click **Next**.
19. Select the application server connection name you specified in Step 5 on page 1-6 from the **Application Server** list.
20. Enter the following details:

Field	Value
Host Name	localhost
Port	Enter the same port value used by the `http.port` property that you identified in Step 2 of "Identifying Ports" on page 1-6. This is likely port number `8888`.
Add host name to the list of proxy exceptions	Select this check box to automatically add this host to the list of proxy exceptions shown in the **Exceptions** field under **Tools** > **Preferences** > **Web Browser and Proxy**. The first time it is started, Oracle JDeveloper by default picks up these proxy exceptions that are set for your browser.

21. Click **Next**.
22. Test the connection by clicking **Test Connection**. If the connection is successful, the following message appears:

    ```
    Application Server: OK
    BPEL Process Manager Server: OK
    ESB Server: OK
    ```

23. Click **Finish**.

Starting and Stopping Oracle BPEL Process Manager Components

Follow the instructions in Table 1–1 to start and stop Oracle BPEL Process Manager components.

Table 1–1 Starting and Stopping Oracle BPEL Process Manager Components

To Access The...	On Windows...	On UNIX...

Oracle BPEL Server	Select **Start** > **All Programs** > **Oracle -** ***Oracle_Home*** > **Start SOA suite**	To start Oracle BPEL Server:
		From `$ORACLE_HOME/bpel/bin`:
		`startorabpel.sh`
		To stop Oracle BPEL Server:
		From `$ORACLE_HOME/bpel/bin:`
		`shutdownorabpel.sh`

Oracle JDeveloper	See Step 2 of "Installing Oracle JDeveloper" on page 1-6	`$ORACLE_HOME/jdev/bin/jdev`
Oracle Enterprise Manager 10g BPEL Control	You must first start Oracle BPEL Server. To start Oracle Enterprise Manager 10g BPEL Control: 1. Select **Start** > **All Programs** > **Oracle -** ***Oracle_Home*** > **Oracle BPEL Process Manager** > **BPEL Control** You can also start Oracle Enterprise Manager 10g BPEL Control using the URL for your installation, which can found in *`SOA_Oracle_Home`*`\install\bpelsetupinfo.txt`.	First start Oracle BPEL Server. To start Oracle Enterprise Manager 10g BPEL Control: - Log on to the URL for your installation, which can found in `bpelsetupinfo.txt`.

Developer Prompt	Select **Start** > **All Programs** > **Oracle -** ***Oracle_Home*** > **Oracle BPEL Process Manager** > **Developer Prompt** to open up a command prompt at the *`SOA_Oracle_Home`*`\bpel\samples` directory. This enables you to easily access demonstrations and start any required Web services.	Setting Developer Prompt in Bourne shell:
		`$ ORACLE_HOME=/home/oracle/installs/midtier`
		`$ export ORACLE_HOME`
		`$ PATH=$ORACLE_HOME/bpel/bin:$PATH`
		`$ export PATH`

Oracle BPEL Process Manager Samples and Tutorials	For details about BPEL samples and additional tutorials available for use: Select **Start** > **All Programs** > **Oracle -** ***Oracle_Home*** > **Oracle BPEL Process Manager** > **Getting Started with Samples**	Log into the following URL: `$ORACLE_HOME/bpel/samples/sampleshome.html`
Sample Worklist Application	To access the login window for Oracle BPEL Worklist Application: Select **Start** > **All Programs** > **Oracle -** ***Oracle_Home*** > **Oracle BPEL Process Manager** > **Sample Worklist Application** You may also start Oracle BPEL Worklist Application using the URL for your installation, which is found in *`SOA_Oracle_Home`*`\install\bpelsetupinfo.txt`.	First start Oracle BPEL Server. To start Oracle BPEL Worklist Application: - Log on to the URL for your installation, which is found in `bpelsetupinfo.txt`.

Note: Always use the **Developer Prompt** to open an operating system command prompt in this guide. This sets all required paths. Opening an operating system command prompt in any other way is not supported.

Verifying the Hostname in Your Web Browser Preferences

The hostname of your computer was automatically added to the Oracle JDeveloper preference settings during installation. If the hostname does not appear in the preference settings, you can receive parsing errors when selecting a WSDL file on the Service Explorer window while creating a partner link.

1. Select **Preferences** from the **Tools** main menu.
2. Click **Web Browser and Proxy**.
3. Verify that your hostname is in the **Exceptions** field. For example, if your hostname is `myhost-pc`:

   ```
   us.acme.com|*.us.acme.com|127.0.0.1|myhost-pc|localhost
   ```

4. Ensure also that `localhost` appears in the **Exceptions** field.
5. Click **OK**.

   You are now ready to design your BPEL process.

Understanding Namespaces and Namespace URIs

As you use Oracle JDeveloper, namespace values such as **ns1:**, **ns2:**, or **client:** can display in an **Expression** field or **Variables** navigation tree when creating copy operations in **Assign** activities, or in fields of other activities.

Oracle BPEL Process Manager attempts to reduce your interaction with namespaces. When a namespace appears in an **Expression** field or **Variables** navigation tree, accept the namespace prefixes that display when filling in the parts of your query or expression.

XML namespaces provide a method for distinguishing between duplicate element type and attribute names. Such duplication can occur, for example, in an XSLT style sheet or document that contains element types and attributes from two different DTDs. A `<refuse>` element means something entirely different for a sanitation agency than a `<refuse>` element for a loan processing agency. For this reason, they are distinguished by different namespace prefixes, such as **ns1:**, **ns2:**, or **client:**.

The URIs provide a system for creating unique identifiers.

2 Credit Flow Tutorial

This tutorial describes how to use Oracle JDeveloper to design, deploy, and test your first BPEL process.

This tutorial contains these topics:

- Credit Flow Introduction
- Using the Tutorial

Credit Flow Introduction

This tutorial teaches you how to use Oracle JDeveloper to build, deploy, and test your first BPEL process. The process is a flow that you create to call a Credit Rating service. When you run this process, you enter your social security number into an HTML user interface. The Credit Rating service takes your number and returns a credit rating. Creating this process is intended to be the first step toward building a more sophisticated application (like that shown in Chapter 3, "Reviewing a Loan Procurement BPEL Process").

Using the Tutorial

This tutorial contains these topics:

- Starting Oracle BPEL Server and Oracle JDeveloper
- Starting and Testing Your Service
- Creating an Application and a Project
- Viewing the WSDL File Source Code
- Viewing the BPEL File Source Code
- Editing the XSD File
- Validating Your Project During the Design Phase
- Creating and Configuring a Partner Link for the Credit Rating Service
- Validating, Compiling, and Deploying the Credit Flow Process
- Running the Credit Flow Process

Starting Oracle BPEL Server and Oracle JDeveloper

Ensure that Oracle JDeveloper and Oracle BPEL Server are started. See "Starting and Stopping Oracle BPEL Process Manager Components" on page 1-8 for instructions.

Starting and Testing Your Service

During this tutorial, the BPEL process that you design communicates with a Credit Rating service described in "Credit Flow Introduction" on page 2-1. You must first start the service and test that it is running.

1. Select **Start** > **All Programs** > **Oracle - *Oracle_Home*** > **Oracle BPEL Process Manager** > **Developer Prompt** to open up a command prompt at the *`SOA_Oracle_Home`*`\bpel\samples` directory.
2. Change directories to the `utils\CreditRatingService` subdirectory:

   ```
   cd utils\CreditRatingService
   ```

3. Enter the following command:

   ```
   ant
   ```

   This deploys and starts the Credit Rating service for using this tutorial. If successful, a message similar to the following appears at the end:

   ```
   BUILD SUCCESSFUL
   ```

4. Log in to Oracle Enterprise Manager 10g BPEL Control by selecting **Start** > **All Programs** > **Oracle - *Oracle_Home*** > **Oracle BPEL Process Manager** > **BPEL Control**.
5. Enter the following details to log in to Oracle Enterprise Manager 10g BPEL Control and click **Login**:

Field	Value
Username	`oc4jadmin`
Password	*`password`*

   Where *`password`* is the value you entered during installation in "Installing Oracle Application Server SOA Suite" on page 1-5.

   All services are running if the **CreditRatingService**, **TaskActionHandler**, and **TaskManager** services display in the Dashboard tab.

Creating an Application and a Project

You must create an application and a project to begin. The project automatically creates the basics for your BPEL process, including:

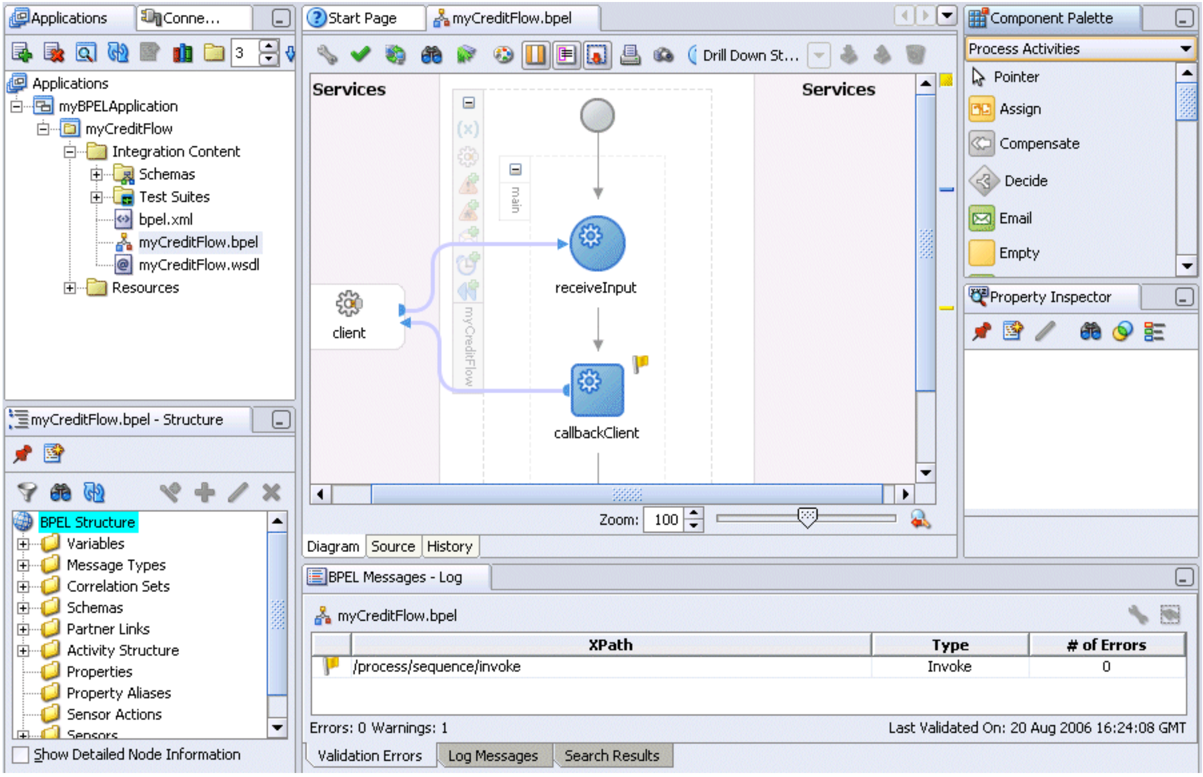
- BPEL process source (*`projectname`*`.bpel`)
- Web services description language (WSDL) client interface (*`projectname`*`.wsdl`)
- BPEL process deployment descriptor (`bpel.xml`)

Follow these instructions to create a new Credit Flow project.

> **Caution:** Do not include any special characters in the project name (such as periods). If you do include special characters, errors appear when you attempt to compile your project.

1. Return to Oracle JDeveloper.

2. Select **File** > **New** from the main menu.
3. Double-click **Application** in the **Items** window to display the Create Application window.
4. Enter **myBPELapplication** in the **Application Name** field and accept the default path in the **Directory Name** field.
5. Click **OK**.
6. Click **Cancel** on the Create Project window.
7. Right-click **myBPELapplication** in the **Application Navigator** section.
8. Select **New Project** to define a new BPEL process project.
9. Double-click **BPEL Process Project** in the **Items** window to display the BPEL Project Creation Wizard window. If this entry does not appear, first select **General** > **Projects**.
10. Enter **myCreditFlow** in the **Name** field. All other fields default to the correct values for creating an asynchronous BPEL process.
11. Click **Finish**. The BPEL process files are created in the *`JDev_Oracle_Home`*`\jdev\mywork\myBPELapplication\myCreditFlow\bpel` directory.
12. The following sections appear. If they do not appear, click **Diagram** in Oracle JDeveloper and double-click **myCreditFlow.bpel** in the **Application Navigator** > **Integration Content** section. See the *Oracle BPEL Process Manager Developer's Guide* for a description of Oracle JDeveloper sections.



myCreditFlow.bpel

XPath	Type	# of Errors
/process/sequence/invoke	Invoke	0

Errors: 0 Warnings: 1

Last Validated On: 20 Aug 2006 16:24:08 GMT

Viewing the WSDL File Source Code

You now review the sections of the WSDL file.

1. Double-click **myCreditFlow.wsdl** in the **Application Navigator** section, then click **Source** at the bottom of the window.
2. View the following section of code. Two port types are defined, each with a one-way operation. One operation initiates the asynchronous process. The other operation calls back the client with the asynchronous response.

```
<portType name="myCreditFlow">
   <operation name="initiate">
      <input message="client:myCreditFlowRequestMessage"/>
   </operation>
</portType>

<!-- portType implemented by the requester of myCreditFlow BPEL process
for asynchronous callback purposes
-->
<portType name="myCreditFlowCallback">
   <operation name="onResult">
      <input message="client:myCreditFlowResponseMessage"/>
   </operation>
</portType>
```

3. View the following section of code. The `partnerLinkType` for this asynchronous process has two roles. One role is for the service provider, and the other is for the requester.

```
<plnk:partnerLinkType name="myCreditFlow">
   <plnk:role name="myCreditFlowProvider">
      <plnk:portType name="client:myCreditFlow"/>
   </plnk:role>
   <plnk:role name="myCreditFlowRequester">
      <plnk:portType name="client:myCreditFlowCallback"/>
   </plnk:role>
</plnk:partnerLinkType>
```

Viewing the BPEL File Source Code

You now review the sections of the BPEL file.

1. Double-click **myCreditFlow.bpel**.
2. Click **Source** at the bottom of the designer window.
3. View the following section of code. The `partnerLink` created for the client interface includes two roles, `myRole` and `partnerRole`. An asynchronous BPEL process typically has two roles for the client interface: one for the flow itself, which exposes an input operation, and one for the client, which gets called back asynchronously.

```
<partnerLinks>
   <partnerLink name="client" partnerLinkType="client:myCreditFlow"
   myRole="myCreditFlowProvider" partnerRole="myCreditFlowRequester"/>
</partnerLinks>
```

4. View the following section of code. The `<receive>` activity in the main body of the process is followed by an `<invoke>` activity to perform an asynchronous callback to the requester. Note the difference between this and a synchronous process, which uses a `<reply>` activity to respond synchronously to the caller.

```
  <sequence name="main">

    <!-- Receive input from requestor.
```

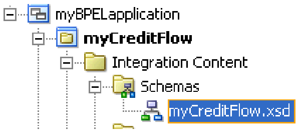
```
   Note: This maps to operation defined in myCreditFlow.wsdl
   -->
   <receive name="receiveInput" partnerLink="client"
portType="client:myCreditFlow" operation="initiate" variable="inputVariable"
createInstance="yes"/>

   <!-- Asynchronous callback to the requester.
    Note: the callback location and correlation id is transparently handled
    using WS-addressing.
    -->
   <invoke name="callbackClient" partnerLink="client"
portType="client:myCreditFlowCallback" operation="onResult"
inputVariable="outputVariable"/>
 </sequence>
</process>
```

5. Click **Diagram**. You are now ready to design your BPEL process.

Editing the XSD File

When you created the BPEL project, a default XSD file was created.

1. Expand **myCreditFlow** > **Integration Content** > **Schemas** in the **Application Navigator**.
2. Double-click **myCreditFlow.xsd**.



