

Oracle® Application Server

Adapter for Oracle Applications User's Guide

10g Release 2 (10.1.2)

B16498-02

December 2005

Oracle Application Server Adapter for Oracle Applications User's Guide, 10g Release 2 (10.1.2)

B16498-02

Copyright © 2004, 2005, Oracle. All rights reserved.

Primary Author: Sheela Vasudevan, Sumit Jeloka

Contributing Author: Michael Chiocca, Neeraj Chauhan, Neeraj Kumar, Rod Fernandez, Shashi Suravarapu, Veshaal Singh

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software—Restricted Rights (June 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Contents

Preface	vii
Audience	vii
Documentation Accessibility	vii
Structure	viii
Related Documents	ix
Conventions	ix
 1 Introduction	
1.1 Overview of Oracle Applications	1-1
1.2 Overview of OracleAS Adapter for Oracle Applications	1-1
1.2.1 Features	1-2
1.2.2 Architecture	1-2
1.2.3 Interfaces	1-3
1.3 What's New in this Release	1-3
1.3.1 Installing e-Business Suite Adapter	1-4
1.3.2 New Features in This Release	1-4
1.3.3 General Issues and Workarounds	1-4
 2 Using Tables and Views	
2.1 Interface Tables.....	2-1
2.1.1 Overview of Interface Tables	2-1
2.1.2 Design-Time Steps for Tables	2-2
2.1.2.1 Prerequisites to Configure Interface Tables.....	2-2
2.1.2.2 Configuring OracleAS Adapter for Oracle Applications	2-2
2.1.2.2.1 Creating a New BPEL Project	2-2
2.1.2.2.2 Adding a Partner Link.....	2-4
2.1.2.2.3 Configuring the Invoke Activity	2-11
2.1.2.2.4 Configuring the Assign Activity	2-13
2.1.3 Run-Time Steps for Tables	2-14
2.1.3.1 Deploying the BPEL Process.....	2-14
2.1.3.2 Testing the BPEL Process	2-16
2.2 Views	2-18
2.2.1 Overview of Views	2-18
2.2.2 Design-Time Steps for Views.....	2-19
2.2.2.1 Prerequisites to Configure Views.....	2-19

2.2.2.2	Configuring Oracle Adapter for Oracle Applications.....	2-19
2.2.2.2.1	Creating a New BPEL Project	2-19
2.2.2.2.2	Adding a Partner Link	2-21
2.2.2.2.3	Configuring the Invoke Activity	2-29
2.2.2.2.4	Configuring the Assign Activity	2-31
2.2.3	Run-Time Step for Views.....	2-32
2.2.3.1	Deploying the BPEL Process	2-32
2.2.3.2	Testing the BPEL Process	2-34

3 Using APIs

3.1	Overview of APIs.....	3-1
3.2	Design-Time Steps	3-1
3.2.1	Prerequisites to Configure APIs	3-1
3.2.2	Configuring OracleAS Adapter for Oracle Applications.....	3-2
3.2.2.1	Creating a New BPEL Project	3-2
3.2.2.2	Adding a New Partner Link	3-4
3.2.2.3	Describing Wrapper APIs	3-10
3.2.2.4	Describing Parameters With DEFAULT Clause	3-13
3.2.2.4.1	Describing DEFAULT Clause Handling in Wrapper Procedures.....	3-14
3.2.2.5	Configuring the Invoke Activity	3-15
3.2.2.6	Configuring the Transform Activity.....	3-16
3.3	Run-Time Steps	3-18
3.3.1	Deploying the BPEL Process	3-18
3.3.2	Testing the BPEL Process.....	3-20

4 Using Concurrent Programs

4.1	Overview of Concurrent Programs.....	4-1
4.2	Design-Time Steps	4-1
4.2.1	Prerequisites to Configuring OracleAS Adapter for Oracle Applications	4-1
4.2.2	Configuring OracleAS Adapter for Oracle Applications.....	4-2
4.2.2.1	Creating a New BPEL Project	4-2
4.2.2.2	Adding a Partner Link	4-4
4.2.2.3	Configuring the Invoke Activity	4-9
4.2.2.4	Configuring the Transform Activity.....	4-11
4.3	Run-Time Steps	4-14
4.3.1	Deploying the BPEL Process	4-14
4.3.2	Testing the BPEL Process.....	4-16
4.3.3	Verifying Records in Oracle Applications	4-18

5 Using e-Commerce Gateway

5.1	Overview of e-Commerce Gateway Integration	5-1
5.2	Design-Time Steps	5-2
5.2.1	Prerequisites to Configuring OracleAS Adapter for Oracle Applications	5-2
5.2.2	Configuring OracleAS Adapter for Oracle Applications.....	5-2
5.2.2.1	Creating a New BPEL Project	5-2
5.2.2.2	Adding a Partner Link	5-4

5.2.2.3	Configuring the Invoke Activity	5-10
5.2.2.4	Configuring the Transform Activity	5-12
5.3	Run-Time Steps	5-15
5.3.1	Deploying the BPEL Process	5-15
5.3.2	Testing the BPEL Process.....	5-17
5.3.3	Verifying Records in Oracle Applications	5-20

6 Using XML Gateway

6.1	Overview of XML Gateway.....	6-1
6.1.1	Standards-Based Messaging.....	6-1
6.1.2	Integration Architecture	6-1
6.1.3	Message Queues.....	6-2
6.1.3.1	Inbound Queues	6-2
6.1.3.2	Outbound Queues	6-3
6.1.4	XML Gateway Envelope	6-3
6.2	Design-Time Steps for XML Gateway Inbound into Oracle Applications.....	6-5
6.2.1	Prerequisites to Configuring OracleAS Adapter for Oracle Applications	6-5
6.2.2	Configuring OracleAS Adapter for Oracle Applications.....	6-5
6.2.2.1	Creating a New BPEL Project	6-5
6.2.2.2	Creating a Partner Link	6-8
6.2.2.3	Configuring the Invoke Activity	6-13
6.2.2.4	Adding a Partner Link for the File Adapter	6-17
6.2.2.5	Configuring the Receive Activity	6-20
6.2.2.6	Configuring the Assign Activity	6-22
6.3	Run-Time Steps for XML Gateway Inbound into Oracle Applications.....	6-24
6.3.1	Deploying the BPEL Process	6-25
6.3.2	Testing the BPEL Process.....	6-26
6.3.3	Verifying Records in Oracle Applications	6-30
6.3.3.1	Using Transaction Monitor	6-30
6.4	Design-Time Steps for XML Gateway Outbound from Oracle Applications	6-31
6.4.1	Prerequisites to Configuring OracleAS Adapter for Oracle Applications	6-31
6.4.2	Configuring OracleAS Adapter for Oracle Applications.....	6-31
6.5	Run-Time Steps for XML Gateway Outbound from Oracle Applications	6-32

A Sample WSDL and oc4j-ra.xml File

A.1	Sample WSDL File	A-1
A.2	Sample oc4j-ra.xml File	A-2

Index

Preface

This preface contains the following topics:

- [Audience](#)
- [Documentation Accessibility](#)
- [Structure](#)
- [Related Documents](#)
- [Conventions](#)

Audience

Oracle Application Server Adapter for Oracle Applications User's Guide is intended for those users who want to use the following interfaces to insert, retrieve, and update data in Oracle Applications:

- Interface tables
- Views
- Application programming interfaces (APIs)
- Concurrent Programs
- e-Commerce Gateway
- XML Gateway

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.

Structure

This document contains the following chapters:

Chapter 1, "Introduction"

This chapter provides an overview of Oracle Applications. In addition, it describes the features, interfaces, architecture, integration, and deployment of OracleAS Adapter for Oracle Applications.

Chapter 2, "Using Tables and Views"

This chapter describes how to insert data into Oracle Applications using interface tables and how to retrieve data from Oracle Applications using views.

Chapter 3, "Using APIs"

This chapter describes how to insert data into Oracle Applications using APIs.

Chapter 4, "Using Concurrent Programs"

This chapter describes how to use concurrent programs to move data from interface tables to base tables.

Chapter 5, "Using e-Commerce Gateway"

This chapter describes e-Commerce Gateway provides a common, standards-based approach for Electronic Data Interchange (EDI) integration between Oracle Applications and third party applications.

Chapter 6, "Using XML Gateway"

This chapter describes XML Gateway provides a common, standards-based approach for XML integration between Oracle Applications and third party applications, both inside and outside your enterprise.

Appendix A, "Sample WSDL and oc4j-ra.xml File"

This appendix shows a sample WSDL and `oc4j-ra.xml` file.

Related Documents

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, please 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/>

If you already have a user name 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/>

Conventions

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- [Conventions in Text](#)
- [Conventions in Code Examples](#)
- [Conventions for Windows Operating Systems](#)

Conventions in Text

We use the following conventions in text to help you more quickly identify special terms. The table also provides examples of their use.

Convention	Meaning	Example
Bold	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an index-organized table .
<i>Italics</i>	Italic typeface indicates book titles or emphasis.	<i>Oracle Database 10g Concepts</i> Ensure that the recovery catalog and target database do <i>not</i> reside on the same disk.
UPPERCASE monospace (fixed-width) font	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, Recovery Manager keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, user names, and roles.	You can specify this clause only for a NUMBER column. You can back up the database by using the BACKUP command. Query the TABLE_NAME column in the USER_TABLES data dictionary view. Use the DBMS_STATS.GENERATE_STATS procedure.