3. Click **Source**.

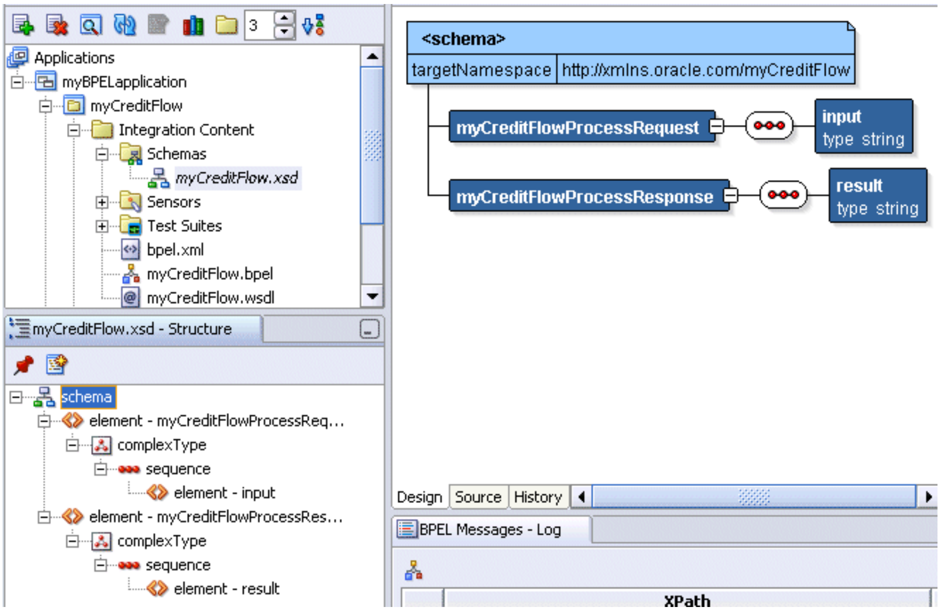
   A `myCreditFlowProcessRequest` element is accepted as input. A `myCreditFlowProcessResponse` element is returned as output.

```
<schema attributeFormDefault="unqualified"
      elementFormDefault="qualified"
      targetNamespace="http://xmlns.oracle.com/myCreditFlow"
      xmlns="http://www.w3.org/2001/XMLSchema">
      <element name="myCreditFlowProcessRequest">
            <complexType>
                  <sequence>
                        <element name="input" type="string"/>
                  </sequence>
            </complexType>
      </element>
      <element name="myCreditFlowProcessResponse">
            <complexType>
                  <sequence>
                        <element name="result" type="string"/>
                  </sequence>
            </complexType>
      </element>
</schema>
```

You now edit the input and output messages of the WSDL file. This WSDL file represents the user interface with which you interact when you run your BPEL process and request a credit rating in"Running the Credit Flow Process" on page 2-13.

4. Click **Design**.

   A graphical view of the XSD file displays in the design window. In the **Structure** section at the lower left, the elements of the XSD file display for editing when fully expanded.



5. Double-click the first **element - input** in the **Structure** window to display the element Properties window.

6. Change **input** to **ssn**.

7. Click **OK**.

8. Double-click the second **element - result** in the **Structure** window to display the element Properties window.

9. Change **result** to **creditRating**.

10. Click the **Advanced Properties** tab.

11. Go to the **type** section and note it says **string**.

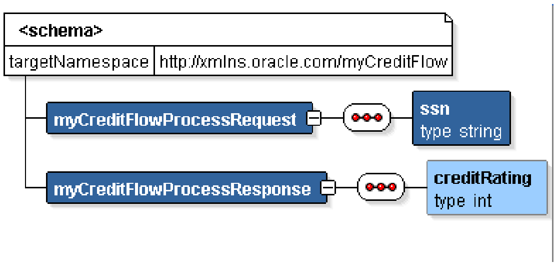
12. Click **string**.

13. Click **...** to the right of **string** to display the type window.

14. Replace **string** with **int** in the **Name** field as the output to be returned.

15. Click **OK** to close the type window and the element Properties window.

    Your changes display in the design view of the XSD file.


<schema>
targetNamespace
http://xmlns.oracle.com/myCreditFlow
myCreditFlowProcessRequest
ssn
type string
myCreditFlowProcessResponse
creditRating
type int

16. Select **Save** from the **File** main menu.

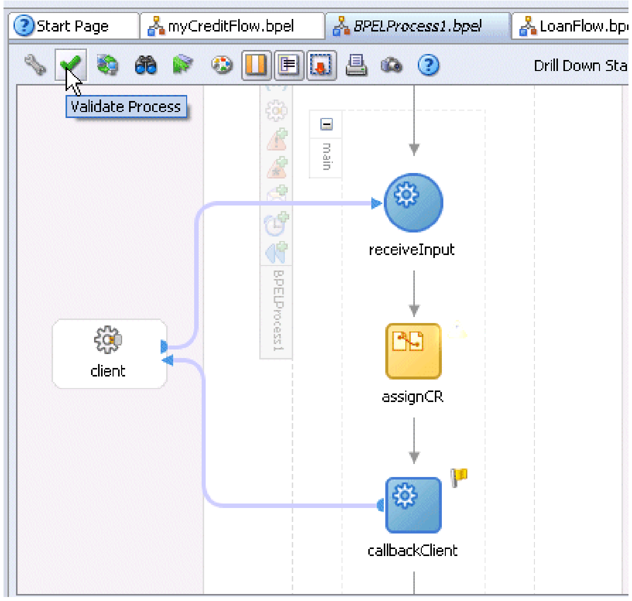
17. Close the XSD file by clicking the **x** in **myCreditFlow.xsd** at the top of the designer window.
18. Double-click **myCreditFlow.bpel** in the **Application Navigator** section.
19. Click **Diagram**.

Validating Your Project During the Design Phase

As you design your BPEL project, small yellow flags can display above activities. You may have correctly configured these activities; they simply may not yet have been validated.

1. Click **Validate** above the designer window to validate your process.

Figure 2–1 Validating a Process



If the activity correctly validates, the flag disappears.

2. If the yellow flag remains, you may have validation errors in activities or your process design is not yet complete.
3. Click the yellow flag above the activity to display details about the validation error and corrective actions to follow.

Note: When you deploy a process, as described in "Validating, Compiling, and Deploying the Credit Flow Process" on page 2-13, your process is automatically validated, compiled, and, if successful, deployed together.

Creating and Configuring a Partner Link for the Credit Rating Service

You now create and configure a partner link for the synchronous Credit Rating service.

This section contains these tasks:

- Creating a Credit Rating Service Partner Link
- Creating a Scope Activity

- Creating an Invoke Activity Inside the Scope Activity
- Creating an Initial Assign Activity Inside the Scope Activity
- Creating a Second Assign Activity Inside the Scope Activity

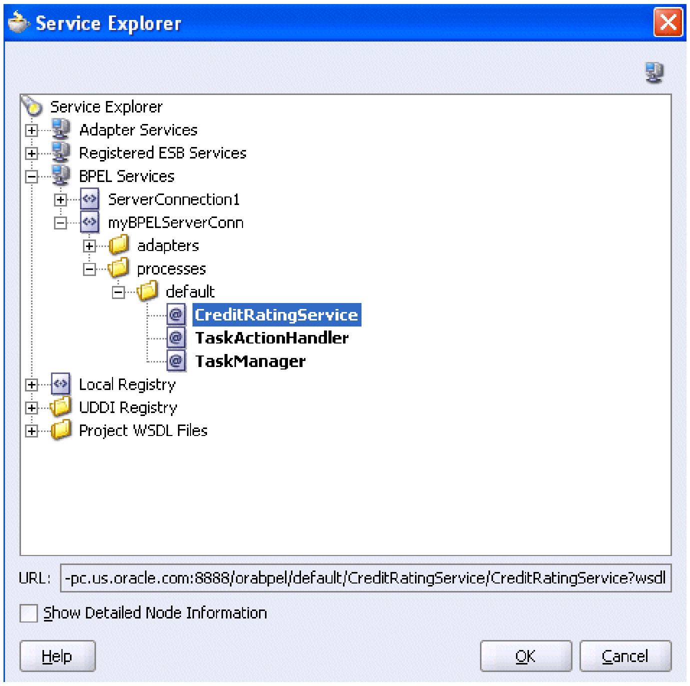
Creating a Credit Rating Service Partner Link

Summary: Partner links define the external services with which your BPEL process interacts. You must create a partner link for the Credit Rating service.

1. Ensure that **Services** is selected in the drop-down list of the **Component Palette** section in the upper right part of Oracle JDeveloper.
2. Drag and drop a **PartnerLink** activity under the **Services** section on the right side of Oracle JDeveloper.

   The Create Partner Link window appears.
3. Enter the following values to create a partner link for the Credit Rating service:

Note: For the **WSDL File** field below, click the **flashlight** (the second icon from the left named **Service Explorer**) to access the Service Explorer window shown below for automatically selecting the Credit Rating service deployed in "Starting and Testing Your Service" on page 2-2.



URL: -pc.us.oracle.com:8888/orabpel/default/CreditRatingService/CreditRatingService?wsdl

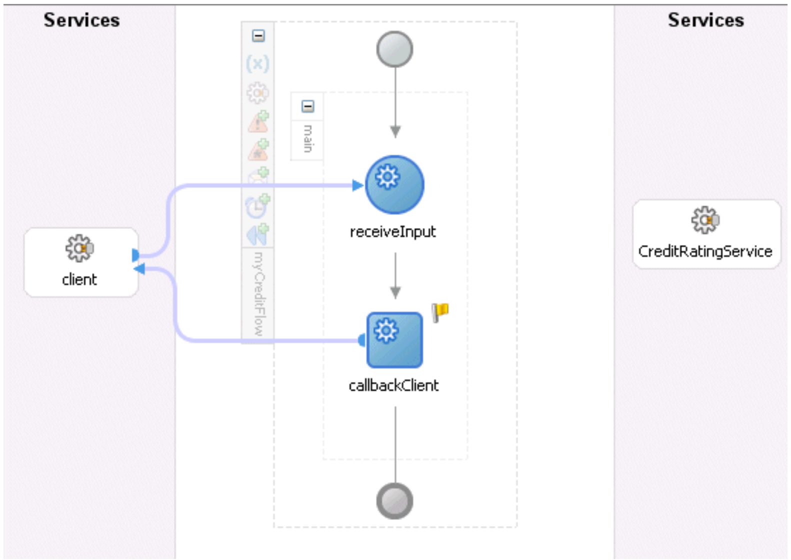
□ Show Detailed Node Information

Field	Value
Name	CreditRatingService

Field	Value
WSDL File	Access this URL by clicking the **Service Explorer flashlight** icon and expanding and selecting **BPEL Services** > ***my_integration_server_connection*** > **processes** > **default** > **CreditRatingService**. where ***my_integration_server_connection*** is the server connection name you specified in Step 17 on page 1-7. **http://localhost:8888/orabpel/default/CreditRatingService/CreditRatingService?wsdl** **See Also:** "Verifying the Hostname in Your Web Browser Preferences" on page 1-9 if you receive a parsing error when attempting to add a WSDL file in the Service Explorer window.
Partner Link Type	**CreditRatingService**
Partner Role	**CreditRatingServiceProvider**
My Role	Leave unspecified. Because this is a synchronous partner link, no role is required.

4. Click **OK**.

   The **CreditRatingService** partner link appears on the right side of the design window.



5. Select **Save** from the **File** main menu.

Creating a Scope Activity

Summary: You first create a **Scope** activity in this section. A **Scope** activity consists of a collection of activities that can have their own local variables, fault handlers, and so on. A **Scope** activity is analogous to a { } block in a programming language.

Within this **Scope** activity, the client's credit rating background is identified.

1. Go to the **Component Palette** section and select **Process Activities**.

2. Drag and drop a **Scope** activity between the **receiveInput** and **callbackClient** activities.
3. Double-click the **Scope** activity to display the Scope window.
4. Enter **GetCreditRating** in the **Name** field of the **General** tab.
5. Click **OK**.
6. Select **Save** from the **File** main menu.

Creating an Invoke Activity Inside the Scope Activity

Summary: You create an **Invoke** activity in this section. An **Invoke** activity enables you to specify the operation you want to invoke for the service (identified by its partner link).

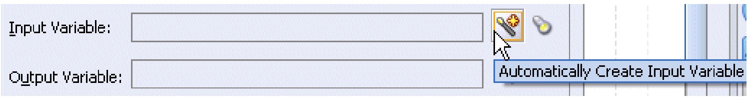
This **Invoke** activity provides the initial interaction operation between the client and the Credit Rating service.

1. Click the **+** sign to expand the **GetCreditRating Scope** activity.
2. Drag and drop an **Invoke** activity from the **Component Palette** section into the **GetCreditRating Scope** activity.
3. Double-click the **Invoke** icon to display the Invoke window.
4. Enter the following details:

Field	Value
Name	**invokeCRS**
Partner Link	**CreditRatingService**

The **Operation** (**process**) field is automatically filled in.

5. Click the first icon to the right of the **Input Variable** field. This is the automatic variable creation icon.



6. Click **OK** on the Create Variable window that appears.

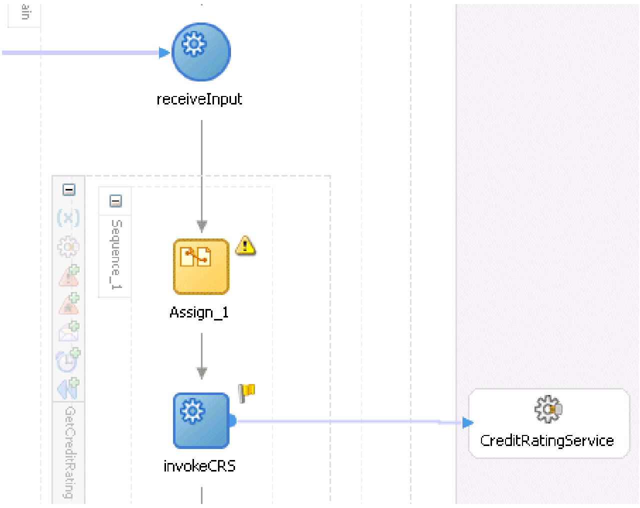
   A variable named **invokeCRS_process_InputVariable** is automatically created in the **Input Variable** field. This variable is automatically assigned a message type of **CreditRatingServiceRequestMessage**.
7. Click the first icon to the right of the **Output Variable** field.
8. Click **OK** on the Create Variable window that appears.

   A variable named **invokeCRS_process_OutputVariable** is automatically created in the **Output Variable** field. This variable is automatically assigned a message type of **CreditRatingServiceResponseMessage**.
9. Click **OK**.
10. Select **Save** from the **File** main menu.

Creating an Initial Assign Activity Inside the Scope Activity

> **Summary:** In this section you create the first of two **Assign** activities in this tutorial. An **Assign** activity provides a method for simple data manipulation, such as copying the contents of one variable to another.
>
> This **Assign** activity takes the client's social security number as input and contacts the Credit Rating service to assign a credit rating.

1. Drag and drop an **Assign** activity from the **Component Palette** section to above the **invokeCRS Invoke** activity.



2. Double-click the **Assign** icon to display the Assign window.
3. Enter **assignSSN** in the **Name** field of the **General** tab.
4. Click **Apply**.
5. Click the **Copy Operation** tab.
6. Click **Create** and select **Copy Operation** to display the Create Copy Operation window.
7. Enter the following values:

Field	Value
From	
■ **Type**	**Variable**
■ **Variables**	Expand and select **Variables** > **inputVariable** > **payload** > **client**:**myCreditFlowProcessRequest** > **client:ssn** **Note:** The namespace number values (for example, **ns1**, **ns2**) can vary. Use the namespace values that automatically appear.
To	
■ **Type**	**Variable**
■ **Variables**	Expand and select **Variables** > **invokeCRS_process_InputVariable** > **payload** > **ns1:ssn**

8. Click **OK** to close the Create Copy Operation window and the Assign window.

9. Select **Save** from the **File** main menu.

Creating a Second Assign Activity Inside the Scope Activity

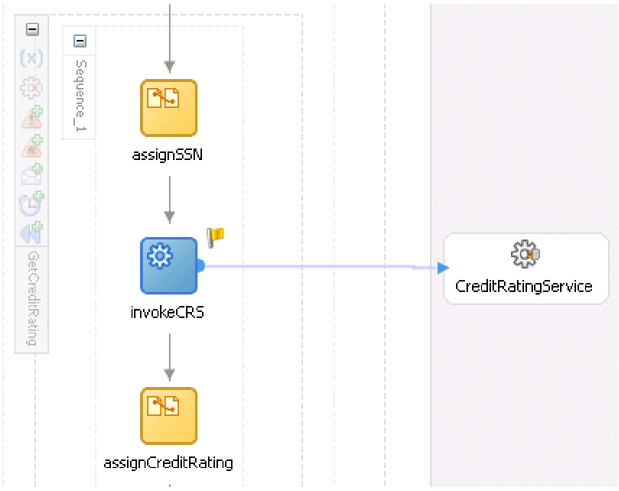
> **Summary:** This **Assign** activity returns details about the client's assigned credit rating.

1. Drag and drop a second **Assign** activity from the **Component Palette** section to below the **invokeCRS Invoke** activity within the **GetCreditRating Scope** activity.
2. Double-click the **Assign** icon to display the Assign window.
3. Enter **assignCreditRating** in the **Name** field of the **General** tab.
4. Click **Apply**.
5. Click the **Copy Operation** tab.
6. Click **Create** and select **Copy Operation** to display the Create Copy Operation window.
7. Enter the following values:

Field	Value
From	
▪ **Type**	**Variable**
▪ **Variables**	Expand and select **Variables > invokeCRS_process_OutputVariable > payload** > **ns1:rating** **Note:** The namespace number values (for example, **ns1**, **ns2**) can vary. Use the namespace values that automatically appear.
To	
▪ **Type**	**Variable**
▪ **Variables**	Expand and select **Variables** > **outputVariable** > **payload** > **client:myCreditFlowProcessResponse** > **client:creditRating**

8. Click **OK** to close the Create Copy Operation window and the Assign window.

   The **Scope** activity looks as follows:



9. Select **Save** from the **File** main menu.

   You have now completed the Credit Rating service design and are ready to proceed to the validation, compilation, and deployment tasks described in the following section.

Validating, Compiling, and Deploying the Credit Flow Process

You are now ready to deploy your BPEL process.

1. Go to the **Application Navigator** section.
2. Right-click **myCreditFlow**.
3. Select **Deploy** > ***my_integration_server_connection*** > **Deploy to default domain**.

   where ***my_integration_server_connection*** is the integration server connection name you specified in Step 17 on page 1-7.
4. Click **OK**.

   This compiles the BPEL process. Review the bottom of the window for any errors. If there are no errors, the following message appears:

   ```
   BUILD SUCCESSFUL
   ```
5. Check for errors by clicking the buttons at the bottom of the window. Errors that do not display under the **Messages** button can sometimes appear under the **Apache Ant** button.

   If there are no errors, deployment was successful. If there are errors, double-click the error to display details about the type and location of the error.
6. Make corrections and deploy again.

Running the Credit Flow Process

You are now ready to begin the process of applying for and receiving a loan offer.

1. Log in to Oracle Enterprise Manager 10g BPEL Control by selecting **Start** > **All Programs** > **Oracle - *Oracle_Home*** > **Oracle BPEL Process Manager** > **BPEL Control**.
2. Enter the following details to log in to Oracle Enterprise Manager 10g BPEL Control:

Field	Value
Username	`oc4jadmin`
Password	*`password`*

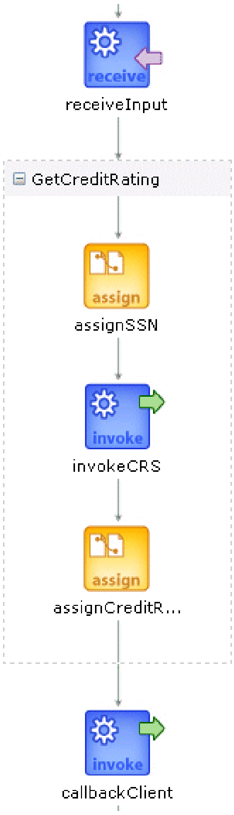
   The **Dashboard** tab of Oracle Enterprise Manager 10g BPEL Control appears.
3. Click **myCreditFlow** in the **Deployed BPEL Processes** list.
4. Enter a nine-digit integer value that does *not* begin with zero in the **ssn** field of the HTML Form and click **Post XML Message**. Note that the name **ssn** that displays in the user interface is the value you entered in "Editing the XSD File" on page 2-5.

   The **BPEL Processes** tab displays a message similar to the following:

   ```
   Test Instance Initiated
   ```

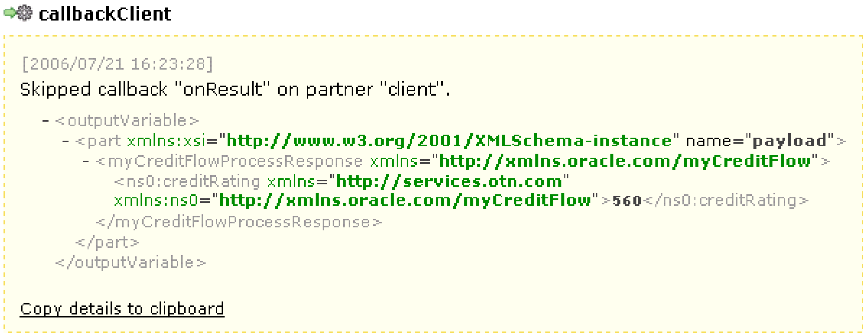
```
Instance '39e706a46ad531be:858bf1:fcc240f310:-7ffc' is being processed
asynchronously.
```

5. Click **Visual Flow** to view a visual audit trail of the current state of the process instance.



An audit trail displaying the current state of the process appears. It indicates that you have successfully invoked the Credit Rating service.

6. Click **callbackClient** in the audit trail to see the results of the loan offer, including the credit rating returned (**560**) by the Credit Rating service to the client.



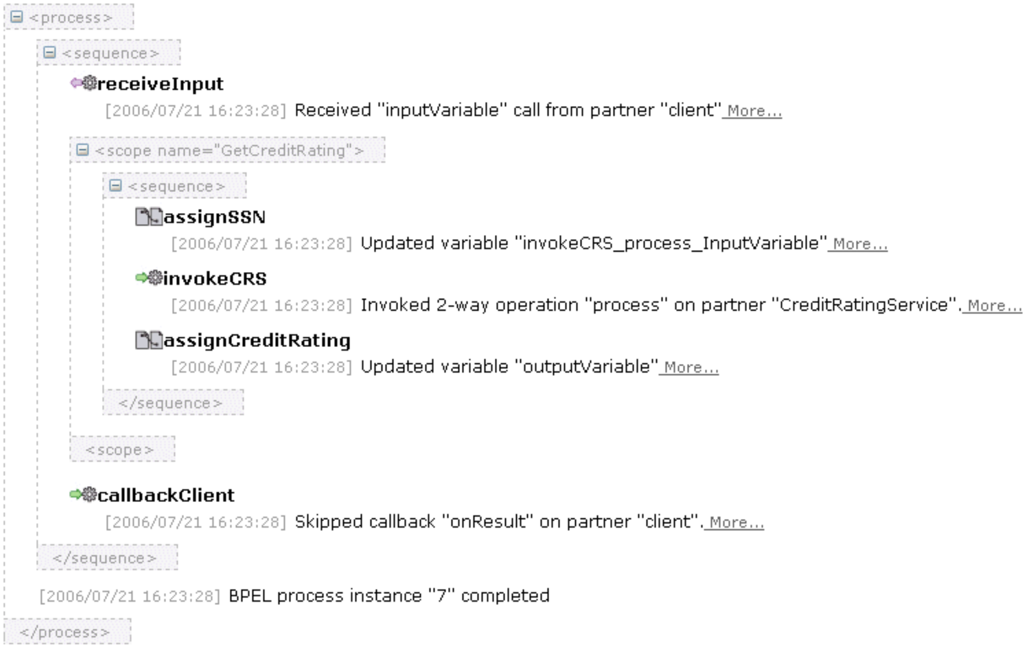
callbackClient

[2006/07/21 16:23:28]
Skipped callback "onResult" on partner "client".

```
 - <outputVariable>
   - <part xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="payload">
     - <myCreditFlowProcessResponse xmlns="http://xmlns.oracle.com/myCreditFlow">
         <ns0:creditRating xmlns="http://services.otn.com"
         xmlns:ns0="http://xmlns.oracle.com/myCreditFlow">560</ns0:creditRating>
       </myCreditFlowProcessResponse>
     </part>
   </outputVariable>
```

Copy details to clipboard

7. Click the **Audit** link at the top to observe additional information.



```
⊟ <process>
    ⊟ <sequence>
        receiveInput
            [2006/07/21 16:23:28] Received "inputVariable" call from partner "client" More...
        ⊟ <scope name="GetCreditRating">
            ⊟ <sequence>
                assignSSN
                    [2006/07/21 16:23:28] Updated variable "invokeCRS_process_InputVariable" More...
                invokeCRS
                    [2006/07/21 16:23:28] Invoked 2-way operation "process" on partner "CreditRatingService". More...
                assignCreditRating
                    [2006/07/21 16:23:28] Updated variable "outputVariable" More...
            </sequence>
        <scope>
        callbackClient
            [2006/07/21 16:23:28] Skipped callback "onResult" on partner "client". More...
    </sequence>
    [2006/07/21 16:23:28] BPEL process instance "7" completed
</process>
```

3

Reviewing a Loan Procurement BPEL Process

This chapter examines how to execute a more sophisticated BPEL process than the one you created in Chapter 2, "Credit Flow Tutorial". Fault-handling, interactions with asynchronous services, user and manual tasks, Java and J2EE code integrations, and custom user interfaces and portals are all key requirements for many business flows. This chapter describes how you build, test, debug, and manage a BPEL process that implements these requirements.

This chapter contains the following topics:

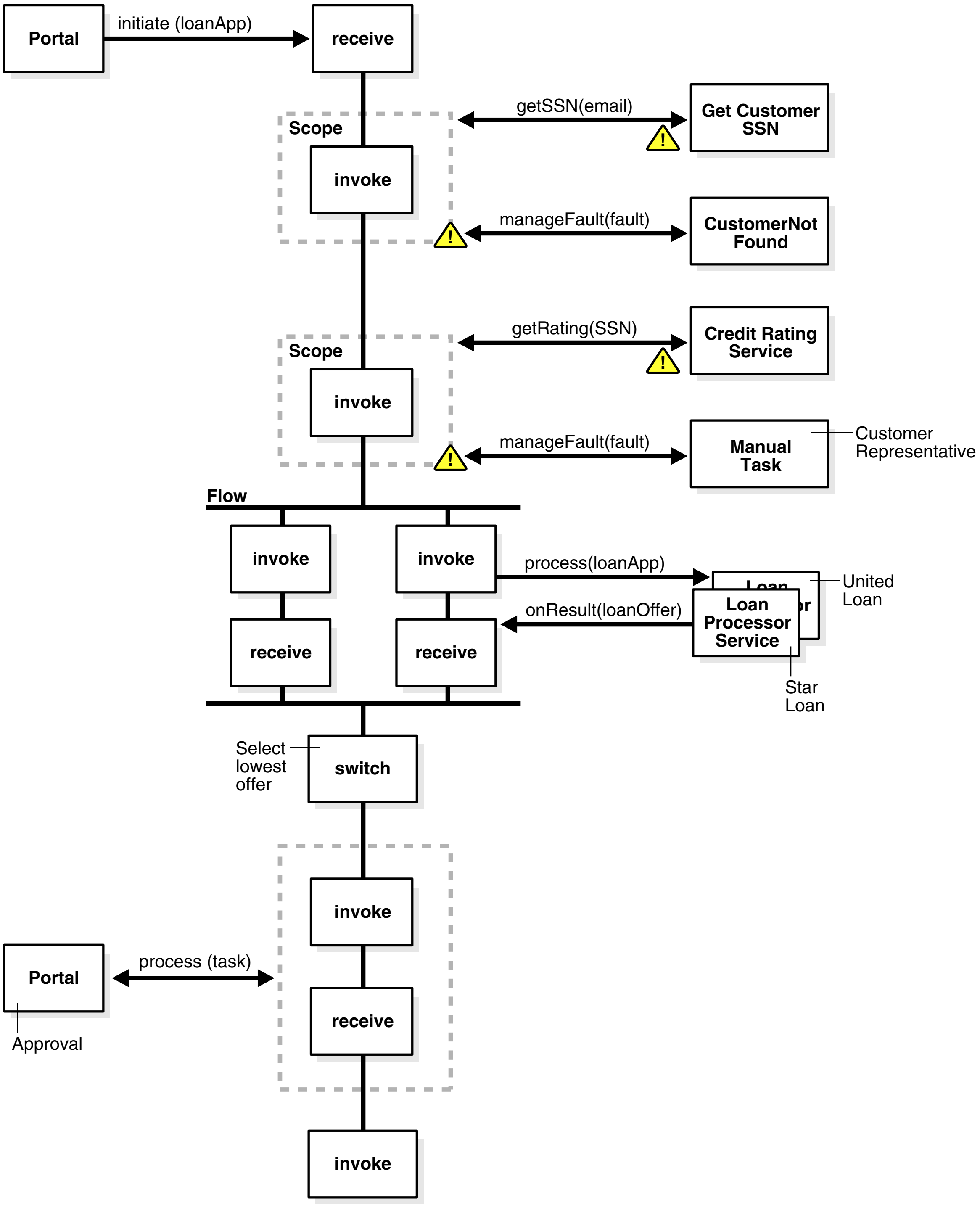
LoanFlowPlus Process Overview

This chapter describes the LoanFlowPlus process, which is shipped with the Oracle BPEL Process Manager samples and available in the following directory after installation:

```
SOA_Oracle_Home\bpel\samples\demos\LoanDemoPlus
```

Figure 3–1 shows the requirements implemented by LoanFlowPlus.

Figure 3–1 LoanFlowPlus Overview



This process is a long-running BPEL flow that takes a loan application document as input and asynchronously returns a selected and approved loan offer.

At the beginning of process execution, this flow gets a social security number for the customer from an Enterprise Java Bean (EJB) (accessed as a service through the native EJB Web Services Invocation Framework (WSIF) binding) and then goes to the synchronous Credit Rating service you saw in Chapter 2, "Credit Flow Tutorial" to request a credit rating for the customer.

The process is coded to handle the negative credit faults that can be thrown by the credit rating service if a negative credit event is seen for the customer (for example, a bankruptcy). If such an exception is thrown, the LoanFlowPlus process starts a human task for a customer service representative to manually handle the exception and then waits for the task to complete. The customer service representative can use a dashboard user interface to see their tasks and specify a credit rating or select to cancel each application.

Once a credit rating is supplied, either through manual or automated activities, the process goes into automated processing. It then sends the completed loan application to two loan processors. Each processor can take an arbitrary amount of time before returning a loan offer. Since the loan processors can be long-running, they are implemented as asynchronous services and the BPEL process invokes them in parallel.

In the case of this BPEL flow, it waits for both loan offers to be received before using simple branching logic to select the best offer (the one with the lowest interest rate). Finally, another human task enables the customer to review and approve the selected loan offer. Once this task has completed, the selected and approved offer is returned as the result of the flow.

Building and Reviewing the Process in Oracle JDeveloper

This section describes how to build and deploy the external services used by your BPEL process and how to start and view your process in Oracle JDeveloper.

Starting and Testing Your Service

As in Chapter 2, "Credit Flow Tutorial", the LoanFlowPlus process is dependent on several external services that you can quickly build and deploy using the `ant` command. To do this:

1. Select **Start** > **All Programs** > **Oracle - *Oracle_Home*** > **Oracle BPEL Process Manager** > **Developer Prompt** to open up a command prompt at the *`SOA_Oracle_Home`*`\bpel\samples` directory.
2. Change directories to `demos\LoanDemoPlus`:

   ```
   cd demos\LoanDemoPlus
   ```

3. Enter the following command:

   ```
   ant
   ```

   This builds and deploys the external loan services required for the LoanFlowPlus process, including the process itself.