Convention	Meaning	Example
lowercase monospace (fixed-width) font	Lowercase monospace typeface indicates executable programs, filenames, directory names, and sample user-supplied elements. <i>Note:</i> Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter <code>sqlplus</code> to start SQL*Plus. The password is specified in the <code>orapwd</code> file. Back up the datafiles and control files in the <code>/disk1/oracle/dbs</code> directory. The <code>department_id</code> , <code>department_name</code> , and <code>location_id</code> columns are in the <code>hr.departments</code> table. Connect as <code>oe</code> user. The <code>JRepUtil</code> class implements these methods.
<i>lowercase italic monospace (fixed-width) font</i>	Lowercase italic monospace font represents placeholders or variables.	You can specify the <i>parallel_clause</i> . Run <i>old_release</i> .SQL where <i>old_release</i> refers to the release you installed prior to upgrading.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[]	Anything enclosed in brackets is optional.	DECIMAL (<i>digits</i> [, <i>precision</i>])
{ }	Braces are used for grouping items.	{ENABLE DISABLE}
	A vertical bar represents a choice of two options.	{ENABLE DISABLE} [COMPRESS NOCOMPRESS]
...	Ellipsis points mean repetition in syntax descriptions. In addition, ellipsis points can mean an omission in code examples or text.	CREATE TABLE ... AS <i>subquery</i> ; SELECT <i>col1</i> , <i>col2</i> , ... , <i>coln</i> FROM employees;
Other symbols	You must use symbols other than brackets ([]), braces ({ }), vertical bars (), and ellipsis points (...) exactly as shown.	<code>acctbal</code> NUMBER(11,2); <code>acct</code> CONSTANT NUMBER(4) := 3;
<i>Italics</i>	Italicized text indicates placeholders or variables for which you must supply particular values.	CONNECT SYSTEM/ <i>system_password</i> DB_NAME = <i>database_name</i>
UPPERCASE	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. Because these terms are not case sensitive, you can use them in either UPPERCASE or lowercase.	SELECT last_name, employee_id FROM employees; SELECT * FROM USER_TABLES; DROP TABLE hr.employees;

Convention	Meaning	Example
lowercase	Lowercase typeface indicates user-defined programmatic elements, such as names of tables, columns, or files. Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	SELECT last_name, employee_id FROM employees; sqlplus hr/hr CREATE USER mjones IDENTIFIED BY ty3MU9;

Conventions for Windows Operating Systems

The following table describes conventions for Windows operating systems and provides examples of their use.

Convention	Meaning	Example
Click Start , and then choose the <i>menu item</i>	How to start a program.	To start the Database Configuration Assistant, click Start , and choose Programs . In the Programs menu, choose Oracle - HOME_NAME and then click Configuration and Migration Tools . Choose Database Configuration Assistant .
File and directory names	File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (<), right angle bracket (>), colon (:), double quotation marks ("), slash (/), pipe (), and dash (-). The special character backslash (\) is treated as an element separator, even when it appears in quotes. If the filename begins with \\, then Windows assumes it uses the Universal Naming Convention.	c:\winnt\\"system32 is the same as C:\WINNT\SYSTEM32
C:\>	Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the <i>command prompt</i> in this manual.	C:\oracle\oradata>
Special characters	The backslash (\) special character is sometimes required as an escape character for the double quotation mark (") special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.	C:\>exp HR/HR TABLES=employees QUERY=\"WHERE job_id='SA_REP' and salary<8000\"
HOME_NAME	Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.	C:\> net start OracleHOME_NAME\TNSListener

Convention	Meaning	Example
<i>ORACLE_HOME</i> and <i>ORACLE_BASE</i>	<p>In releases prior to Oracle8i release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level <i>ORACLE_HOME</i> directory.</p> <p>This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level <i>ORACLE_HOME</i> directory. There is a top level directory called <i>ORACLE_BASE</i> that by default is C:\oracle\product\10.1.0. If you install the latest Oracle release on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is C:\oracle\product\10.1.0\db_n, where <i>n</i> is the latest Oracle home number. The Oracle home directory is located directly under <i>ORACLE_BASE</i>.</p> <p>All directory path examples in this guide follow OFA conventions.</p> <p>Refer to <i>Oracle Database Installation Guide for Windows</i> for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.</p>	Change to the <i>ORACLE_BASE\ORACLE_HOME\rdms\admin</i> directory.

Introduction

This chapter provides an overview of Oracle Applications and Oracle Application Server Adapter for Oracle Applications (OracleAS Adapter for Oracle Applications). It contains the following sections:

- [Overview of Oracle Applications](#)
- [Overview of OracleAS Adapter for Oracle Applications](#)
- [What's New in this Release](#)

1.1 Overview of Oracle Applications

Oracle Applications is a set of integrated business applications that runs entirely on the Internet. Oracle Applications offers you the following:

- Reduced costs
- Increased revenue across front-office and back-office functions
- Access to current, accurate, and consistent data

The applications in Oracle Applications are built on a unified information architecture that consolidates data from Oracle and non-Oracle applications and enables a consistent definition of customers, suppliers, partners, and employees across the entire enterprise. This results in a suite of applications that can give you information, such as current performance metrics, financial ratios, profit and loss summaries. To connect Oracle Applications to other non-Oracle applications, you use OracleAS Adapter for Oracle Applications.

1.2 Overview of OracleAS Adapter for Oracle Applications

OracleAS Adapter for Oracle Applications provides comprehensive, bidirectional, multimodal, synchronous, and asynchronous connectivity to Oracle Applications. The adapter supports all modules of Oracle Applications for versions 11.5.1 to 11.5.10. In addition, OracleAS Adapter for Oracle Applications provides real-time and bidirectional connectivity to Oracle Applications through interface tables, views, Application programming interfaces (APIs), concurrent programs, e-commerce Gateway, and XML Gateway.

OracleAS Adapter for Oracle Applications inserts data into Oracle Applications using interface tables, APIs, and concurrent programs. To retrieve data from Oracle Applications, OracleAS Adapter for Oracle Applications uses views. In addition, it uses XML Gateways for bidirectional integration with Oracle Applications. XML Gateways are also used to insert as well as receive Open Application Group Integration Specification (OAGIS) compliant documents from Oracle Applications.

Note: The XML Gateway feature is available in the preview mode of this release. To run the preview mode, run the following command from \$ORACLE_HOME\integration\jdev\jdev\bin:

```
Jdev.exe -J"-Dpreview_mode=true"
```

This section contains the following topics:

- [Features](#)
- [Architecture](#)
- [Interfaces](#)

1.2.1 Features

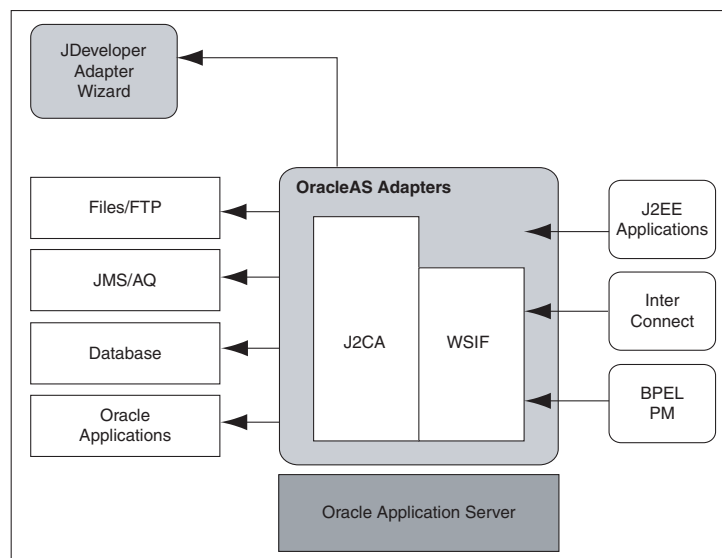
OracleAS Adapter for Oracle Applications consists of the following features:

- Supports open standards, such as J2EE Connector Architecture (J2CA), Extensible Markup Language (XML), Web Service Invocation Framework (WSIF), Web Service Inspection Language (WSIL), and Web Service Definition Language (WSDL)
- Uses JDeveloper based design-time tool for dynamically browsing the Oracle Applications interface and configuring the adapter metadata
- Integrates applications based on open standards, such as OAGIS, by interfacing with the XML Gateway component of the Oracle Applications product
- Generates adapter metadata as WSDL files with J2CA extension

See Also: *Oracle Application Server Adapter Concepts* for more information

1.2.2 Architecture

OracleAS Adapter for Oracle Applications is based on J2CA 1.0 standards and deployed as a resource adapter in the same Oracle Application Server Containers for J2EE (OC4J) container as BPEL Process Manager. The architecture of OracleAS Adapter for Oracle Applications is similar to the architecture of technology adapters. [Figure 1-1](#) illustrates the architecture of OracleAS Adapter for Oracle Applications.

Figure 1–1 OracleAS Adapter for Oracle Applications Architecture

1.2.3 Interfaces

OracleAS Adapter for Oracle Applications acts as a highly flexible integration interface for Oracle Applications. The adapter provides the following interfaces to integrate with Oracle Applications:

- **Interface tables:** Enable you to insert or update data into Oracle Applications. The associated concurrent program should be running to move the data from the interface tables to base tables.
- **Views:** Help you to retrieve data from Oracle Applications.
- **APIs:** Enable you to insert and update data into Oracle Applications.
- **Concurrent Programs:** Enables you to move data from interface tables to base tables.
- **e-Commerce Gateway:** Provides you a common, standards-based approach for Electronic Data Interchange (EDI) integration between Oracle Applications and third party applications.
- **XML Gateway:** Enables bidirectional integration with Oracle Applications. It helps you to insert and retrieve data from Oracle Applications. XML Gateway is a higher-level API that exposes OAGIS-formatted XML documents for commonly used Oracle Application business objects and business interfaces. XML Gateway integrates with interface tables, Business Event System (BES), and views to insert and retrieve data from Oracle Applications. It maps the underlying table data to XML and back.