Opening the LoanFlowPlus Process in Oracle JDeveloper

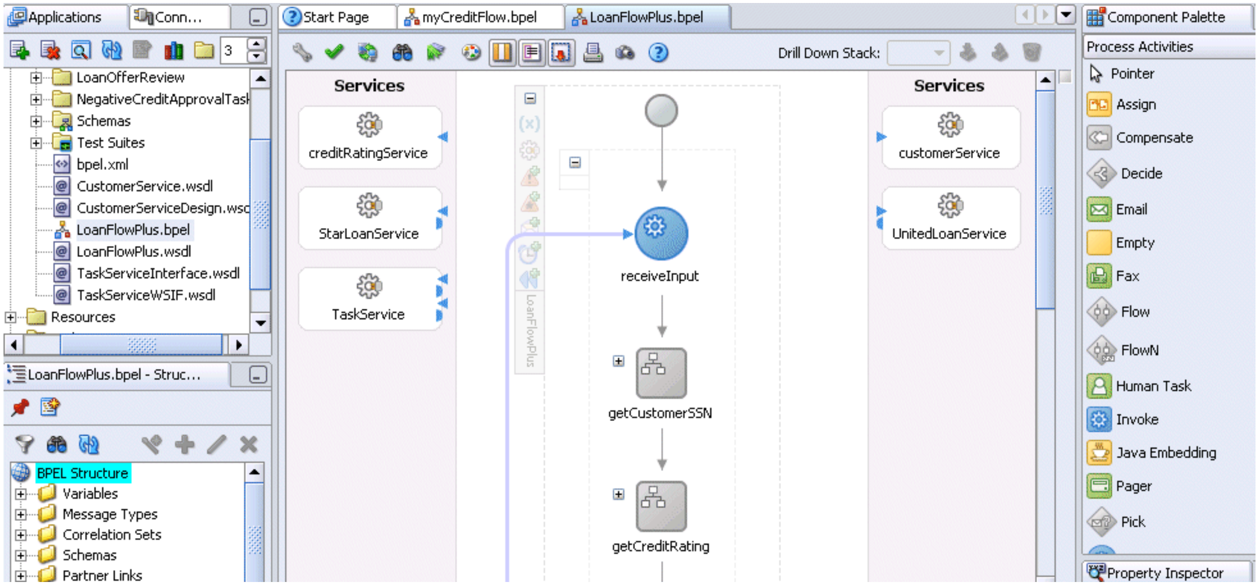
1. Return to Oracle JDeveloper.
2. Highlight **myBPELapplication** in the **Application Navigator**.
3. Select **Open** from the **File** main menu.

   The Open window appears.
4. Go to the *`SOA_Oracle_Home`*`\bpel\samples\demos\LoanDemoPlus\LoanFlowPlus` directory.
5. Select `LoanFlowPlus.jpr` and click **Open**.

This is the **LoanFlowPlus** project file, which now displays in Oracle JDeveloper.

6. Open **LoanFlowPlus** > **Integration Content** in the **Application Navigator**.
7. Double-click **LoanFlowPlus.bpel**.

   This displays the **Diagram** view of your process.



You see that **creditRatingService** partner link is the same synchronous service used in Chapter 2, "Credit Flow Tutorial". However, the **StarLoanService** and **UnitedLoanService** partner links are asynchronous, supporting a one-way initiate operation and an asynchronous onResult callback that is called after a loan offer is ready from each service. That can take anywhere from a few seconds to a few days or more. As with your previous process, the client interface to the **LoanFlowPlus** process is also asynchronous.

8. View a visual representation in the center swim-lane of the entire process. Oracle JDeveloper uses standard BPEL source as its native format. This means you can always select the BPEL **Source** view to see the underlying BPEL source code, make changes in either view and see those changes represented immediately in the alternate view, or do both. Here you see that the **Diagram** view shows that the process gets initiated, then encapsulates the logic to get the social security number (including the exception handling code) in a BPEL scope, and so on.

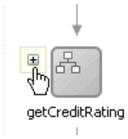
   In the following sections of this chapter, you review the code that implements this process.

Managing Faults

The **LoanFlowPlus** process uses BPEL fault handling to catch and manage exceptions thrown by the social security number and credit rating services. Fault handlers are associated with a particular scope activity and faults encountered within a scope and not handled by that scope are sent up to the enclosing scope. This is just like `try-catch` in Java.

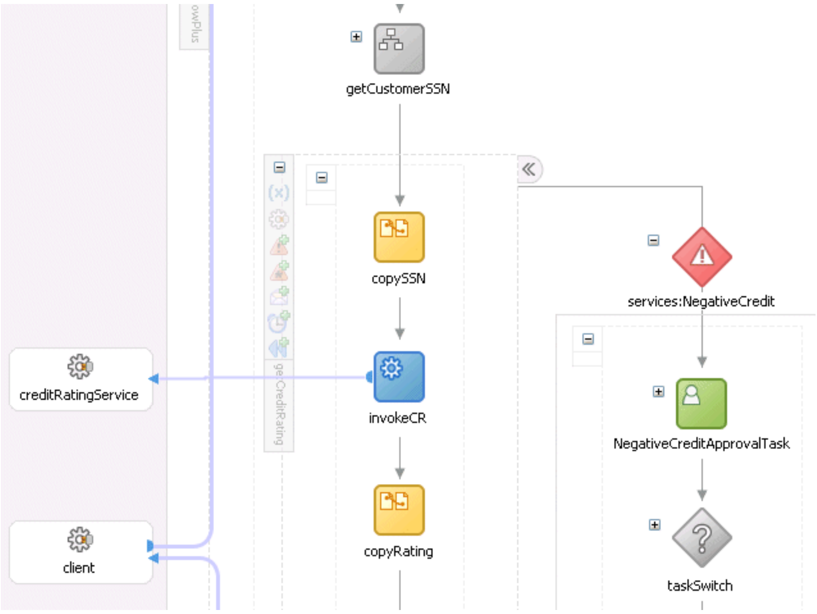
Follow these instructions to view the **NegativeCredit** fault handler defined for the **getCreditRating**.

1. Click the **+** sign to expand the **getCreditRating** scope.



This opens the view of that scope.

2. View the main flow running top-to-bottom within the scope. You also see a secondary flow as an extension to the right. The top of this flow contains the error activity **services:NegativeCredit**.



3. Open the scope activity within the exception flow and view the steps that are executed as the logic for handling this fault. Any activity in BPEL (that can be arbitrarily complex, short or long running, and so on) can be used in a fault handler. In this case, a human task is defined for a customer service representative to manually handle the exception. Faults can also be handled through automated logic or rethrown.

   A portion of the code that implements fault handling is shown:

```
<scope name="getCreditRating" variableAccessSerializable="no">
   <variables>
      . . .
      . . .
   </variables>
<!-- Watch for faults (exceptions) being thrown from creditRatingService -->
   <faultHandlers>
       <catch faultName="services:NegativeCredit" faultVariable="crError">
           . . .
       </catch>
   </faultHandlers>
   <sequence>
   . . .
   . . .
   <!-- Invoke the CreditRating Service, the URL of this service's
               WSDL is specified in the deployment descriptor -->
       <invoke name="invokeCR" partnerLink="creditRatingService"
               portType="services:CreditRatingService" operation="process"
```

```
                    inputVariable="crInput" outputVariable="crOutput"/>
     . . .
     </sequence>
</scope>
```

See Also: The technotes for building resilient BPEL processes and running crash tests on Oracle BPEL Server at the following URL for more information about exception management, reliability, and crash testing:

http://www.oracle.com/technology/bpel

Interacting with an Asynchronous Loan Processor Service

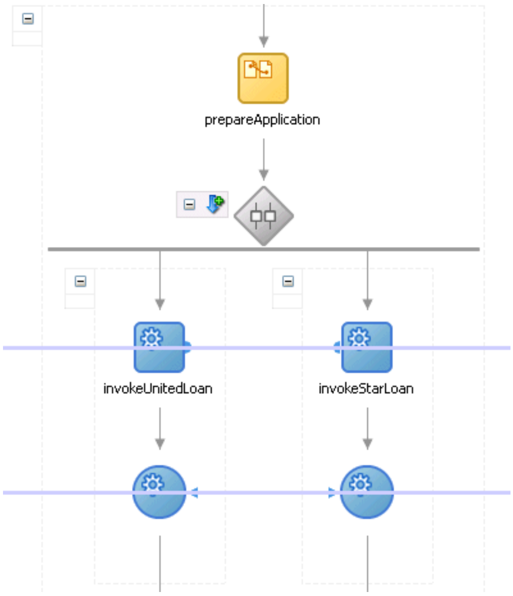
Once the loan application document has been filled in, it is sent to the two asynchronous loan processor services. In this section, we examine what is required to interact with an asynchronous service in a BPEL process. As mentioned previously, BPEL has support for asynchronous activities at its core. Oracle BPEL Process Manager supports a dehydration capability. This means that flows are reliably and efficiently persisted to a data store along with all their current state information whenever they wait for asynchronous events.

BPEL also enables support for several standard methods of correlating asynchronous messages so that asynchronous callbacks can find the appropriate waiting process instance. Specifically, Oracle BPEL Server supports the following:

- Correlation set mechanism, which uses message content for correlation
- WS-Addressing specification, which uses SOAP message headers for correlation of asynchronous messages

Follow these steps to see how the **LoanFlowPlus** process invokes the UnitedLoan service and waits for its asynchronous callback.

1. Expand the **scope_collectOffers scope** activity.
2. Expand the **collectOffers flow** activity.
3. Double-click the **invokeUnitedLoan invoke** activity in the left-hand sequence.



In the Invoke window, you see that this activity passes the loan application document as an input variable (named **loanApplication**) to **UnitedLoanService**. The **initiate** operation returns immediately. However, the next activity (the **receive** activity for the **onResult** callback) waits until the service has called back with a loan offer.

WS-Addressing is used for message correlation (by default) and is handled completely transparently by Oracle BPEL Server. If you want to see the WS-Addressing correlation information, you can use a TCP tunnel to see the SOAP messages exchanged with the services.

The BPEL source code for the invoke and receive activities that receive the offer back from **UnitedLoanService** can be seen by clicking the BPEL **Source** view or by right-clicking the activity and selecting **View** > **Go to Source**. This source code is as follows:

```
<sequence>
      <!--  initiate the remote service -->
      <invoke name="invokeUnitedLoan" partnerLink="UnitedLoanService"
              portType="services:LoanService" operation="initiate"
              inputVariable="loanApplication"/>
   . . .
   . . .
 <!--  receive the result of the remote service -->
      <receive partnerLink="UnitedLoanService"
               portType="services:LoanServiceCallback" operation="onResult"
               variable="loanOffer1" createInstance="no"/>
     </sequence>
```

> **See Also:** *Oracle BPEL Process Manager Developer's Guide* for details about using TCP tunneling to view SOAP messages exchanged with services

Reviewing Parallel Processing

As mentioned in "LoanFlowPlus Process Overview" on page 3-1, **LoanFlowPlus** must invoke the two loan service providers in parallel since they can take an arbitrary amount of time to complete. This is done in BPEL by using the flow activity, which enables several actions to be taken in parallel. An example of this is shown in the **collectOffers flow** activity in "Interacting with an Asynchronous Loan Processor Service" on page 3-6. This flow contains two parallel activities — the sequence that invokes **UnitedLoanService** and the sequence that invokes **StarLoanService**.

The simple BPEL source code that implements this flow activity is as follows:

```
<flow name="collectOffers">
     <!-- ************************************************************
         Invoke the first loan provider (UnitedLoan)
           ************************************************************ -->
     <!-- invoke first loan provider -->
     <sequence>
      <!--  initiate the remote service -->
      <invoke name="invokeUnitedLoan" partnerLink="UnitedLoanService"
              portType="services:LoanService" operation="initiate"
              inputVariable="loanApplication"/>
      <!--  receive the result of the remote service -->
      <receive partnerLink="UnitedLoanService"
               portType="services:LoanServiceCallback" operation="onResult"
               variable="loanOffer1" createInstance="no"/>
     </sequence>
     <!-- ************************************************************
```

```
      Invoke the second loan provider (StarLoan)
      ************************************************************ -->
 <sequence>
  <!--  initiate the remote service -->
  <invoke name="invokeStarLoan" partnerLink="StarLoanService"
          portType="services:LoanService" operation="initiate"
          inputVariable="loanApplication"/>
  <!--  receive the result of the remote service -->
  <receive partnerLink="StarLoanService"
           portType="services:LoanServiceCallback" operation="onResult"
           variable="loanOffer2" createInstance="no"/>
 </sequence>
</flow>
```

Reviewing Human Tasks

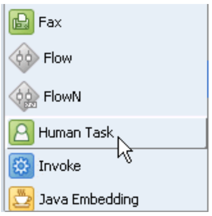
BPEL has fundamental support for asynchronous services, which integrate people and manual tasks into BPEL processes. A set of prebuilt services are integrated with Oracle BPEL Server that enable you to include human workflow in your processes. By implementing this as a true BPEL service, the interface to these services is described in the WSDL and people can be included in 100% standard BPEL processes. To the BPEL process, the person and manual task look like any other asynchronous Web service.

The **LoanFlowPlus** process illustrates manual task support in several places:

- For exception management, as described in "Managing Faults" on page 3-4
- For the approval step, where the flow waits for the customer to confirm the selected loan offer before completing

Follow these instructions to see how a human workflow is implemented:

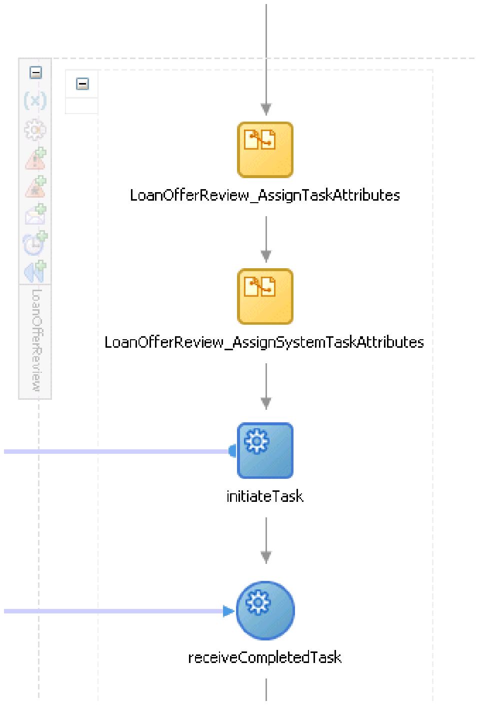
1. Select **Process Activities** from the **Component Palette**.
2. View the **Human Task** activity, which can be dropped into a process flow. This enables you to create a task definition with the Human Task editor. This editor enables you to select assignees, approval patterns, notification channels, and so on.



   The **Human Task** palette element is a template activity that enables the developer to treat a human task as a higher-level abstraction, although it is implemented as a series of BPEL activities.

3. Expand the **LoanOfferReview** human task activity.

   There are several initial assign activities. The process then makes a call to the **TaskService** partner link. This partner link exposes the operations required to act on a task.


LoanOfferReview_AssignTaskAttributes
LoanOfferReview_AssignSystemTaskAttributes
initiateTask
receiveCompletedTask
LoanOfferReview

APIs are provided so that clients can query the tasks assigned to a user, update the task data, and complete or cancel the task. In addition, check out of tasks for group and role support expiration of tasks, escalation, and other common human task requirements are implemented by the task service.

In the **LoanFlowPlus** process, all user interaction is handled by custom Java Server Pages utilizing the Java API. During the customer confirmation, you:

- Interact with the process through the LoanFlowPlusUI interface at `http://localhost:8888/LoanFlowPlusUI`
- Interact with the exception handler through the ExceptionDashboardUI interface at `http://localhost:8888/ExceptionDashboardUI`

Oracle also supplies an out-of-the-box Oracle BPEL Worklist Application that can be used (and customized) to handle human interaction within a workflow-enabled process.

> **See Also:** The Workflow Services chapter of Oracle BPEL Process Manager Developer's Guide for more information on human workflow within BPEL and building task service clients

Viewing the Web Services Invocation Framework

An enterprise frequently has existing Java code or Java APIs for accessing back-end systems that it wants to use in a BPEL process. There are three general approaches for this feature:

- Wrapping Java code as a Web service
- Inlining Java code into a BPEL process
- Using the WSIF invocation framework to call the Java code as if it were a Web service

The first two approaches are straightforward and illustrated by several code examples shipped with Oracle BPEL Process Manager.

As an example of the third approach, the **LoanFlowPlus** process uses the WSIF to invoke an EJB that returns a social security number for the customer. From your BPEL process, you can treat the EJB service as you do any other service that has a WSDL interface and create a partner link for it in your BPEL process. Functionality offered by the EJB is mapped directly to the services WSDL interface through an EJB binding. The **CustomerService.wsdl** file in the **Application Navigator** shows this binding:

```
  <!-- binding declns -->
  <binding name="EJBBinding" type="tns:CustomerService">
    <ejb:binding/>
    <format:typeMapping encoding="Java" style="Java">
      <format:typeMap typeName="xsd:string" formatType="java.lang.String" />
      <format:typeMap typeName="tns:CustomerNotFoundExceptionType"
 formatType="com.otn.samples.CustomerNotFoundException" />
    </format:typeMapping>
    <operation name="getCustomerSSN">
      <ejb:operation
         methodName="getCustomerSSN"
         parameterOrder="email"
         interface="remote"
         returnPart="ssn" />
      <input name="GetCustomerSSNRequest"/>
      <output name="GetCustomerSSNResponse"/>
      <fault name="CustomerNotFoundException"/>
    </operation>
  </binding>

  <!-- service decln -->
  <service name="CustomerService">
    <port name="EJBPort" binding="tns:EJBBinding">
      <!-- Put appserver vendor-specific deployment information here -->
<ejb:address className="com.otn.samples.CustomerServiceHome"
     jndiName="ejb/session/CustomerService"
     initialContextFactory="com.evermind.server.rmi.RMIInitialContextFactory"
     jndiProviderURL="ormi://${hostname}/CustomerService"/>

    </port>
  </service>

    <plnk:partnerLinkType name="CustomerService">
    <plnk:role name="CustomerServiceProvider">
      <plnk:portType name="tns:CustomerService"/>
    </plnk:role>
   </plnk:partnerLinkType>
</definitions>
```

This means that the EJB is being accessed as a WSDL-described service, where WSIF is mapping the operations to EJB methods.

> **Note:** Oracle JDeveloper 10.1.3.1.0 automatically generate WSDLs with Java and EJB bindings for plain old Java objects (POJOs) and EJBs, in addition to the currently supported SOAP bindings.

Building a Custom User Interface for Initiating the BPEL Process

All BPEL processes are themselves Web services. In addition, Oracle BPEL Process Manager provides a Java API for invoking deployed BPEL flows, fetching state and status information from active instances, and so on. A BPEL process is frequently instantiated from a portal or other custom user interface. For example, the LoanFlowPlus process has a JSP loan interface that customers can use to initiate new loan applications, see the offers they have received, and approve offers. The LoanFlowPlus user interface is deployed onto the same application server as Oracle BPEL Process Manager and full source for it can be found in your samples at:

```
SOA_Oracle_Home\bpel\samples\demos\LoanDemoPlus\LoanFlowPlusUI
```

JavaDocs for the Java API for initiating a deployed BPEL process can be found in:

```
SOA_racle_Home\bpel\docs\apidocs\index.html
```

In addition, a JSP tag library is available for easy use of this Java API from a JSP (this tag library is used by the LoanFlowPlusUI sample). Developers can also use the Web services SOAP and WSDL API to invoke BPEL processes. The Web services approach enables BPEL flows to be invoked and accessed from any language or tool kit that supports Web services.

Initiating the LoanFlowPlus Process

You now use the portal user interface to initiate and complete a LoanFlowPlus instance and Oracle Enterprise Manager 10g BPEL Control to view the status, audit trail, debugging information, performance metrics, and other information gathered automatically by Oracle BPEL Server.

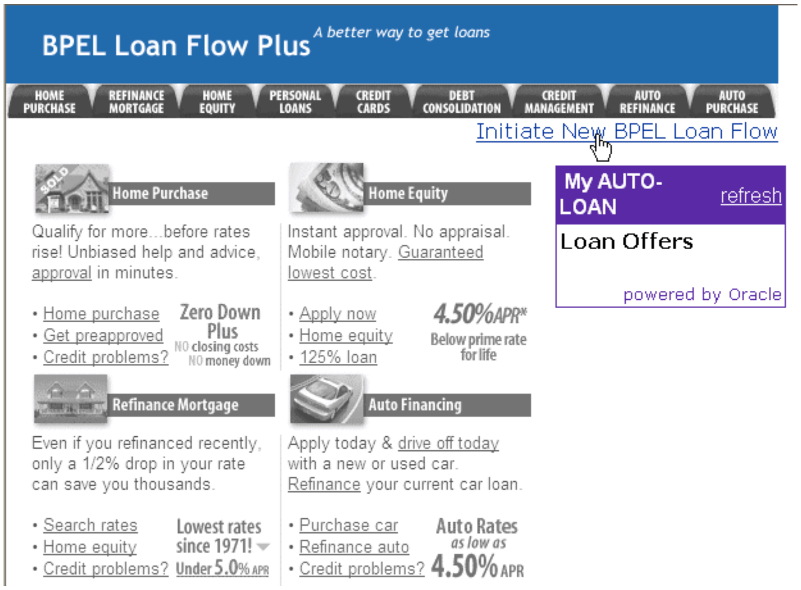
The LoanFlowPlus BPEL process was compiled and deployed, along with its dependent services and the portal user interfaces, when you executed the `ant` command in "Starting and Testing Your Service" on page 3-3. You can also build and deploy processes from Oracle JDeveloper, as you did in Chapter 2, "Credit Flow Tutorial".

Follow these instructions to initiate and complete a LoanFlowPlus instance:

1. Go to the following portal user interface:

   ```
   http://localhost:8888/LoanFlowPlusUI/Homepage.html
   ```

2. Click the **Initiate New BPEL Loan Flow** link on the portal page:



Qualify for more...before rates rise! Unbiased help and advice, approval in minutes.

- Home purchase
- Get preapproved
- Credit problems?

Zero Down Plus NO closing costs NO money down

Instant approval. No appraisal. Mobile notary. Guaranteed lowest cost.

- Apply now
- Home equity
- 125% loan

4.50%APR* Below prime rate for life

Even if you refinanced recently, only a 1/2% drop in your rate can save you thousands.

- Search rates
- Home equity
- Credit problems?

Lowest rates since 1971! Under 5.0% APR

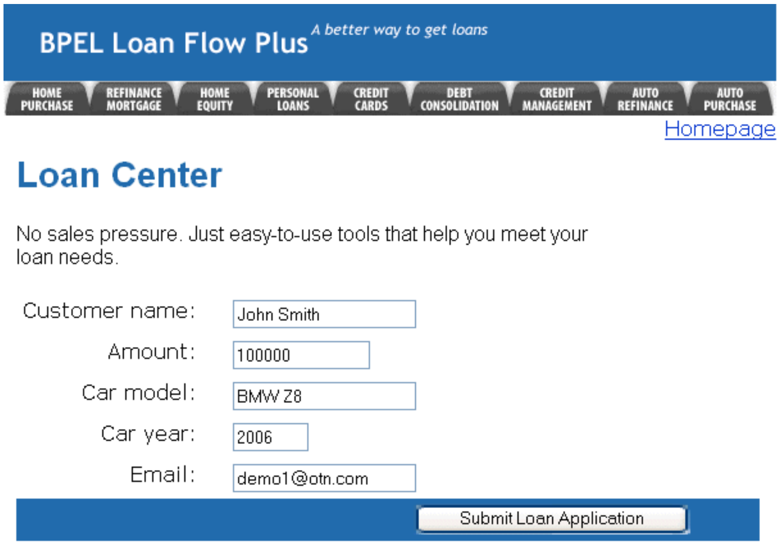
Apply today & drive off today with a new or used car. Refinance your current car loan.

- Purchase car
- Refinance auto
- Credit problems?

Auto Rates as low as 4.50% APR

A Web page appears that enables you to submit a loan application to initiate a new LoanFlowPlus instance.

3. Change the fields in the user interface, if you want, and click **Submit Loan Application** to initiate the new process instance:



Loan Center

No sales pressure. Just easy-to-use tools that help you meet your loan needs.

Customer name: John Smith

Amount: 100000

Car model: BMW Z8

Car year: 2006

Email: demo1@otn.com

Viewing the Visual Audit Trail

You now perform the tasks of a developer or administrator and use Oracle Enterprise Manager 10g BPEL Control to view the audit trail and other status information regarding this process.

1. Access Oracle Enterprise Manager 10g BPEL Control through one of the following methods:

   - Selecting **Start** > **All Programs** > **Oracle -** ***Oracle_Home*** > **Oracle BPEL Process Manager** > **BPEL Control**
   - Going to the following URL:

```
http://localhost:8888/BPELConsole/
```

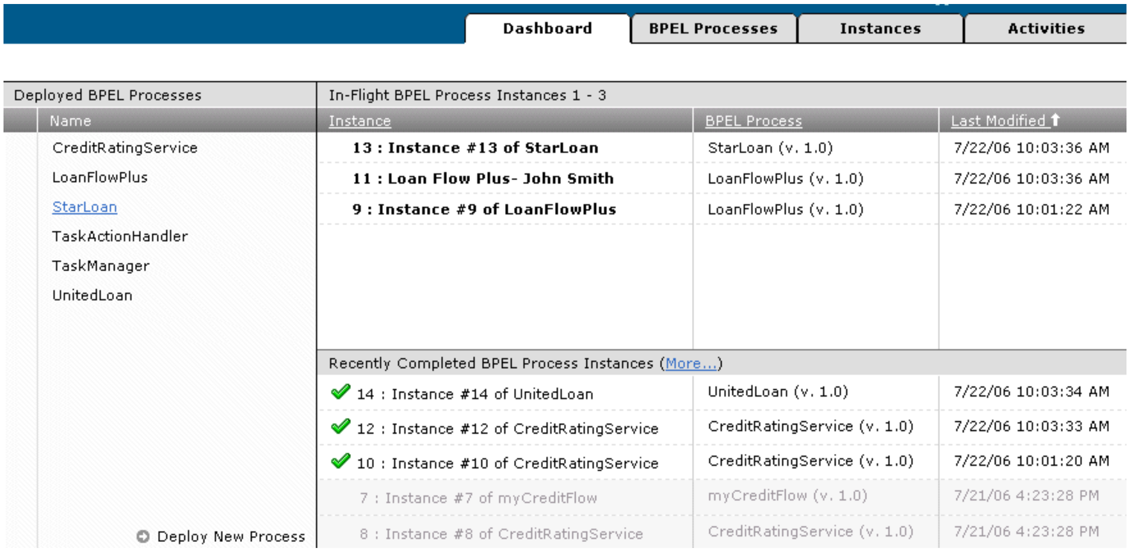
2. Enter the following details to log in to Oracle Enterprise Manager 10g BPEL Control and click **Login**:

Field	Value
Username	`bpeladmin`
Password	*`password`*

Where *`password`* is the value you entered during installation in "Installing Oracle Application Server SOA Suite" on page 1-5.

Several in-flight instances and several completed instances of BPEL processes appear.

3. Click the active instance of the **LoanFlowPlus** process in the **Instance** column.



Deployed BPEL Processes

Name
CreditRatingService
LoanFlowPlus
StarLoan
TaskActionHandler
TaskManager
UnitedLoan

In-Flight BPEL Process Instances 1 - 3

Instance	BPEL Process	Last Modified
13 : Instance #13 of StarLoan	StarLoan (v. 1.0)	7/22/06 10:03:36 AM
11 : Loan Flow Plus- John Smith	LoanFlowPlus (v. 1.0)	7/22/06 10:03:36 AM
9 : Instance #9 of LoanFlowPlus	LoanFlowPlus (v. 1.0)	7/22/06 10:01:22 AM

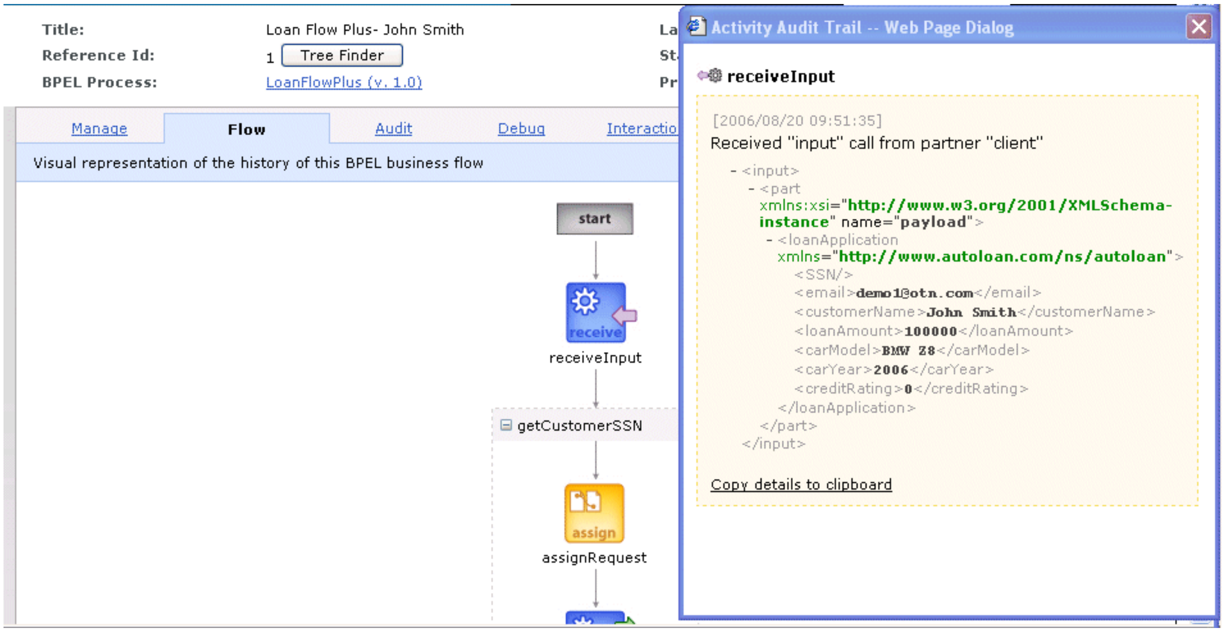
Recently Completed BPEL Process Instances (More...)

✓ 14 : Instance #14 of UnitedLoan	UnitedLoan (v. 1.0)	7/22/06 10:03:34 AM
✓ 12 : Instance #12 of CreditRatingService	CreditRatingService (v. 1.0)	7/22/06 10:03:33 AM
✓ 10 : Instance #10 of CreditRatingService	CreditRatingService (v. 1.0)	7/22/06 10:01:20 AM
7 : Instance #7 of myCreditFlow	myCreditFlow (v. 1.0)	7/21/06 4:23:28 PM
8 : Instance #8 of CreditRatingService	CreditRatingService (v. 1.0)	7/21/06 4:23:28 PM

Everything you see in Oracle Enterprise Manager 10g BPEL Control is also available through an API. In fact, Oracle Enterprise Manager 10g BPEL Control is just a set of JSPs that can serve as a code example of how to use Oracle BPEL Server APIs. Developers frequently use these APIs to build custom process dashboards, search screens, and other interfaces so users and administrators can more efficiently access and manage process instances. This is particularly common as applications are deployed to a production environment where there can be thousands or even millions of process instances to manage.

Once you select the **LoanFlowPlus** instance in Oracle Enterprise Manager 10g BPEL Control, you see the visual audit trail that shows the current status of the process and its execution history. This audit trail is automatically generated and maintained by Oracle BPEL Server, though you can control what information gets logged.

4. Click an activity in the visual audit trail to view detailed information for that activity (for this example, **receiveInput**). In the following window, for example, the client initiation has been selected and you see the XML input message that was sent to initiate the flow.



Title: Loan Flow Plus- John Smith

Reference Id: 1 Tree Finder

BPEL Process: LoanFlowPlus (v. 1.0)

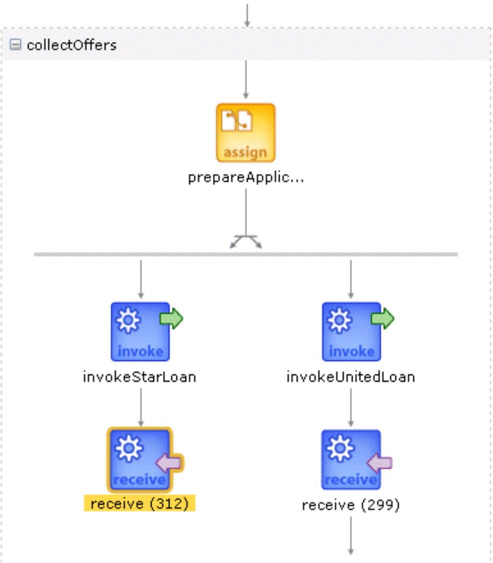
Manage
Flow
Audit
Debug
Interactio
Visual representation of the history of this BPEL business flow
start
receive
receiveInput
getCustomerSSN
assign
assignRequest
Activity Audit Trail -- Web Page Dialog

receiveInput

```
[2006/08/20 09:51:35]
Received "input" call from partner "client"
  - <input>
    - <part
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-
      instance" name="payload">
      - <loanApplication
        xmlns="http://www.autoloan.com/ns/autoloan">
          <SSN/>
          <email>demo1@otn.com</email>
          <customerName>John Smith</customerName>
          <loanAmount>100000</loanAmount>
          <carModel>BMW Z8</carModel>
          <carYear>2006</carYear>
          <creditRating>0</creditRating>
        </loanApplication>
      </part>
    </input>
```

Copy details to clipboard

5. Scroll down to the bottom of the visual audit trail to view the latest status for this instance. In this case, the StarLoan service has been initiated but the flow is waiting to receive a response (as indicated by the orange highlighting). However, the UnitedLoan service has already called back with a loan offer.