1.3 What's New in this Release

This section describes the new integration interface types that have been added, and issues associated with the Oracle e-Business Suite Adapter. This section includes the following topics:

- [Installing e-Business Suite Adapter](#)
- [New Features in This Release](#)

- [General Issues and Workarounds](#)

1.3.1 Installing e-Business Suite Adapter

This section describes the installation of the e-Business Suite Adapter.

The installation of the Oracle e-Business Suite Adapter happens out-of-the-box with the Oracle BPEL PM product. OracleAS adapter for Oracle Applications is deployed using the Oracle BPEL Process Manager (PM) in Oracle JDeveloper.

See Also: *Oracle Application Server Integration Business Activity Monitoring User's Guide* for more details about installing Oracle BPEL Process Manager. Refer to section, *Notes on Installing Oracle BPEL Process Manager*.

1.3.2 New Features in This Release

This section describes the new features that have been added in 10.1.2 phase 2.

The Oracle Application Server Integration InterConnect has added the following three new integration interface types to the existing list:

- [Concurrent Programs](#)
- [XML Gateway](#)
- [e-Commerce Gateway](#)

The new integration interfaces are exposed as Web services, and these Web services are available for process orchestration through the Oracle BPEL Process Manager.

Concurrent Programs

A concurrent program is an instance of an execution file, along with parameter definitions and incompatibilities. Concurrent programs use concurrent program executable to locate the correct execution file. OracleAS Adapter for Applications can be configured to use the concurrent program integration interface to perform specific tasks.

XML Gateway

Oracle XML gateway is a set of services that allow easy integration with Oracle Applications to support XML messaging. OracleAS Adapter for Applications can be configured to use the XML gateway integration interface to interact for inbound or outbound transactions with third-party applications.

e-Commerce Gateway

Oracle e-commerce gateway is the Electronic Data Interchange (EDI) integration enabler between Oracle Applications and third-party applications based on a common, standards-based approach. OracleAS Adapter for Applications can be configured to use the Oracle e-Commerce Gateway integration interface to dynamically generate outbound or consume inbound flat files based on user-defined trading partner, mapping, transformation, and data validation rules.

1.3.3 General Issues and Workarounds

This section describes the following issues and workarounds:

- In the Oracle Applications 11.5.10 release, only concurrent programs, which have PL/SQL stored procedure as the execution method, would be supported. This is only a design time browsing functionality as provided by the IRep based browser.
- For Oracle Applications 11.5.9 and lower versions, the adapter wizard shows all concurrent programs both private and public. However, from 11.5.10 onwards, only those concurrent programs that are classified as public are shown. This is the intended functionality.
- Due to a known issue with the adapter wizard not being able to preserve DEFAULT clauses for PL/SQL wrappers that it generates underneath the covers, you have to adopt the following workaround when working with concurrent program and EDI interfaces to Oracle Applications. You *must* load SQL file (apps_cp_ecx_wrapper.sql) once into the apps schema (through SQL*Plus) before launching the adapter wizard to create services for either concurrent programs or EDI programs. You will find the file, apps_cp_ecx_wrapper.sql at the following location:

`ORACLE_HOME\integration\orabpel\system\database\scripts\apps_cp_ecx_wrapper.sql`

Note: Please do not modify the contents of the SQL file (XX_BPPEL_FND_REQUEST_SUBMIT_REQ.sql)

Using Tables and Views

OracleAS Adapter for Oracle Applications use interface tables to insert and update data in Oracle Applications, and uses views to retrieve data from Oracle Applications.

This chapter describes the following interfaces:

- [Interface Tables](#)
- [Views](#)

2.1 Interface Tables

OracleAS Adapter for Oracle Applications use interface tables to insert data in Oracle Applications. For example, by using interface tables, you can insert a purchase order into Oracle Applications to generate the sales order automatically. Data is never loaded directly into Oracle Applications base tables. Instead, data is first loaded into interface tables, and then Oracle-supplied concurrent programs move data from interface tables to base tables. This ensures that all business logic and processing is handled using Oracle components.

This section contains the following topics:

- [Overview of Interface Tables](#)
- [Design-Time Steps for Tables](#)
- [Run-Time Steps for Tables](#)

2.1.1 Overview of Interface Tables

OracleAS Adapter for Oracle Applications use open interface tables to integrate with Oracle Applications through direct database access. The OracleAS Adapter for Oracle Applications inserts data into the open interface tables. These interface tables can be used only for insert operations and support only an inbound connection into Oracle Applications.

Interface tables are intermediate tables into which the data is inserted first. Once the data gets inserted into the interface tables, the data is validated, and then transferred to the base tables. Base tables are real application tables that reside in the application database. The data that resides in the interface tables is transferred to the base tables using concurrent programs. A concurrent program is an instance of an execution file, along with parameter definitions and incompatibilities. Concurrent programs are scheduled in Oracle Applications to move data from interface tables to base tables. These programs perform the application-level checks and run validation before inserting data into base tables.

2.1.2 Design-Time Steps for Tables

This section describes how to configure the OracleAS Adapter for Oracle Applications to use interface tables. It includes the following topics:

- [Prerequisites to Configure Interface Tables](#)
- [Configuring OracleAS Adapter for Oracle Applications](#)

2.1.2.1 Prerequisites to Configure Interface Tables

The prerequisites to configure interface tables include the following:

- Define primary keys on all the interface tables being used
- Define parent-child relationships among all the interface tables used.

2.1.2.2 Configuring OracleAS Adapter for Oracle Applications

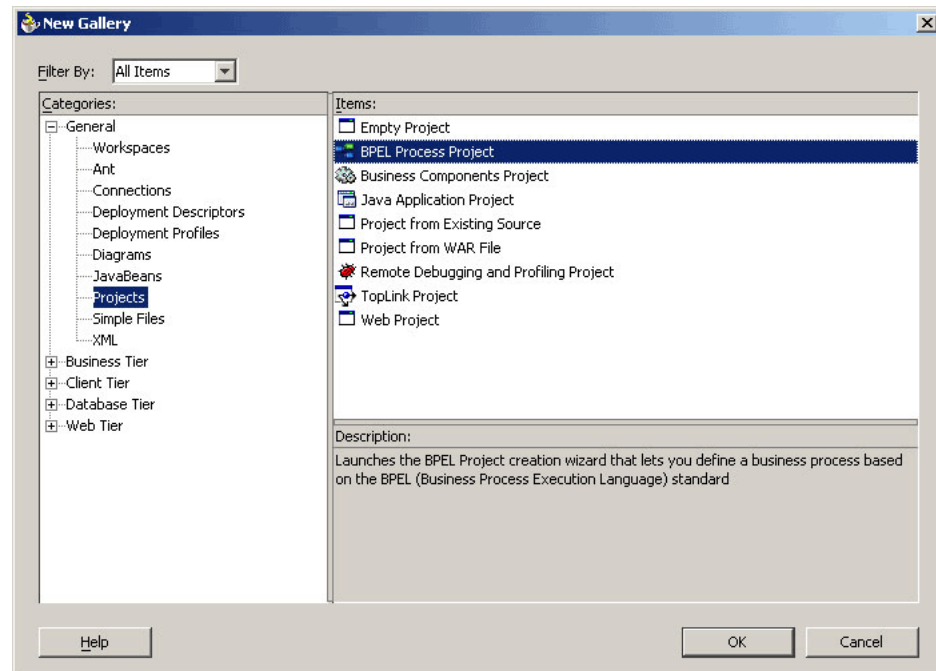
This section describes the steps to configure OracleAS Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper. It includes the following topics:

- [Creating a New BPEL Project](#)
- [Adding a Partner Link](#)
- [Configuring the Invoke Activity](#)
- [Configuring the Assign Activity](#)

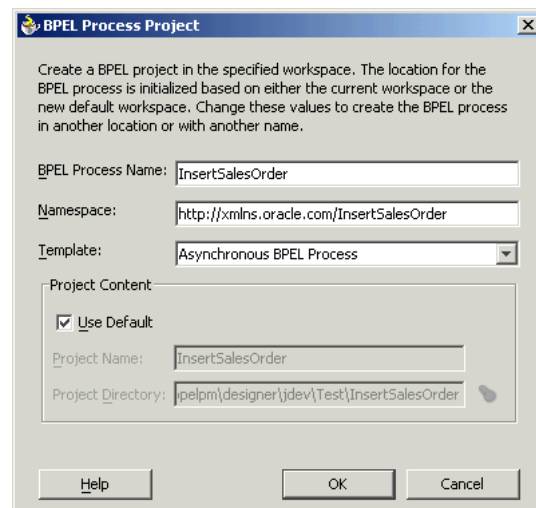
2.1.2.2.1 Creating a New BPEL Project

To create a new BPEL project:

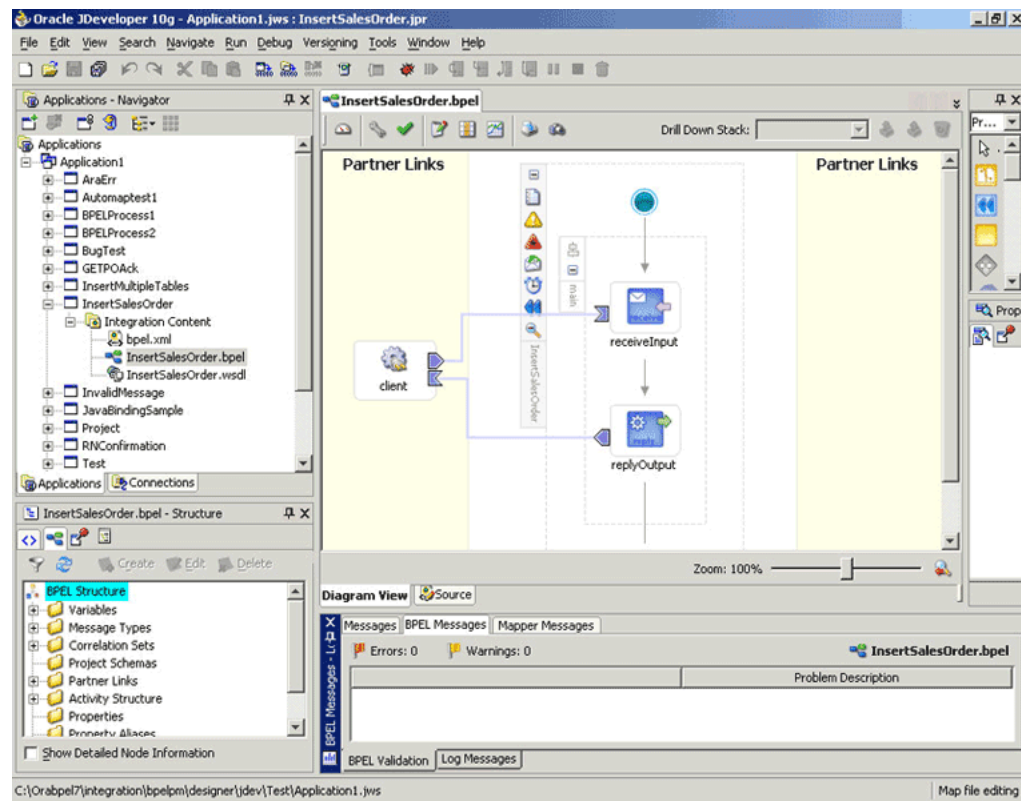
1. Open JDeveloper BPEL Designer.
2. From the **File** menu, select **New**. The New Gallery dialog box is displayed.
3. Select **All Items** from the **Filter By** list. This displays a list of available categories.
4. Expand the **General** node, and then select **Projects**.
5. Select **BPEL Process Project** from the **Items** list, as shown in [Figure 2-1](#).

Figure 2–1 Creating a New BPEL Process Project

6. Click **OK**. The BPEL Process Project dialog box is displayed.
7. In the **BPEL Process Name** field, enter a descriptive name. For example, `InsertSalesOrder`.
8. From the **Template** list, select **Asynchronous BPEL Process**. Keep the default selection for Use Default in the Project Content section, as shown in [Figure 2–2](#).

Figure 2–2 Specifying a Name for the New BPEL Process Project

9. Click **OK**. A new asynchronous BPEL process, with the required source files including `bpel.xml`, `InsertSalesOrder.xml`, and `InsertSalesOrder.wsdl`, is created, as shown in [Figure 2–3](#).

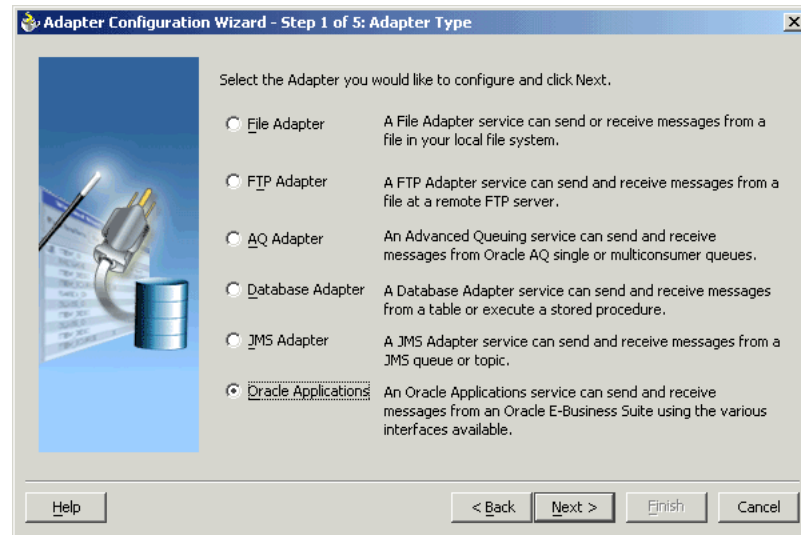
Figure 2–3 New BPEL Process Project

2.1.2.2.2 Adding a Partner Link

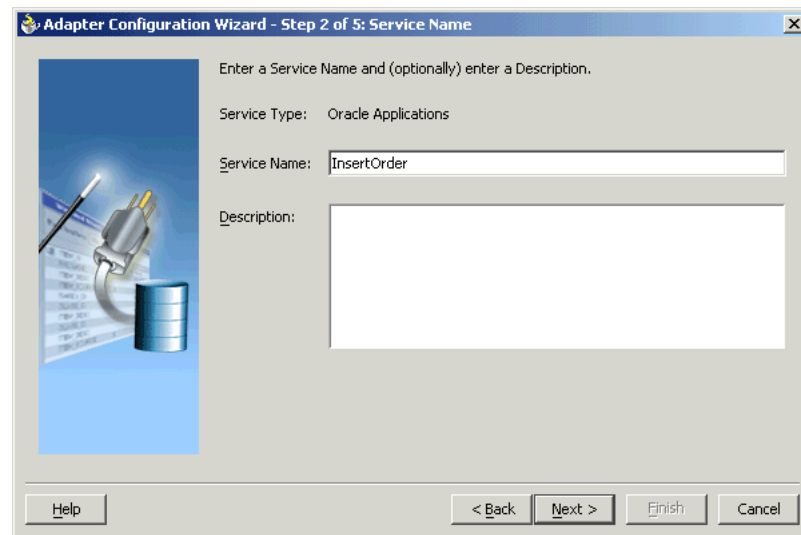
This section describes how to add a partner link to your BPEL process. A partner link defines the link name, type, and the role of the BPEL process that interacts with the partner service.

To add a partner link:

1. Drag and drop **PartnerLink** into the border area of the process diagram. The Create Partner Link dialog box is displayed.
2. Click **Define Adapter Service** in the WSDL Settings section. The Adapter Configuration Wizard is displayed.
3. Click **Next**. The Adapter Type dialog box is displayed.
4. Select **Oracle Applications** to specify the adapter you want to configure, as shown in [Figure 2–4](#).

Figure 2–4 Selecting OracleAS Adapter for Oracle Applications

5. Click **Next**. The Service Name dialog box is displayed.
6. Enter the following information:
 1. In the **Service Name** field, enter a service name.
 2. In the **Description** field, enter a description for the service. This is an optional field. The Service Name dialog box is displayed, as shown in [Figure 2–5](#).

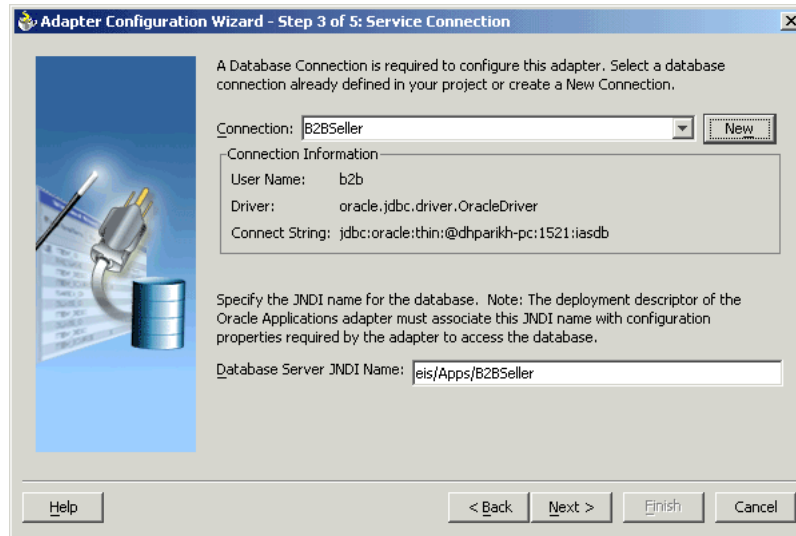
Figure 2–5 Specifying the Service Name

7. Click **Next**. The Service Connection dialog box is displayed.
8. Enter the Java Naming and Directory Interface (JNDI) name in the **Database Server JNDI Name** field. The JNDI name acts as a placeholder for the connection used when your service is deployed to the BPEL server. This enables you to use different databases for development and later production.

See Also: *Oracle Application Server Adapter Concepts* for understanding JNDI concepts

Figure 2–6 shows how to create a new database connection.

Figure 2–6 Creating a New Database Connection

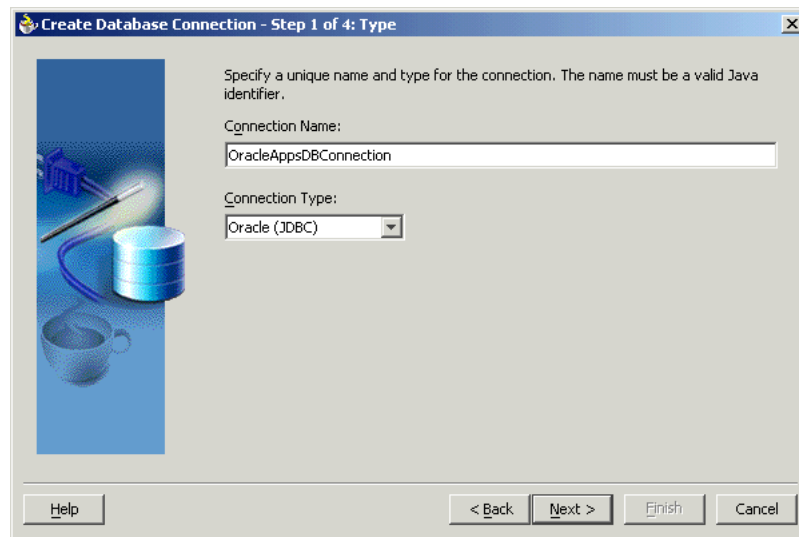


9. Click **New** to define a database connection. The Create Database Connection Wizard is displayed.

Note: You need to connect to the database where Oracle Applications is running.