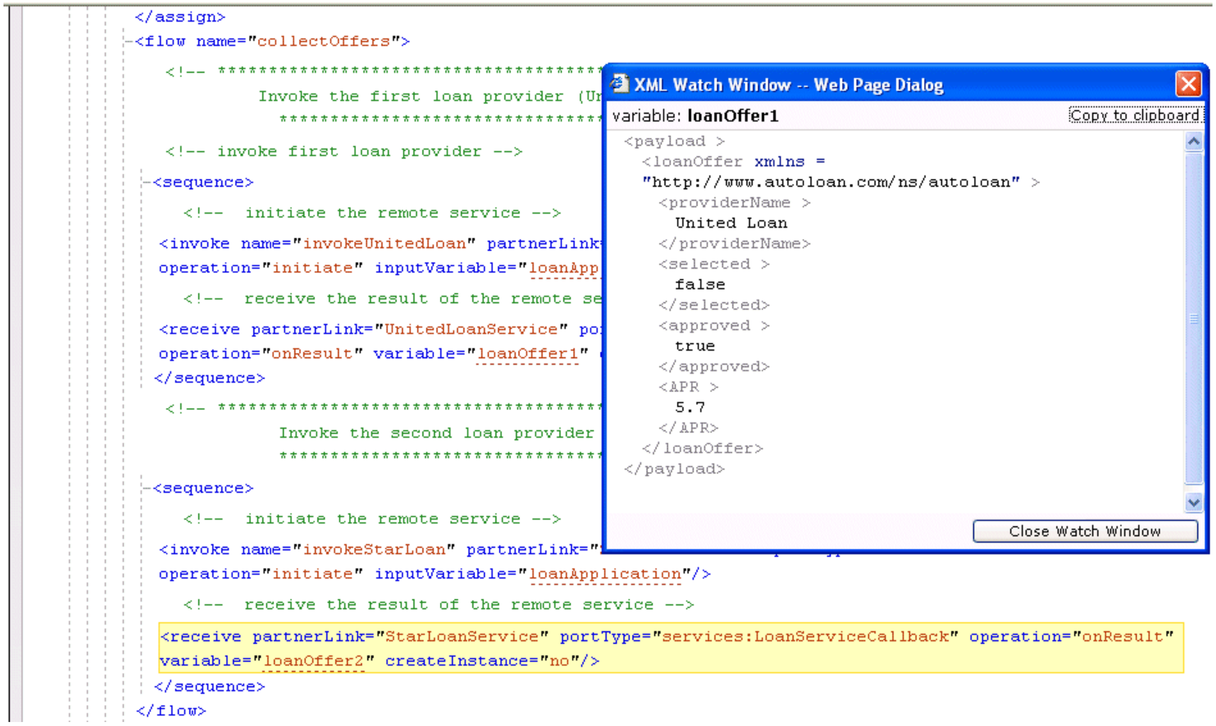


6. Click the **Audit** tab to see a text-based audit trail. The text audit trail includes information such as the messages exchanged with services and timestamps for when each activity was started or completed. This is also configurable and accessible through the API.

Debugging the In-Flight BPEL Process

1. Click the **Debug** tab to see the BPEL Debugger, which takes the BPEL source code that implements this process and matches it against the state of this particular instance. Points in the code where execution is currently paused are highlighted in

yellow. As shown at the bottom of this code, the process is currently waiting for the StarLoan service to call back with a loan offer.



```
</assign>
<flow name="collectOffers">

  <!-- ***************************************
           Invoke the first loan provider (Ur
             ********************************

  <!-- invoke first loan provider -->
 <sequence>
    <!--  initiate the remote service -->
   <invoke name="invokeUnitedLoan" partnerLink
   operation="initiate" inputVariable="loanApp
    <!--  receive the result of the remote se
   <receive partnerLink="UnitedLoanService" po
   operation="onResult" variable="loanOffer1"
  </sequence>
  <!-- ***************************************
             Invoke the second loan provider
             ********************************
 <sequence>
    <!--  initiate the remote service -->
   <invoke name="invokeStarLoan" partnerLink="
   operation="initiate" inputVariable="loanApplication"/>
    <!--  receive the result of the remote service -->
   <receive partnerLink="StarLoanService" portType="services:LoanServiceCallback" operation="onResult"
   variable="loanOffer2" createInstance="no"/>
  </sequence>
</flow>
```

variable: **loanOffer1**

```
<payload >
  <loanOffer xmlns =
  "http://www.autoloan.com/ns/autoloan" >
    <providerName >
      United Loan
    </providerName>
    <selected >
      false
    </selected>
    <approved >
      true
    </approved>
    <APR >
      5.7
    </APR>
  </loanOffer>
</payload>
```

2. Select variables in the debugger to inspect their current values. The value of the loan offer variable (named **loanOffer1**) that contains the result of the UnitedLoan service callback operation is shown in the preceding figure. As you can see, UnitedLoan has approved the loan application and returned a 5.7% interest rate loan offer.

Completing the Asynchronous StarLoan Service

If you now want to complete the process, you can start the StarLoan customer service representative dashboard, since StarLoan requires manual processing of loan applications. StarLoan has also been implemented as a BPEL process (though implementations are available where StarLoan is built with other Web services tool kits, such as Microsoft .Net, Apache Axis, and so on) and serves as an additional BPEL code example.

Follow these instructions to play the role of the StarLoan loan officer.

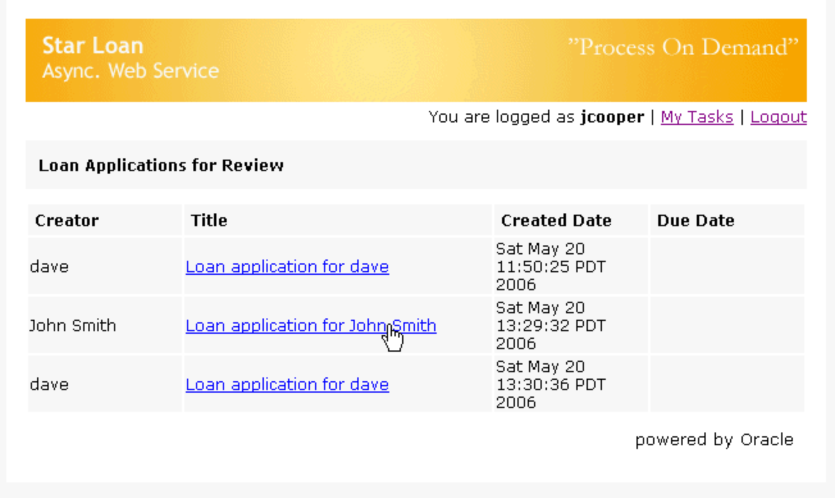
1. Go to the following URL:

   `http://localhost:8888/StarLoanUI/home.jsp`

2. Enter the following details to log in and click **Submit**:

Field	Value
Username	`jcooper`
Password	*`password`*

   Pending loan applications appear.



Star Loan
Async. Web Service

"Process On Demand"

You are logged as **jcooper** | My Tasks | Logout

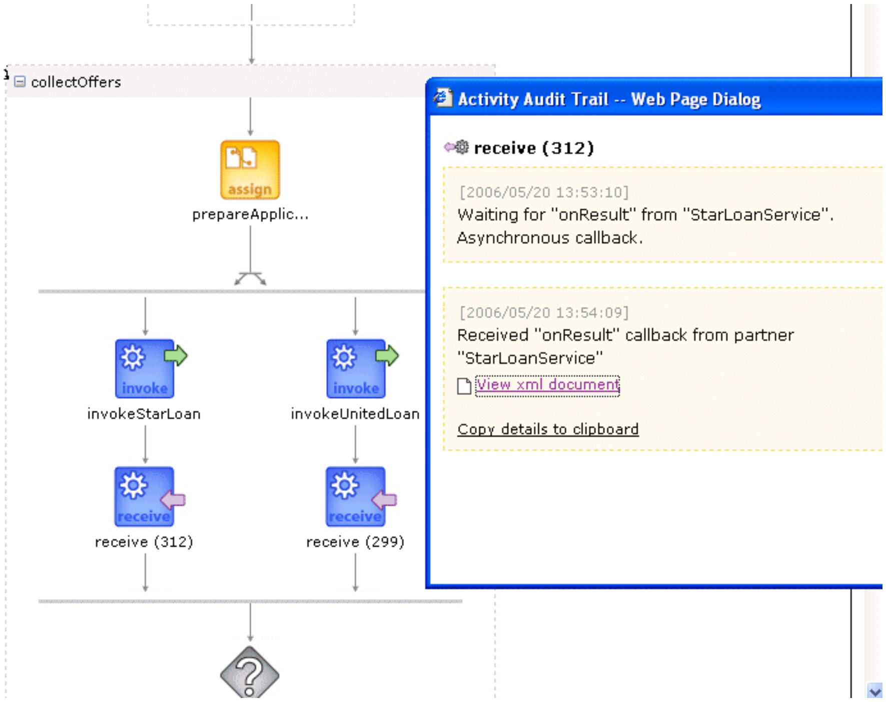
Loan Applications for Review

Creator	Title	Created Date	Due Date
dave	Loan application for dave	Sat May 20 11:50:25 PDT 2006	
John Smith	Loan application for John Smith	Sat May 20 13:29:32 PDT 2006	
dave	Loan application for dave	Sat May 20 13:30:36 PDT 2006	

powered by Oracle

3. Select the loan application, enter an interest rate (enter a low rate since you are approving your own loan application), and click **Approve**. This causes the StarLoan service to return a callback to the LoanFlowPlus process.

4. Refresh the audit trail in Oracle Enterprise Manager 10g BPEL Control to see that processing has progressed, having received the StarLoan callback with the second loan offer and selected the best loan offer. It is now waiting for the customer to approve the offer.



Activity Audit Trail -- Web Page Dialog

receive (312)

[2006/05/20 13:53:10]
Waiting for "onResult" from "StarLoanService". Asynchronous callback.

[2006/05/20 13:54:09]
Received "onResult" callback from partner "StarLoanService"

View xml document

Copy details to clipboard

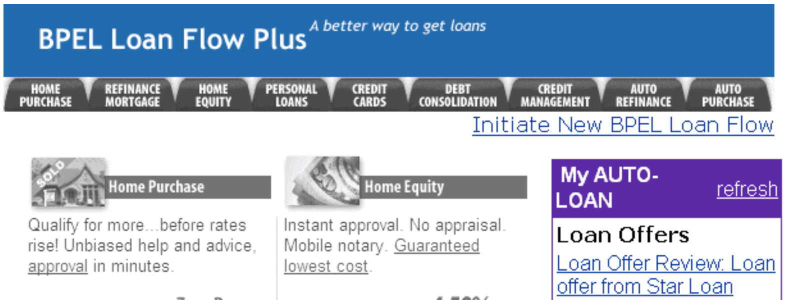
5. Click **View xml document** to view specific details about the loan offer:

```
  <?xml version="1.0" encoding="UTF-8" ?>
- <loanOffer xmlns="http://www.autoloan.com/ns/autoloan">
  <providerName>Star Loan</providerName>
  <selected />
  <approved>true</approved>
  <APR>5.5</APR>
```

```
        </loanOffer>
```

Completing a Human Task

1. Refresh the customer loan portal home page. You now see the selected loan offer waiting for review. This is the final step in the process.

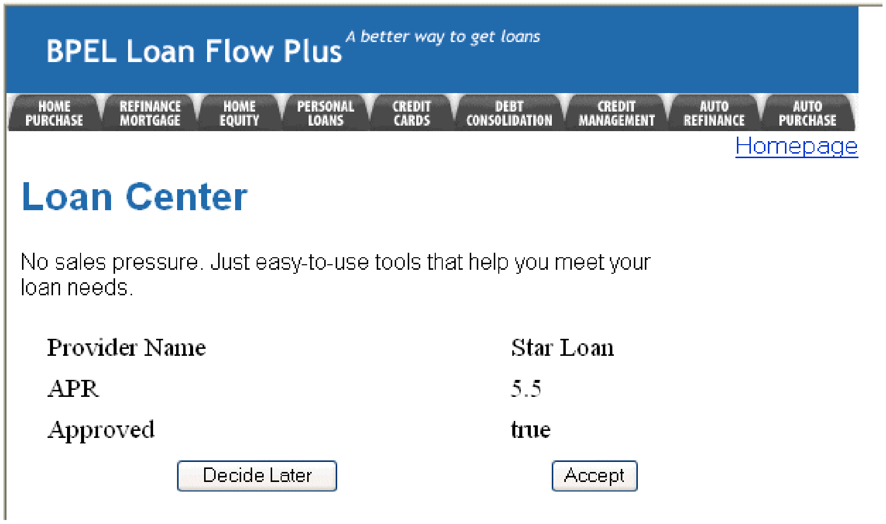


Qualify for more...before rates rise! Unbiased help and advice, approval in minutes.

Instant approval. No appraisal. Mobile notary. Guaranteed lowest cost.

Loan Offer Review: Loan offer from Star Loan

2. If you select this loan offer, a page displays that enables the customer to accept this offer. This completes the execution of this instance of the LoanFlowPlus BPEL process.



Loan Center

No sales pressure. Just easy-to-use tools that help you meet your loan needs.

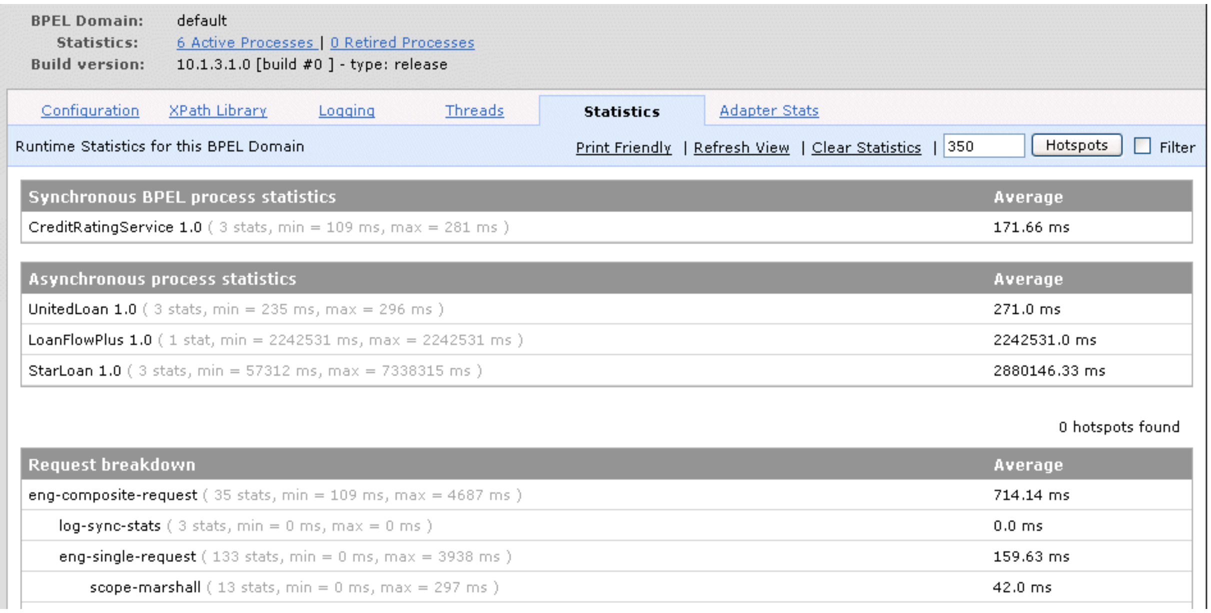
Provider Name	Star Loan
APR	5.5
Approved	true

Tuning Performance

Performance (including throughput and response time requirements) is frequently a critical requirement for effectively implementing a BPEL process. Built in to Oracle BPEL Server and Oracle Enterprise Manager 10g BPEL Control is a performance-tuning framework. This framework assists developers in understanding the performance of their applications and identifying and resolving bottlenecks (whether in the developer’s code, an external service, or Oracle BPEL Server).

1. Return to Oracle Enterprise Manager 10g BPEL Control.
2. Click **Manage BPEL Domain** in the upper right corner, and then click **Statistics**.

   A page with the performance data for your completed flows appears:



BPEL Domain:	default
Statistics:	6 Active Processes | 0 Retired Processes
Build version:	10.1.3.1.0 [build #0] - type: release

Configuration
XPath Library
Logging
Threads
Statistics
Adapter Stats

Runtime Statistics for this BPEL Domain

Print Friendly | Refresh View | Clear Statistics | 350 Hotspots □ Filter

Synchronous BPEL process statistics	Average
CreditRatingService 1.0 (3 stats, min = 109 ms, max = 281 ms)	171.66 ms

Asynchronous process statistics	Average
UnitedLoan 1.0 (3 stats, min = 235 ms, max = 296 ms)	271.0 ms
LoanFlowPlus 1.0 (1 stat, min = 2242531 ms, max = 2242531 ms)	2242531.0 ms
StarLoan 1.0 (3 stats, min = 57312 ms, max = 7338315 ms)	2880146.33 ms

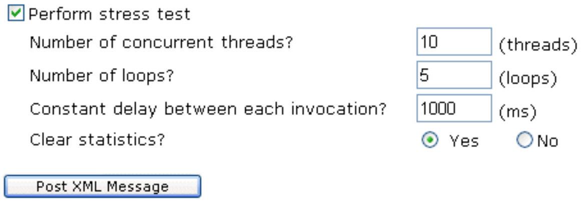
0 hotspots found

Request breakdown	Average
eng-composite-request (35 stats, min = 109 ms, max = 4687 ms)	714.14 ms
log-sync-stats (3 stats, min = 0 ms, max = 0 ms)	0.0 ms
eng-single-request (133 stats, min = 0 ms, max = 3938 ms)	159.63 ms
scope-marshall (13 stats, min = 0 ms, max = 297 ms)	42.0 ms

Oracle Enterprise Manager 10g BPEL Control enables you to test any of your deployed processes through an automatically generated HTML form interface or by passing specific XML message content to it. This test page includes stress test capabilities that enable you to do load testing of a deployed BPEL process and view the performance statistics of the flow under stress.

3. Click the **Dashboard** tab.
4. Click **LoanFlowPlus** in the **Deployed BPEL Processes** list.
5. Scroll to the bottom and select **Perform stress test**.

> **Note:** Do not use this built-in stress test capability with the Oracle Database Lite database included with the Oracle Application Server SOA Basic Install type.



☑ Perform stress test

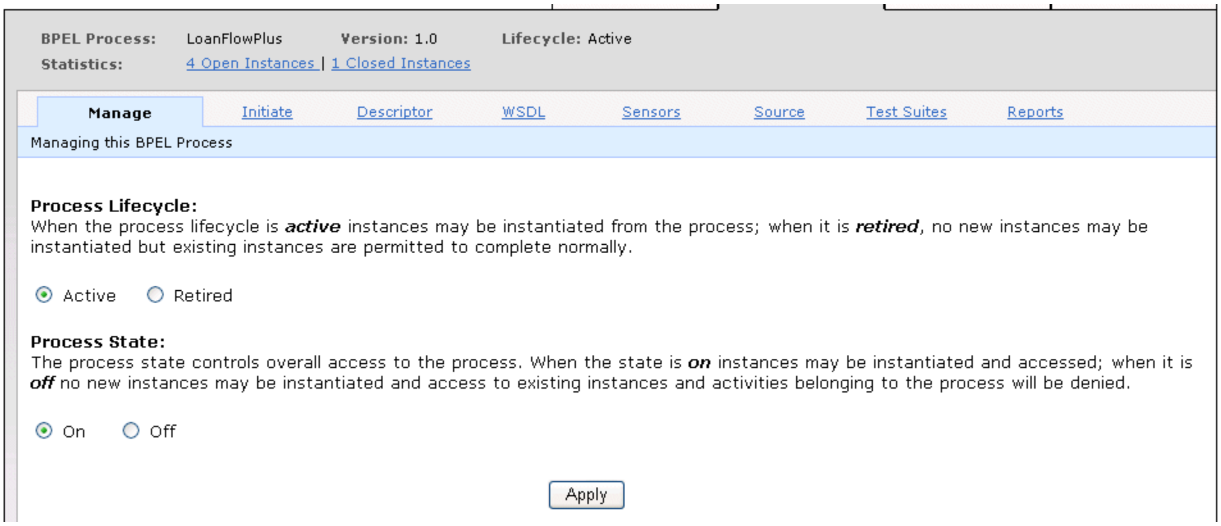
Number of concurrent threads?	10	(threads)
Number of loops?	5	(loops)
Constant delay between each invocation?	1000	(ms)
Clear statistics?	● Yes	○ No

Managing the Process Lifecycle

Oracle BPEL Process Manager and Oracle Enterprise Manager 10g BPEL Control provide support for process lifecycle management. For example, side-by-side versioning is supported so that new implementations of a process can be hot-deployed without disturbing current in-flight instances of prior implementations of the process. In addition, Oracle Enterprise Manager 10g BPEL Control can turn off an entire process, undeploy a process, or retire a process such that existing instances can complete, but new instances of the process are not allowed.

1. Select the name of a deployed BPEL process in Oracle Enterprise Manager 10g BPEL Control and click the **Manage** tab on the left side.

   The following page appears.



Managing this BPEL Process

Process Lifecycle:
When the process lifecycle is ***active*** instances may be instantiated from the process; when it is ***retired***, no new instances may be instantiated but existing instances are permitted to complete normally.

● Active ○ Retired

Process State:
The process state controls overall access to the process. When the state is ***on*** instances may be instantiated and accessed; when it is ***off*** no new instances may be instantiated and access to existing instances and activities belonging to the process will be denied.

● On ○ Off

See Also: *Oracle BPEL Process Manager Developer's Guide* for additional details about the process lifecycle.

Building and Viewing Process Reports

Oracle BPEL Process Manager provides the following process reporting features:

- Building A Reporting Dashboard
- Viewing Oracle Enterprise Manager 10g BPEL Control Reports
- Viewing BAM Reports

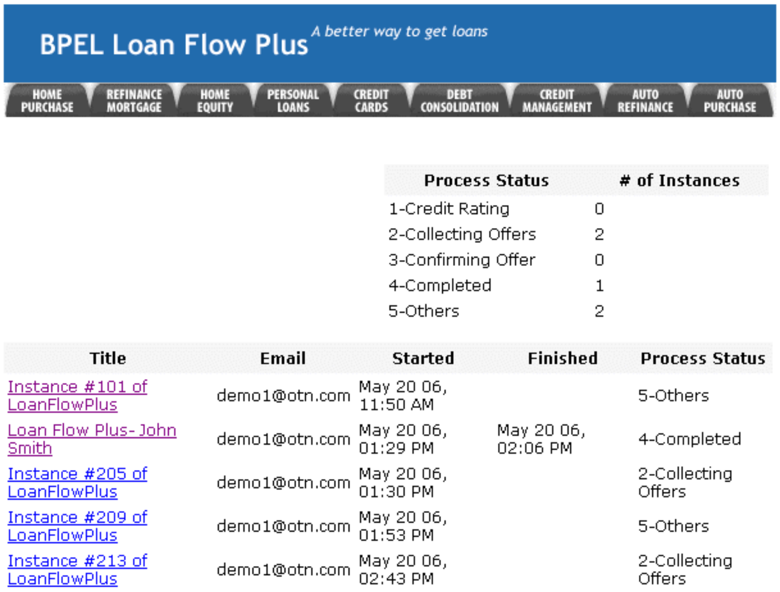
Building A Reporting Dashboard

An important feature of automating a business process is often to provide greater visibility into process state information. This is often realized by building a process dashboard that displays aggregate summary information or process instance state information. For example, an executive may want to know how many loans are currently at each stage of processing, the average time to present a loan offer to a customer, or other key performance indicators against live and completed process instances.

Oracle BPEL Process Manager maintains a great deal of information regarding process state and makes this information available through the API. In addition, several simple reporting dashboard templates are shipped with Oracle BPEL Server or are available from professional services and support organizations.

1. Access a template included with the LoanFlowPlus process through the following URL:

   ```
   http://localhost:8888/LoanFlowPlusUI/dashboard.jsp
   ```



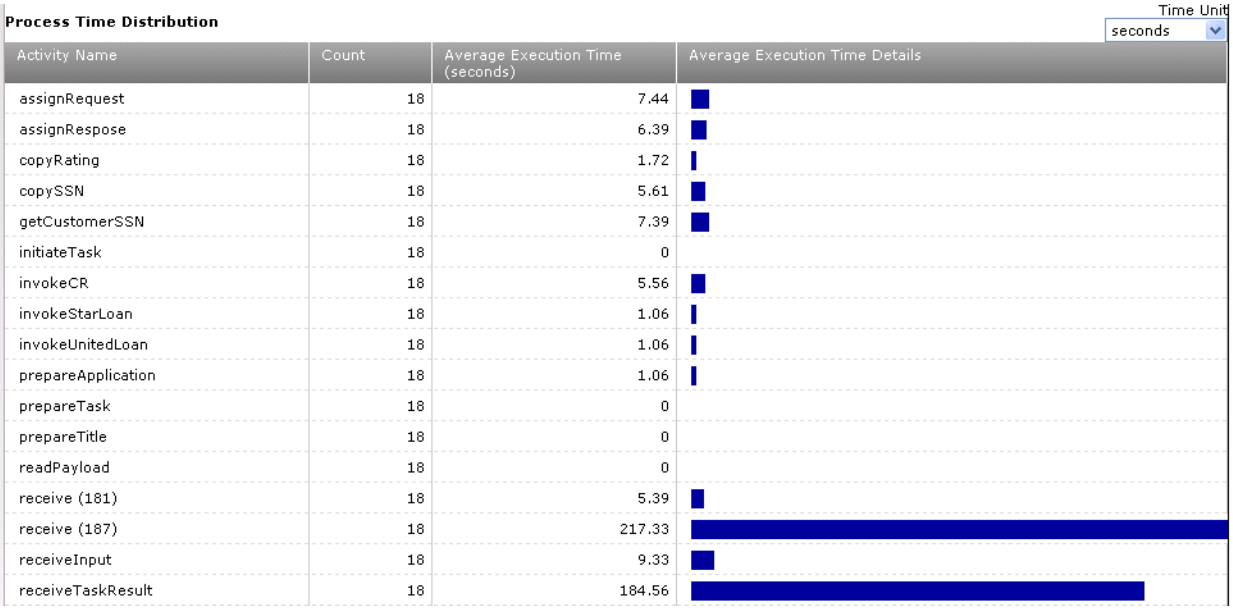
Process Status	# of Instances
1-Credit Rating	0
2-Collecting Offers	2
3-Confirming Offer	0
4-Completed	1
5-Others	2

Title	Email	Started	Finished	Process Status
Instance #101 of LoanFlowPlus	demo1@otn.com	May 20 06, 11:50 AM		5-Others
Loan Flow Plus- John Smith	demo1@otn.com	May 20 06, 01:29 PM	May 20 06, 02:06 PM	4-Completed
Instance #205 of LoanFlowPlus	demo1@otn.com	May 20 06, 01:30 PM		2-Collecting Offers
Instance #209 of LoanFlowPlus	demo1@otn.com	May 20 06, 01:53 PM		5-Others
Instance #213 of LoanFlowPlus	demo1@otn.com	May 20 06, 02:43 PM		2-Collecting Offers

Viewing Oracle Enterprise Manager 10g BPEL Control Reports

Oracle BPEL Process Manager also provides built-in reports to Oracle Enterprise Manager 10g BPEL Control to enable process optimization.

1. Return to Oracle Enterprise Manager 10g BPEL Control.
2. Click **LoanFlowPlus** in the **Deployed BPEL Processes** list of the **Dashboard** tab.
3. Click the **Reports** tab.
4. Select a specific report type, specify specific query details, and click **Go**. An example of the process time distribution report type is as follows.


Process Time Distribution

Time Unit: seconds

Activity Name	Count	Average Execution Time (seconds)	Average Execution Time Details
assignRequest	18	7.44	
assignRespose	18	6.39	
copyRating	18	1.72	
copySSN	18	5.61	
getCustomerSSN	18	7.39	
initiateTask	18	0	
invokeCR	18	5.56	
invokeStarLoan	18	1.06	
invokeUnitedLoan	18	1.06	
prepareApplication	18	1.06	
prepareTask	18	0	
prepareTitle	18	0	
readPayload	18	0	
receive (181)	18	5.39	
receive (187)	18	217.33	
receiveInput	18	9.33	
receiveTaskResult	18	184.56	

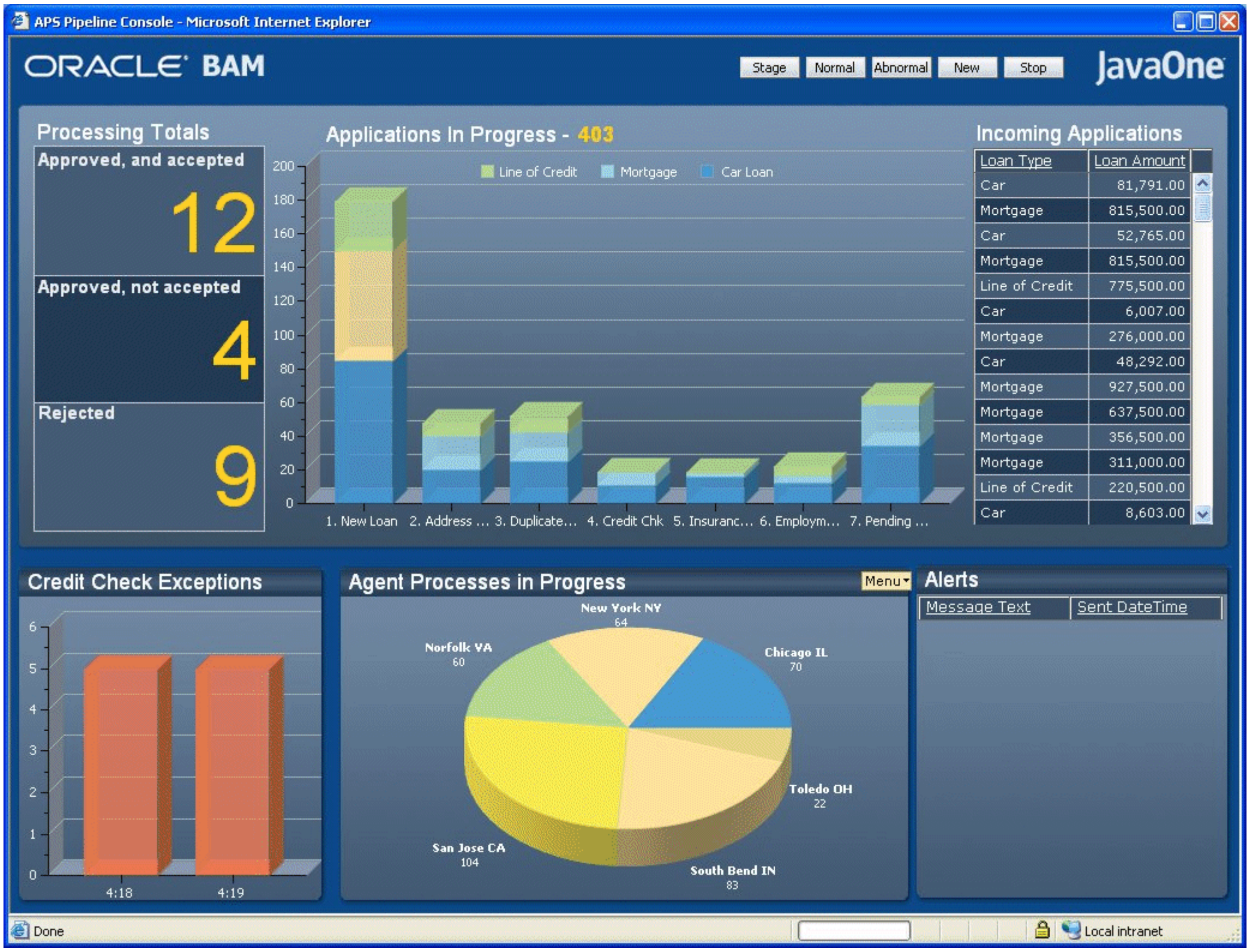
> **See Also:** *Oracle BPEL Process Manager Developer's Guide* for additional details about Oracle Enterprise Manager 10g BPEL Control reports

Viewing BAM Reports

Oracle also provides a business activity monitoring (BAM) component that integrates with Oracle BPEL Process Manager. Oracle BAM enables business executives to monitor their business events and processes in the enterprise, correlate messages and define KPIs (key performance indicators) and define alerts and take corrective action based on real-time information accessed through a live-data, thin-client web interface.