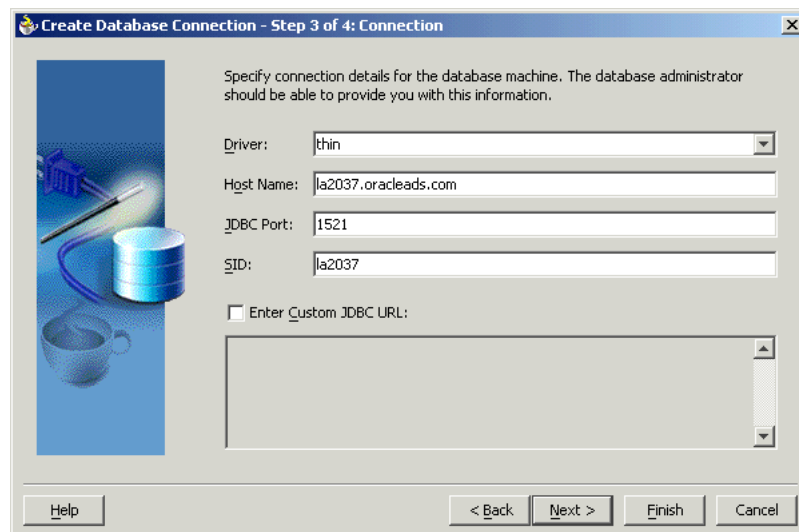
10. Enter the following information in the Type dialog box:
 1. In the **Connection Name** field, specify a unique name for the database connection.
 2. From the **Connection Type** list, select the type of connection for your database connection.

The Type dialog box is displayed in Figure 2–7.

Figure 2–7 Specifying the Connection Name and Type of Connection

11. Click **Next**. The Authentication dialog box is displayed.
12. Enter information in the following fields:
 - a. In the **UserName** field, specify a unique name for the database connection.
 - b. In the **Password** field, specify a password for the database connection.
13. Click **Next**. The Connection dialog box is displayed.
14. Enter information in the following fields:
 - a. From the **Driver** list, select **Thin**.
 - b. In the **Host Name** field, specify the host name for the database connection.
 - c. In the **JDBC Port** field, specify the port number for the database connection.
 - d. In the **SID** field, specify a unique SID value for the database connection.

The Connection dialog box is displayed, as shown in [Figure 2–8](#).

Figure 2–8 Specifying the New Database Connection Information

15. Click **Next**. The Test dialog box is displayed.
16. Click **Test Connection** to determine whether the specified information establishes a connection with the database.
17. Click **Next**. The Service Connection dialog box is displayed, providing a summary of the database connection.
18. Click **Finish** to complete the process of creating a new database connection.

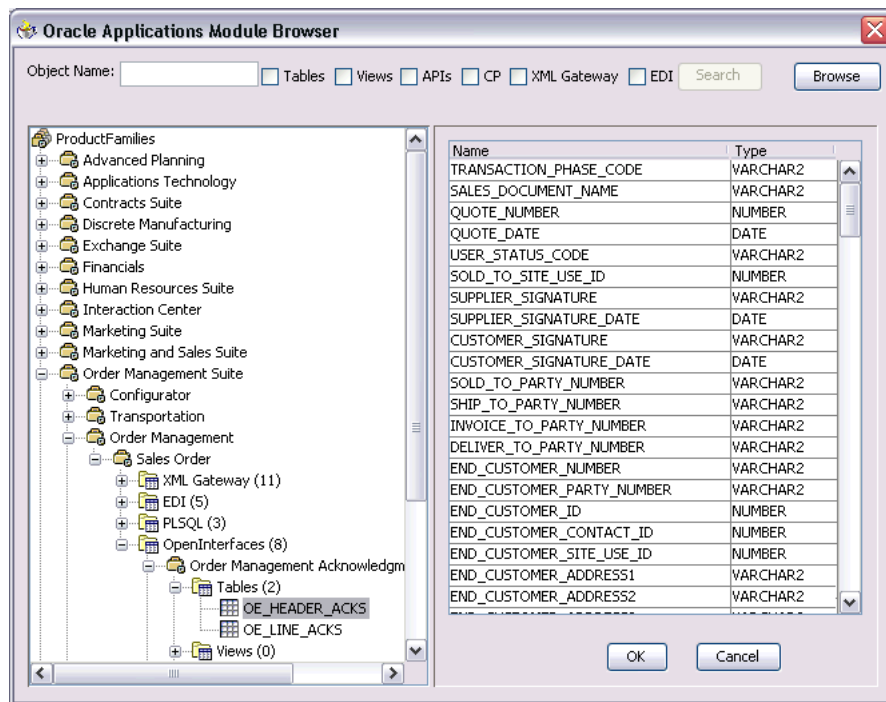
Once you have completed creating a new connection for the service, you can add an interface table by browsing through the maps available in Oracle Applications.

19. Click **Next** in the Service Connection dialog box. The Operation dialog box is displayed.

Note: If you are connecting to a pre-11.5.10 Oracle Applications instance, then you would be required to select the interface to Oracle Applications data. Select **Tables/Views/APIs/Concurrent Programs** to proceed.

20. Click **Get Object** to open the Oracle Applications Module Browser. [Figure 2–9](#) shows the Oracle Applications Module Browser.

Figure 2–9 Adding a Table from the Oracle Applications Module Browser



Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. For example, Applications Technology or Order Management Suite are product families in Oracle Applications. The product families contain the individual products. For example, Order Management Suite contains the Order Management product. The product contains the business entities associated with the product. For example, the Order Management product contains the Sales Order business entity.

Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide. Tables can be found under the OpenInterfaces category.

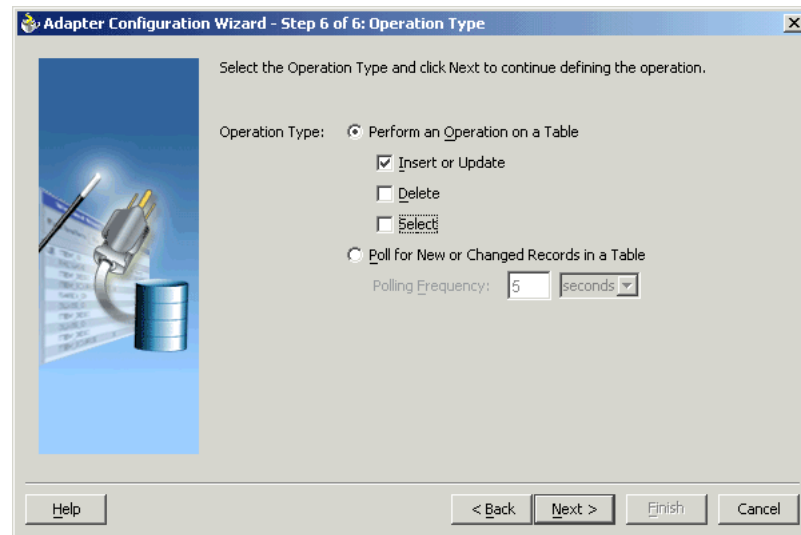
21. Select a table, and then click **OK**.

Note: You can also search for a table by entering the name of the program in the **Object Name** field. Select the **Tables** check box, and then click **Search**.

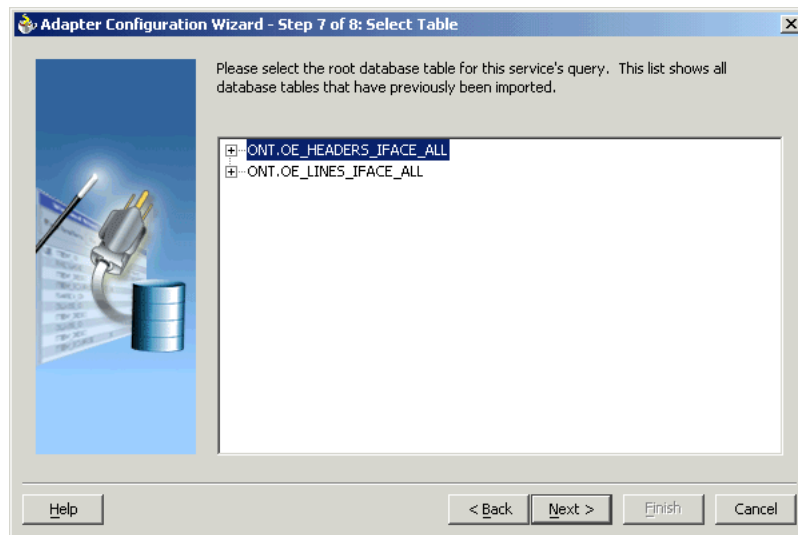
22. The table is added to the Operation Objects. Repeat steps 2 and 3 to add more tables.
23. Click **Next** in the Operation dialog box. The Operation Type dialog box is displayed.
24. Select **Perform an Operation on a Table**, and then select **Insert or Update**, as shown in [Figure 2-10](#).

Note: You can only insert data into the interface tables even though the Delete and Select options are enabled.

Figure 2-10 *Selecting the Type of Operation*

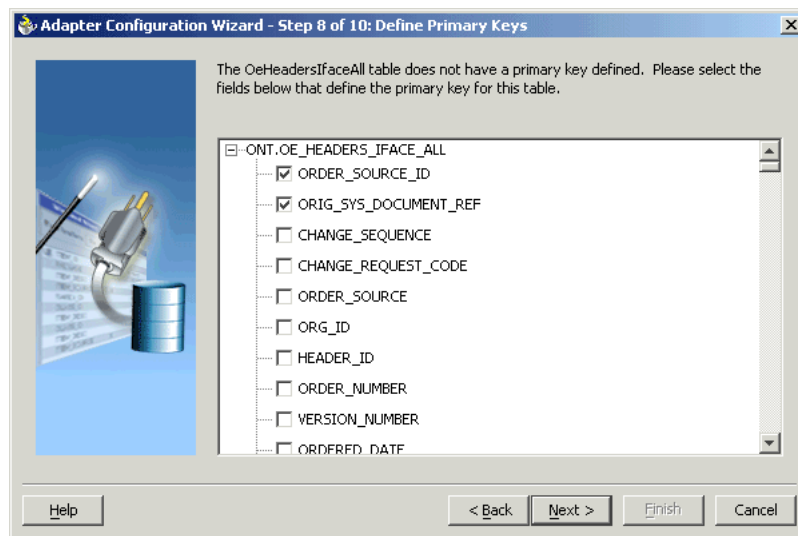


25. Click **Next**. The Select Table screen is displayed.
26. Select the root database table in the Select Table screen, which displays the tables that have been previously imported in this JDeveloper project (including tables that were imported for other partner links). This enables you to reuse configured table definitions in multiple partner links. [Figure 2-11](#) shows selecting a root table.

Figure 2–11 Selecting a Root Table

Note: If the required root database table for the operation is not imported, then click **Import Tables**. In addition, you can reimport a table and overwrite the previously configured table definition. However, in this case, you will lose any custom relationships that you may have defined on that table as well as any WHERE clause if a root table is imported.

27. Click **Next**. The Define Primary Keys screen is displayed.
28. Select the primary key fields, as shown in [Figure 2–12](#). You can also select multiple fields.

Figure 2–12 Selecting the Primary Key

29. Click **Next**. The Relationships screen is displayed.
30. Click **Create Relationship** to define a new relationship. The Create Relationship dialog box is displayed.

Note: If foreign key constraints between tables already exist in the database, then two relationships are created automatically while importing tables. One of the relationships is 1:M relationship from the source table, which is the table containing the foreign key constraints, to the target table. The other relationship is a 1:1 back pointer from the target table to the source table.

31. Specify the following information to create a new relationship:
 - a. Select the parent and child tables.
 - b. Select the mapping type (1:M, 1:1, or 1:1 with Foreign Key on Child Table).
 - c. Associate the foreign key fields to the primary key fields.
 - d. Enter a name for the relationship you are creating. It is optional to specify a name. By default, a name is generated for the relationship.

Note: You can select only those tables as parent tables that can be accessed from the root table.

The Create Relationship dialog box is displayed in [Figure 2–13](#).

Figure 2–13 Defining Relationships Between Parent and Child Tables

Please specify the parent and child tables, relation type and relation name. You also must specify the foreign key / primary key associations in the table below.

Parent Table:

Child Table:

☐ OE_HEADERS_IFACE_ALL has a 1:1 Relationship with OE_LINES_IFACE_ALL
☐ OE_HEADERS_IFACE_ALL has a 1:1 Relationship with OE_LINES_IFACE_ALL (Foreign Key on Child table)
☒ OE_HEADERS_IFACE_ALL has a 1:M Relationship with OE_LINES_IFACE_ALL

OE_HEADERS_IFACE_ALL	OE_LINES_IFACE_ALL
ORDER_SOURCE_ID	<input type="text" value="ORDER_SOURCE_ID"/>
ORIG_SYS_DOCUMENT_REF	<input type="text" value="ORIG_SYS_DOCUMENT_REF"/>

Relationship Name:

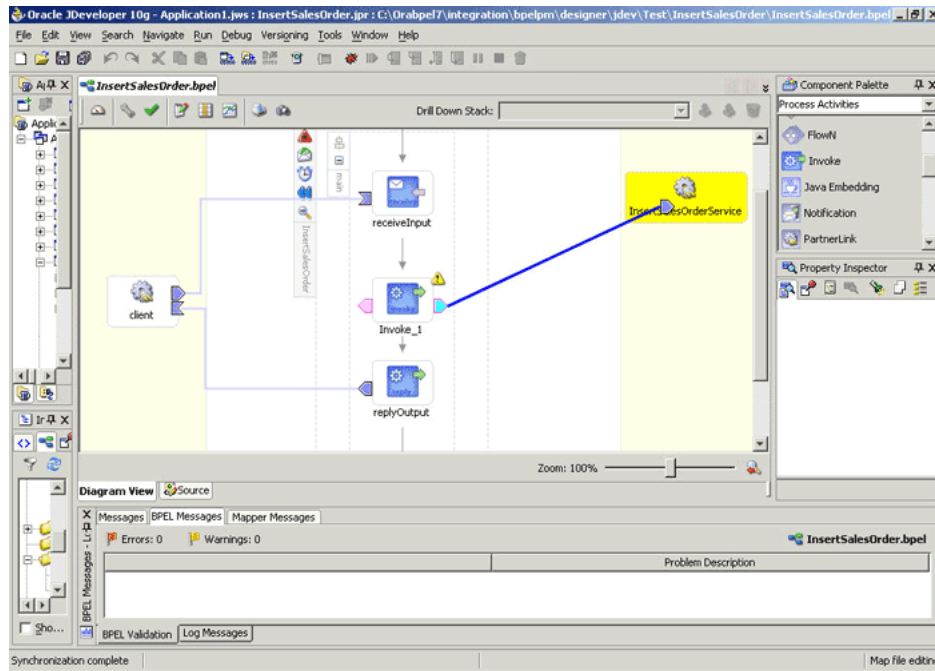
32. Click **OK**. The Relationships screen is displayed.
33. Click **Next**, and then click **Finish** to complete the process of configuring OracleAS Adapter for Oracle Applications.

After adding and configuring the partner link, the next task is to configure the BPEL process.

2.1.2.2.3 Configuring the Invoke Activity

1. Drag **Invoke** from the Component palette and drop it at the location where you want to insert the invoke activity in your BPEL process, as shown in [Figure 2–14](#).

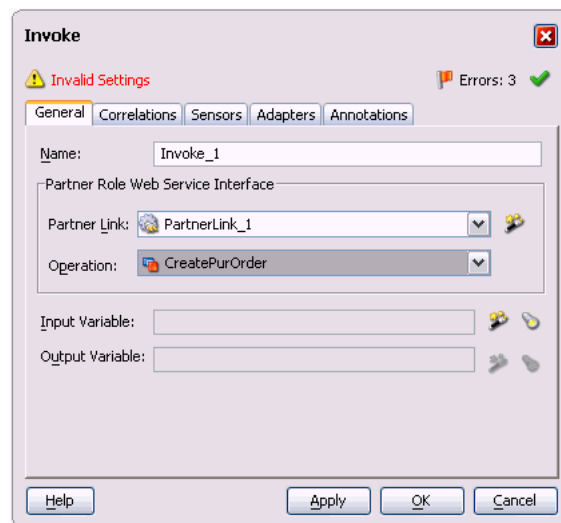
Figure 2–14 Dragging an Invoke Activity



The Edit Invoke dialog box is displayed.

2. Double-click **Invoke** in the process map to open the Invoke dialog box. The General tab is selected by default. [Figure 2–15](#) shows the Invoke dialog box.

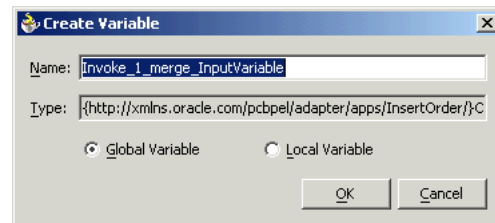
Figure 2–15 Configuring the Invoke Activity



3. In the **Partner Link** box, select the partner link to invoke. This is the partner link that you configured in the previous section.

- Click the **Create** icon next to the **Input Variable** field. Enter a descriptive name for the variable in the Create Variable dialog box that appears. You can also accept the default name. Click **OK**. [Figure 2–16](#) shows the Create Variable dialog box.

Figure 2–16 Creating a Variable



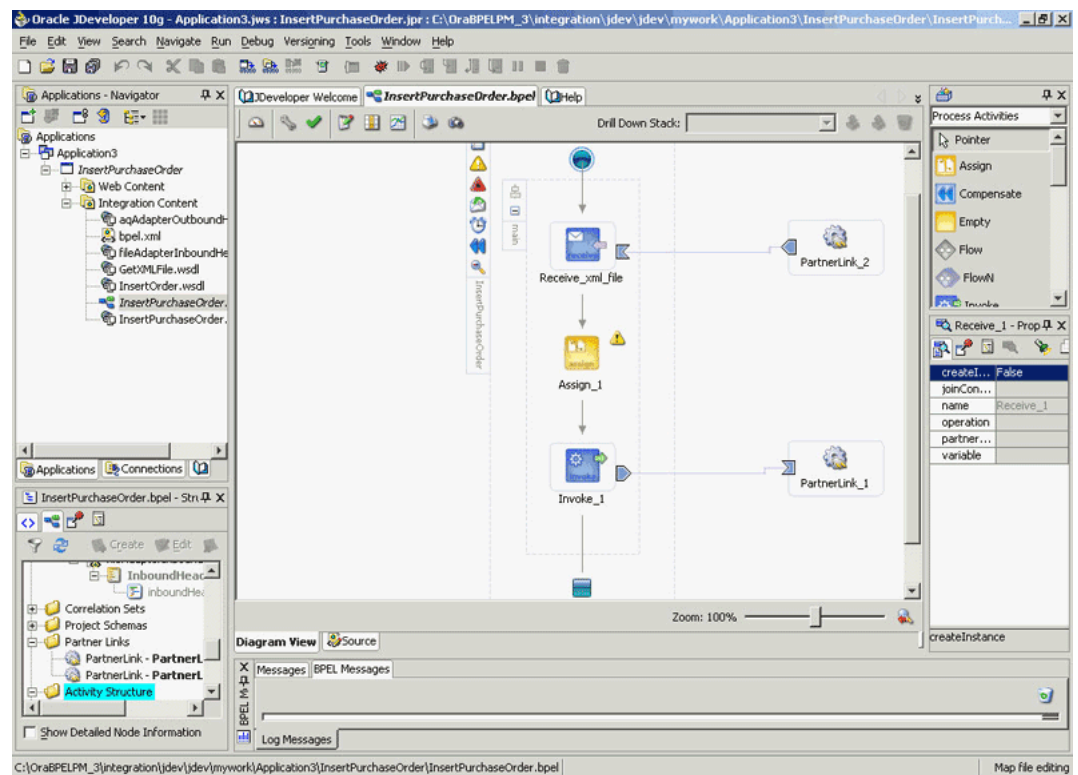
- In the Invoke dialog box, click **Apply**, and then click **OK**.