Figure 3–2 provides an example of the Oracle BAM user interface displaying information:

Figure 3–2 Oracle BAM Reports



Processing Totals

Approved, and accepted 12

Approved, not accepted 4

Rejected 9

Incoming Applications

Loan Type	Loan Amount
Car	81,791.00
Mortgage	815,500.00
Car	52,765.00
Mortgage	815,500.00
Line of Credit	775,500.00
Car	6,007.00
Mortgage	276,000.00
Car	48,292.00
Mortgage	927,500.00
Mortgage	637,500.00
Mortgage	356,500.00
Mortgage	311,000.00
Line of Credit	220,500.00
Car	8,603.00

Alerts

Message Text	Sent DateTime

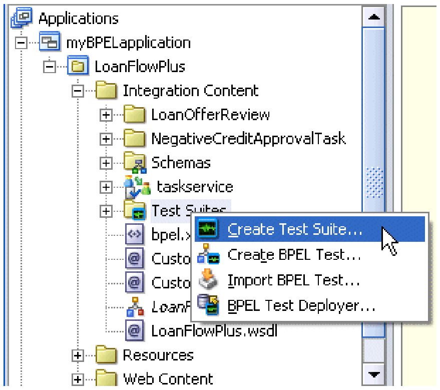
Oracle JDeveloper includes support for sensors that make it easy to send events from BPEL processes to Oracle BAM Server. However, Oracle BAM Server can gather events from any source within the enterprise (JMS messages, database events, and so on) that can write to a JMS queue, and is not limited to BPEL process events.

Testing BPEL Processes in a Development Environment

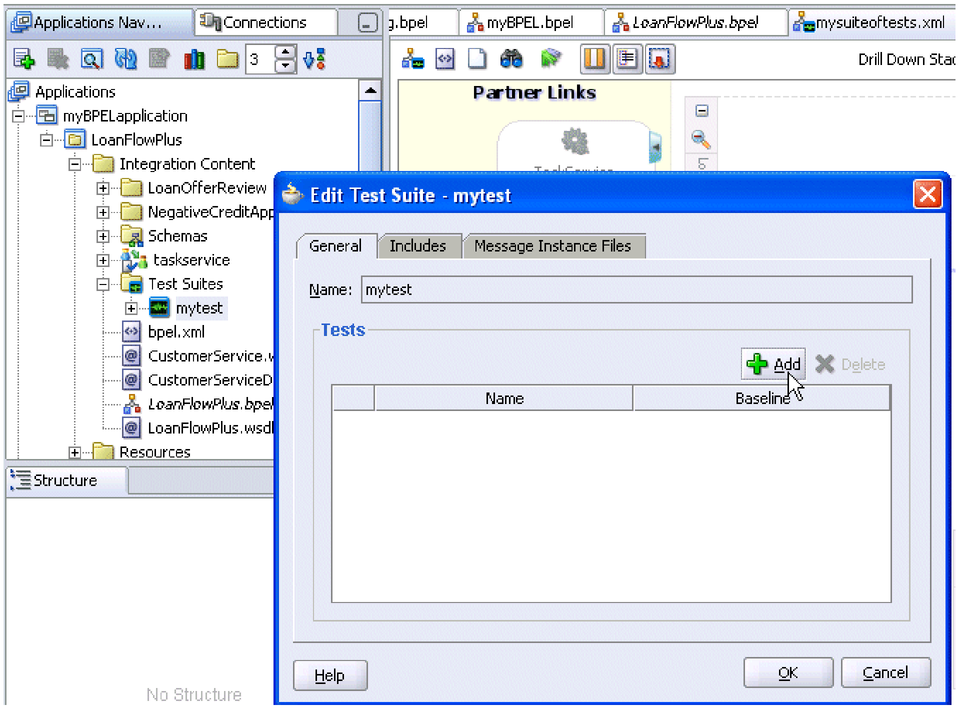
Oracle BPEL Process Manager enables you to create and deploy test suites with a BPEL process to Oracle BPEL Server. Test suites consist of XML test files that enable you to automate testing of a BPEL process in a development environment. The tests contain instructions you create for simulating partner Web service interactions, validating process actions with test data, and calculating the percentage of source code executed in simple activities. This testing ensures that a process runs correctly before its deployment to a production environment.

Follow these instructions to create and deploy test suites.

1. Right-click **Test Suites** in Oracle JDeveloper and select **Create Test Suite**.



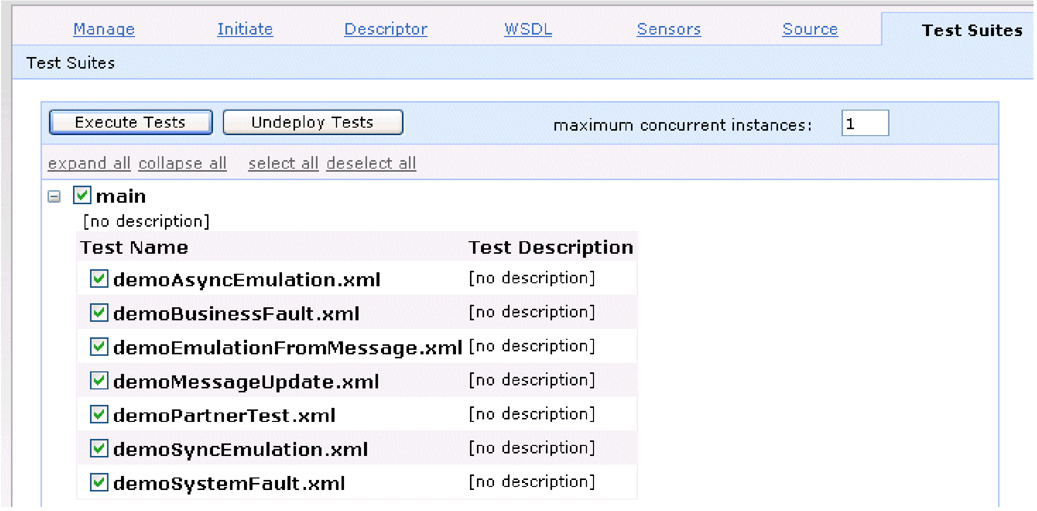
2. Enter a name for the test suite and click **OK**.
3. Right-click the test suite you created and select **Edit Test Suite**.



Name: mytest

4. Add baseline and emulated message instance test files to your test suite.
5. Deploy test suites for the BPEL process to Oracle BPEL Server.

6. Run the test suite on a BPEL process instance and create XML document reports from Oracle Enterprise Manager 10g BPEL Control (under the **BPEL Processes** tab > ***process_instance_name*** > **Test Suites**) or an `ant` task.



maximum concurrent instances: 1

☑ main
[no description]

Test Name	Test Description
☑ demoAsyncEmulation.xml	[no description]
☑ demoBusinessFault.xml	[no description]
☑ demoEmulationFromMessage.xml	[no description]
☑ demoMessageUpdate.xml	[no description]
☑ demoPartnerTest.xml	[no description]
☑ demoSyncEmulation.xml	[no description]
☑ demoSystemFault.xml	[no description]

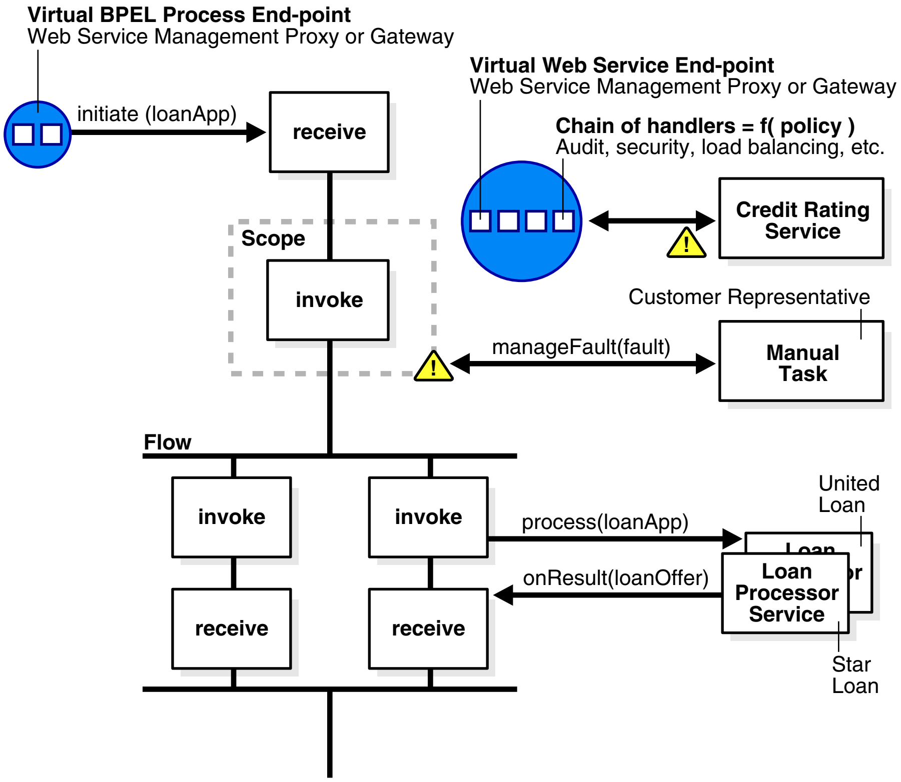
You can view report results, fix any errors, and repeat test execution.

> **See Also:** *Oracle BPEL Process Manager Developer's Guide* for additional details about BPEL process testing

Integrating with a Web Service Management Solution

A frequently asked question is how do you secure and manage both services and the BPEL processes that orchestrate them within an enterprise adopting service-oriented architecture (SOA). A Web service management (WSM) solution such as Oracle Web Services Manager (OWSM) provides features such as authentication, authorization, encryption, and logging for services and processes through the definition of policies. As shown in Figure 3–3, Web service management solutions typically provide proxies or gateways in front of Web services. This includes both back-end Web services that can be implemented in many different technologies and BPEL processes deployed to Oracle BPEL Server, or other BPEL servers, since a BPEL process is itself a Web service.

Figure 3–3 Web Service Management

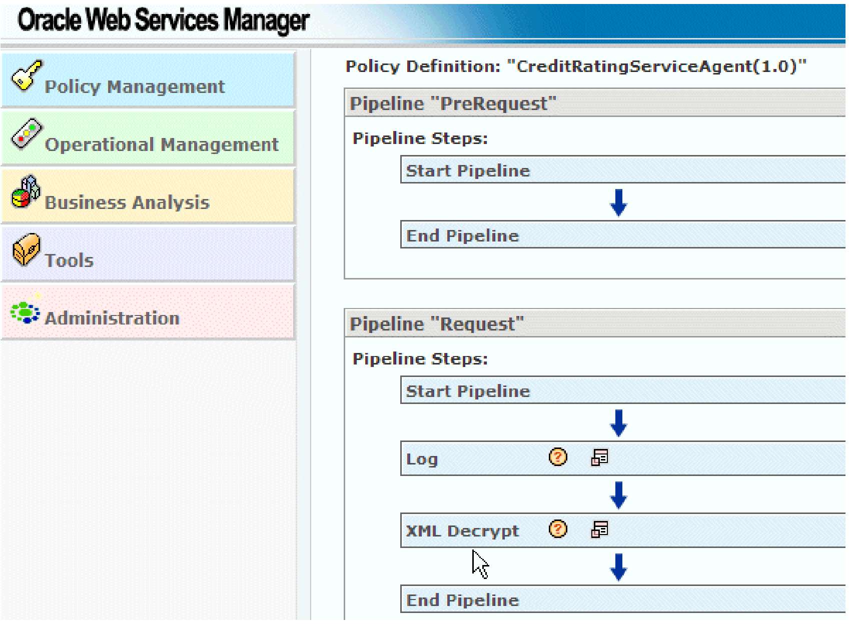


****

OWSM provides a managed end-point here, which looks to the BPEL process just like the underlying Web service. The BPEL process does not need to know that it is connecting to a managed end-point rather than the service itself. OWSM (or other products that support the same standards) then provides a pipeline of steps so that encryption, authentication, service virtualization, and other key requirements can be implemented in a uniform manner. Likewise, OWSM can be placed logically in front of Oracle BPEL Process Manager to manage the connections to BPEL processes.

Figure 3–4 and Figure 3–5 show the Oracle Web Services Manager user interface:

Figure 3–4 Oracle Web Services Manager User Interface



Oracle Web Services Manager

Policy Definition: "CreditRatingServiceAgent(1.0)"

Pipeline "PreRequest"

Pipeline Steps:

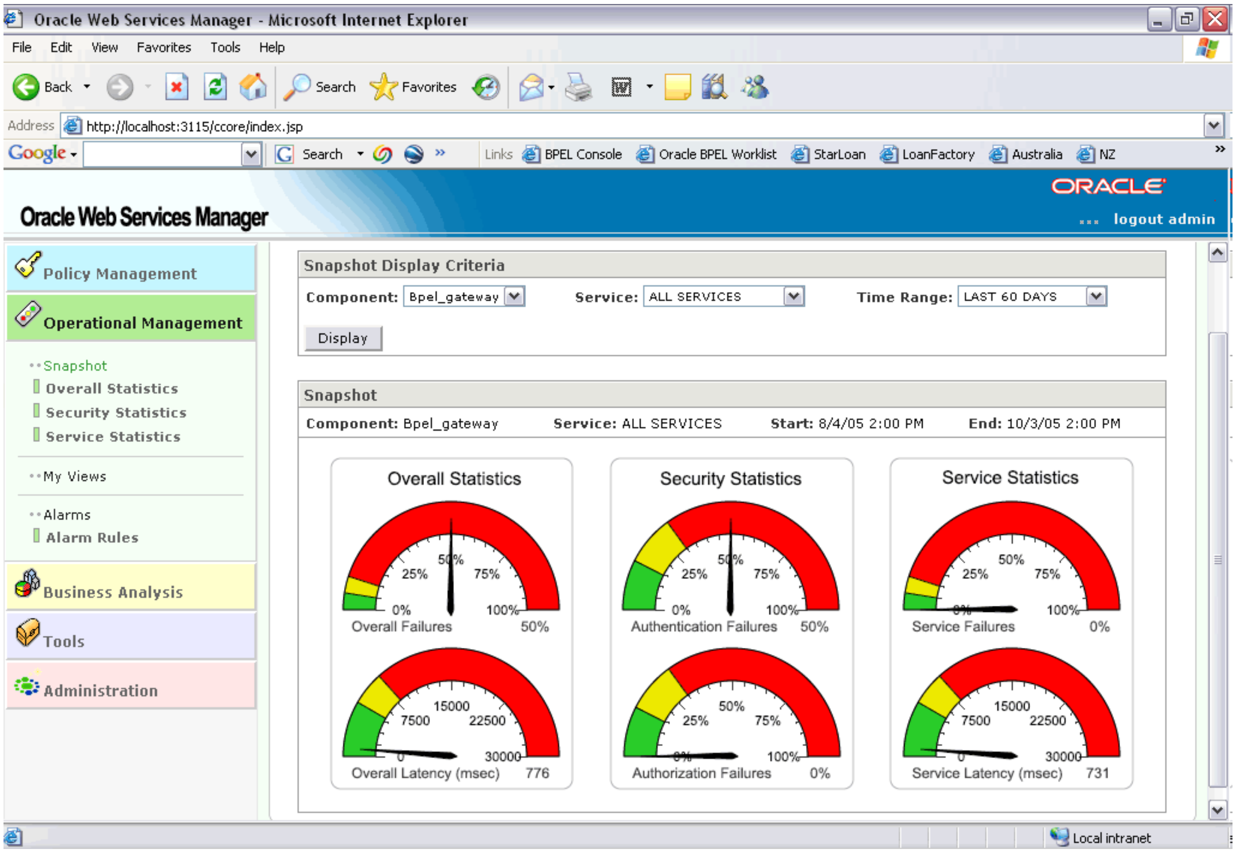
- Start Pipeline
- End Pipeline

Pipeline "Request"

Pipeline Steps:

- Start Pipeline
- Log
- XML Decrypt
- End Pipeline

Figure 3–5 Oracle Web Services Manager User Interface



Address: http://localhost:3115/ccore/index.jsp

Oracle Web Services Manager

Snapshot Display Criteria

Component: Bpel_gateway Service: ALL SERVICES Time Range: LAST 60 DAYS

Snapshot

Component: Bpel_gateway Service: ALL SERVICES Start: 8/4/05 2:00 PM End: 10/3/05 2:00 PM

Demonstrations and Tutorials

A series of demonstrations, activity and conceptual reference materials, and tutorials are also provided to increase conceptual knowledge and hands-on experience with Oracle BPEL Process Manager. These materials are installed with Oracle BPEL Process Manager in the `SOA_Oracle_Home\bpel\samples` directory.

> **See Also:** Chapter 1 of the *Oracle BPEL Process Manager Developer's Guide* for descriptions of the demonstrations, activity and conceptual reference materials, and tutorials provided with Oracle BPEL Process Manager

Index