2.1.2.2.4 Configuring the Assign Activity

The next task is to add an Assign activity to the process map. This is used to provide values to the input variables. The following steps discuss this task:

- Drag and drop the Assign activity to the process map. The Assign activity must be dropped in between the Receive and Invoke activities. [Figure 2–17](#) shows the Assign activity after it has been added to the process map.

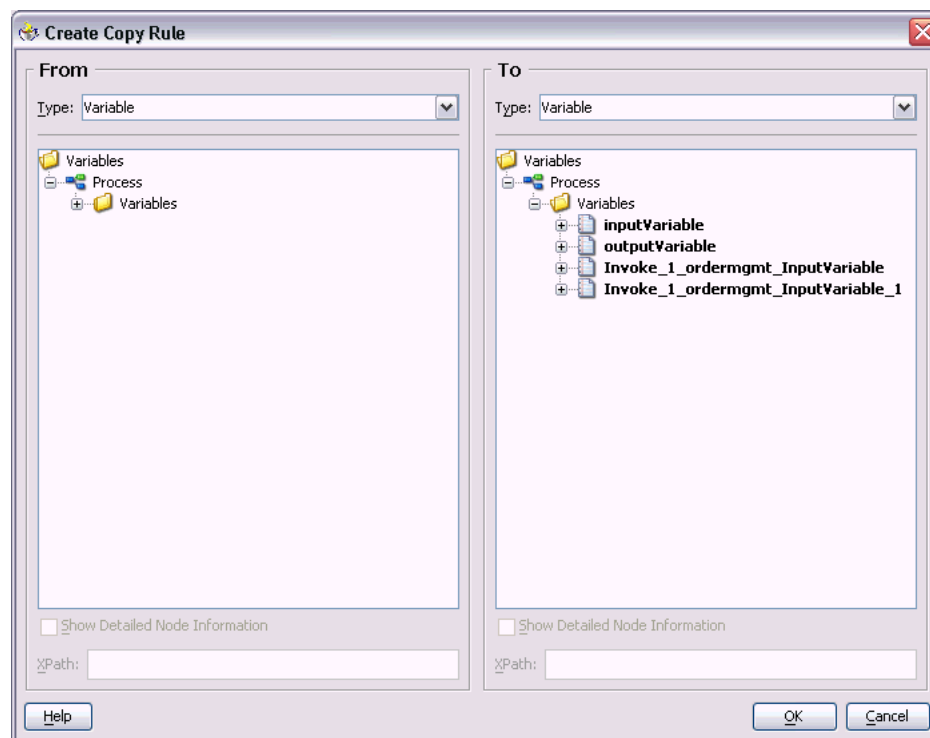
Figure 2–17 Adding the Assign Activity



- You now need to configure the Assign activity. Double-click the **Assign** activity in the process map.
- The Assign dialog box is displayed. The Copy Rules tab is displayed by default. Click **Create**.

4. The Create Copy Rule dialog box is displayed. In the To group, expand the Variables node by clicking the plus sign next to it, and then select **Expression** from the From group to assign values to the input variables. [Figure 2–18](#) shows assigning values to the input variables.

Figure 2–18 Assigning Values to Input Variables



5. After assigning values to the input variables, click **OK**.
6. Select **Make** from the Run menu or press Ctrl+F9 to compile the BPEL process to check for errors. The compilation result is displayed.

2.1.3 Run-Time Steps for Tables

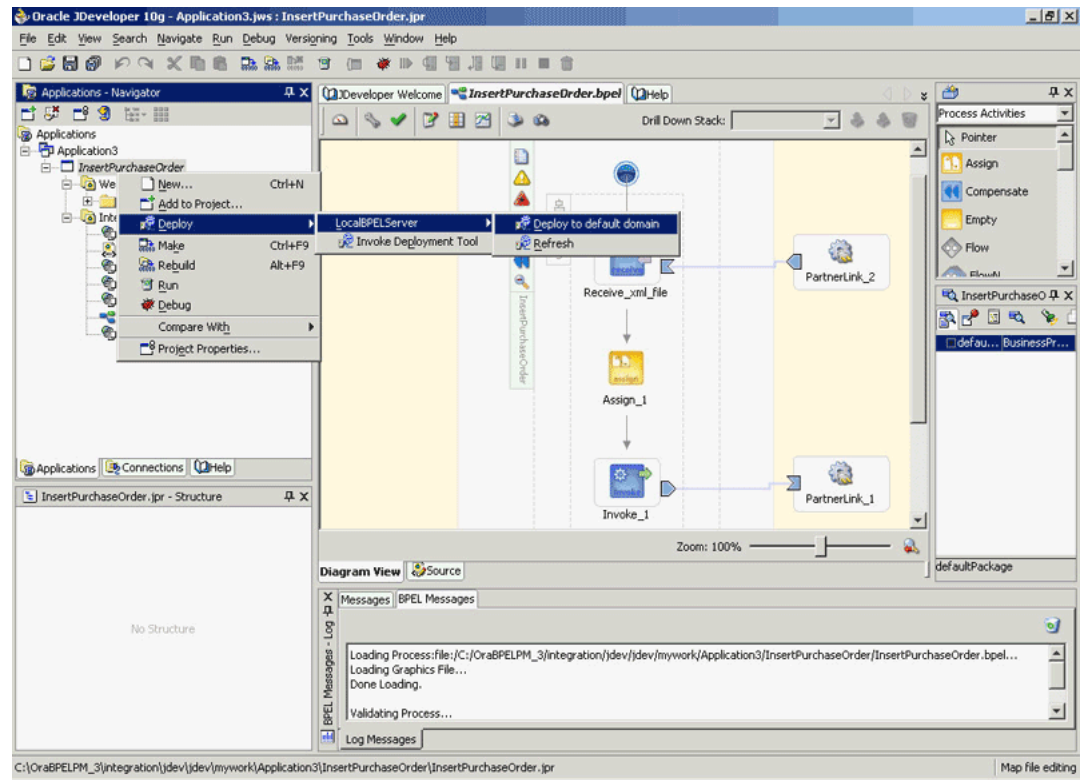
After designing the BPEL process, the next step is to deploy, run, and monitor it. This section discusses the following:

- [Deploying the BPEL Process](#)
- [Testing the BPEL Process](#)

2.1.3.1 Deploying the BPEL Process

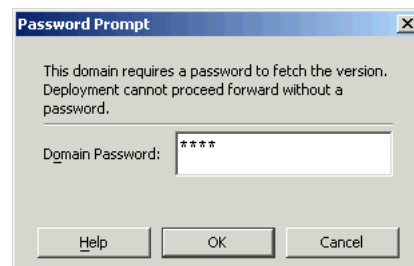
You need to deploy the BPEL process before you can run it. The BPEL process is first compiled, and then deployed to the BPEL server. The following steps discuss deploying the BPEL process to a BPEL server:

1. Select the BPEL project in the Applications window.
2. Right-click the project name. Select **Deploy** from the menu that appears.
3. Select **Local BPEL Server** followed by **Deploy to Default Domain**, if you are deploying the process on the local BPEL server. [Figure 2–19](#) illustrates deploying a BPEL process to a local BPEL server.

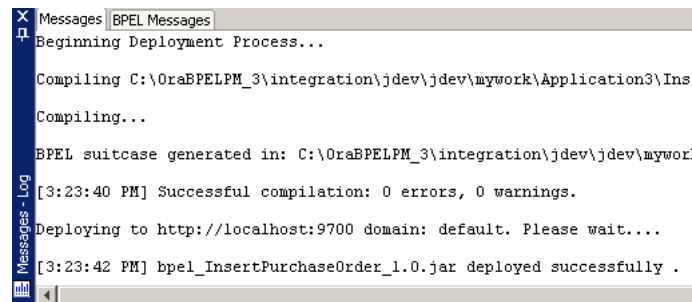
Figure 2–19 Deploying the BPEL Process

Note: You can select **Invoke Deployment Tool** if you want to deploy to a different BPEL server.

4. The Password Prompt dialog box appears. Enter the password for the default domain in the **Domain Password** field. Click **OK**. [Figure 2–20](#) shows the Password Prompt dialog box.

Figure 2–20 Specifying the Domain Password

5. The BPEL process is compiled and deployed. You can check the progress in the Messages window. [Figure 2–21](#) shows the Messages window.

Figure 2–21 Messages Window

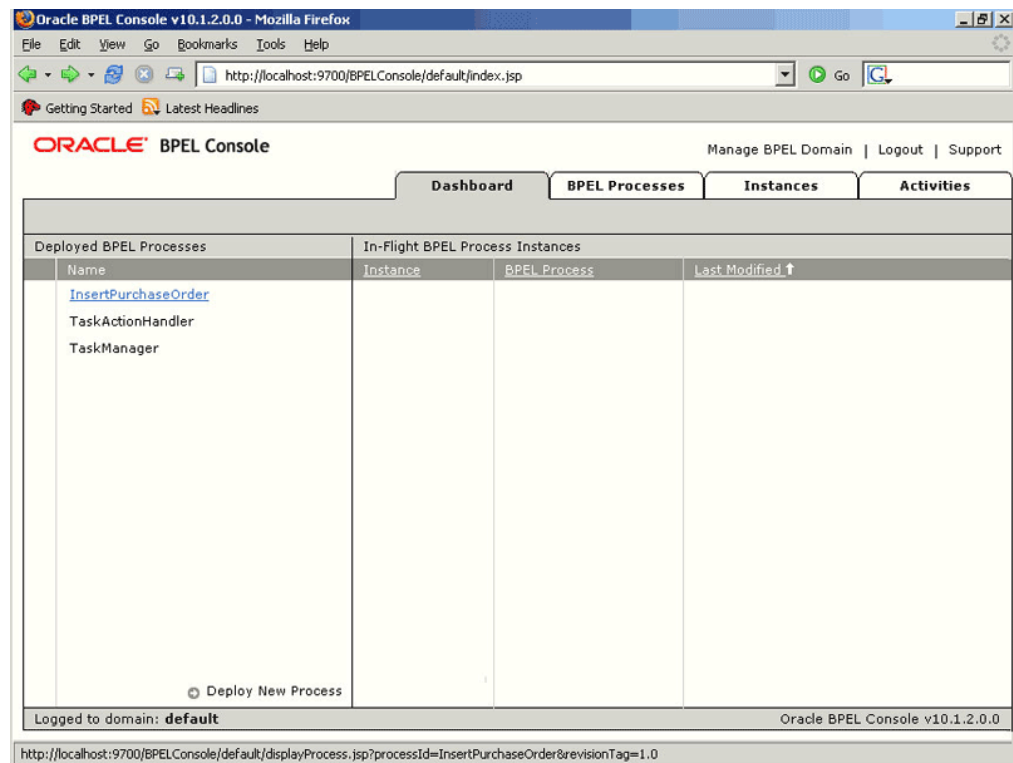
2.1.3.2 Testing the BPEL Process

Once the BPEL process is deployed, it can be seen in the BPEL console. You can manage and monitor the process from the BPEL console. You can also test the process and the integration interface by manually initiating the process. The following steps discuss manually initiating and monitoring the BPEL process:

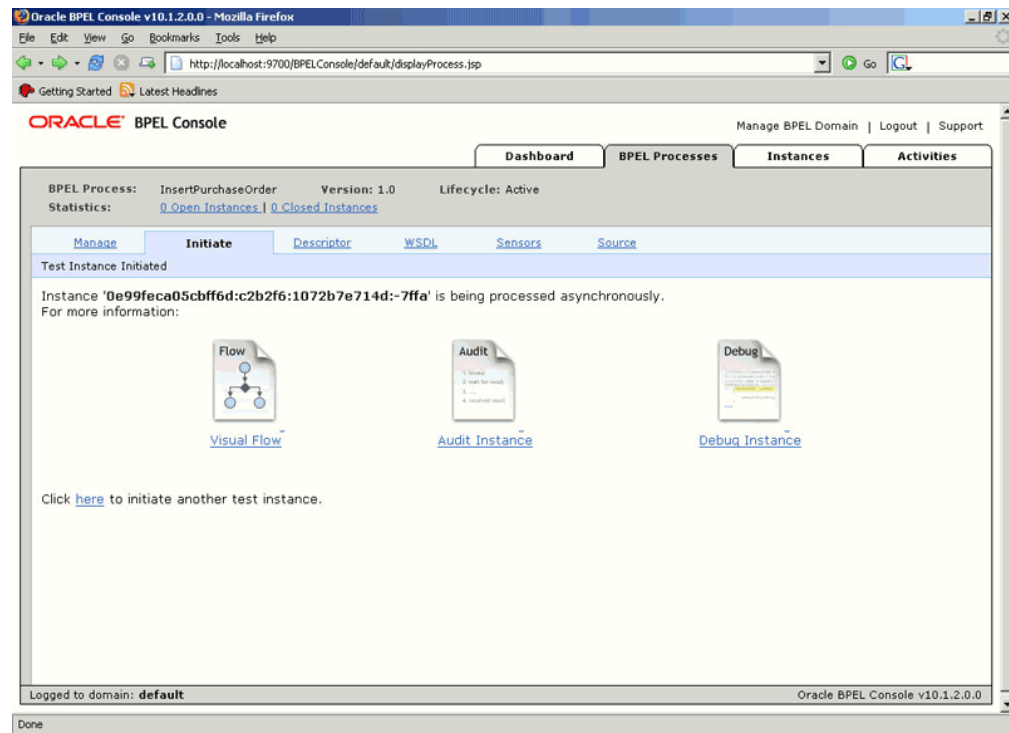
1. To open the BPEL Console, click **Start**, and then select **Programs**. In the Programs menu, select **Oracle - ORACLE_HOME, Oracle BPEL Process Manager 10.1.2**, and then select **BPEL Console**.
2. The BPEL console login screen is displayed. Select **Default** in the **Domain** box. Enter the password for the default domain in the **Password** field, and then click **Login**. Figure 2–22 shows the BPEL console login screen.

Figure 2–22 BPEL Console Login Screen

3. Oracle BPEL console is displayed. The list of deployed processes is shown under Deployed BPEL Processes. Figure 2–23 shows the BPEL console screen.

Figure 2–23 Deployed BPEL Processes

4. Click the BPEL process that you want to initiate. The Initiate page is displayed. Enter the input values required by the process.
5. Click **Post XML Message** to initiate the process.
6. The BPEL process is now initiated. You can check the process flow by clicking the **Visual Flow** icon. [Figure 2–24](#) shows the BPEL Console Initiate page.

Figure 2–24 BPEL Console Initiate Page

7. The audit trail provides information about the steps that have been executed. You can check the audit trail by clicking the **Audit Instance** icon.

Note: To confirm that the records have been written into the open interface tables, you can write the SQL `SELECT` statements and fetch the results showing the latest records inserted into the open interface tables. Alternatively, open the forms of the module for which the record has been inserted, and then check for changes in the form.

2.2 Views

OraclesAS Adapter for Oracle Applications uses views to retrieve data from Oracle Applications. For example, by using views, you can retrieve information about your customers from the required tables in Oracle Applications.

This section contains the following topics:

- [Overview of Views](#)
- [Design-Time Steps for Views](#)
- [Run-Time Step for Views](#)

2.2.1 Overview of Views

OracleAS Adapter for Oracle Applications uses views to retrieve data from Oracle Applications. Views allow only simple definition. By using views, you can get synchronous data access to Oracle Applications. In OracleAS Adapter for Oracle Applications, views are created on base tables as well as interface tables. These views can be used *only* for select operations.

In the Oracle Applications 11.5.10 release, you cannot work on multiple views. A work around to address this would be to create a view that spans multiple views.

2.2.2 Design-Time Steps for Views

This section contains the design-time concepts for views. It contains the following topics:

- [Prerequisites to Configure Views](#)
- [Configuring Oracle Adapter for Oracle Applications](#)

2.2.2.1 Prerequisites to Configure Views

You need to define a uniqueness criteria, which could be a single key or composite keys.

2.2.2.2 Configuring Oracle Adapter for Oracle Applications

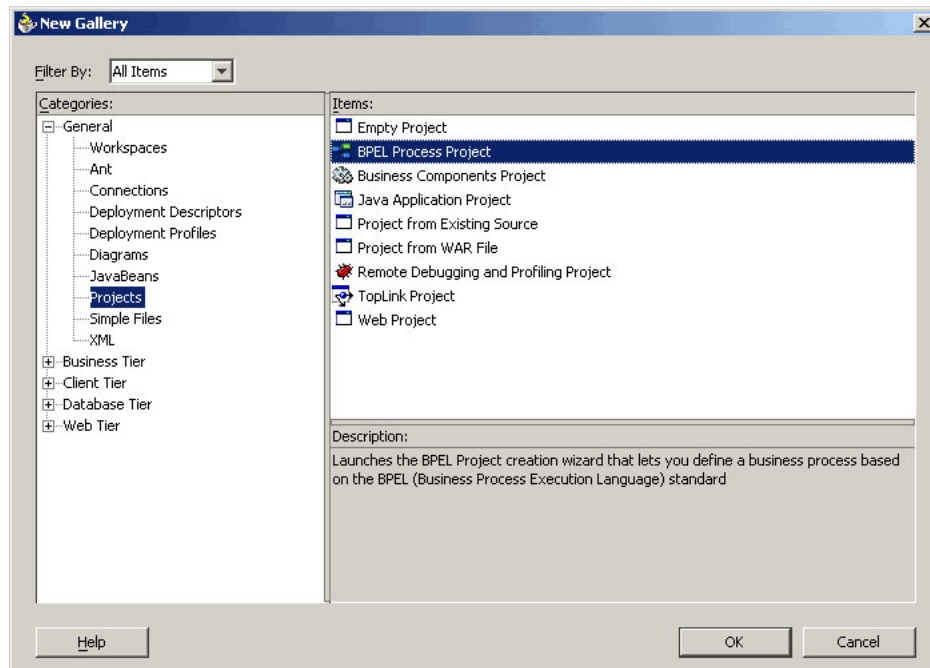
This section describes the steps to configure OracleAS Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper. It includes the following topics:

- [Creating a New BPEL Project](#)
- [Adding a Partner Link](#)
- [Configuring the Invoke Activity](#)
- [Configuring the Assign Activity](#)

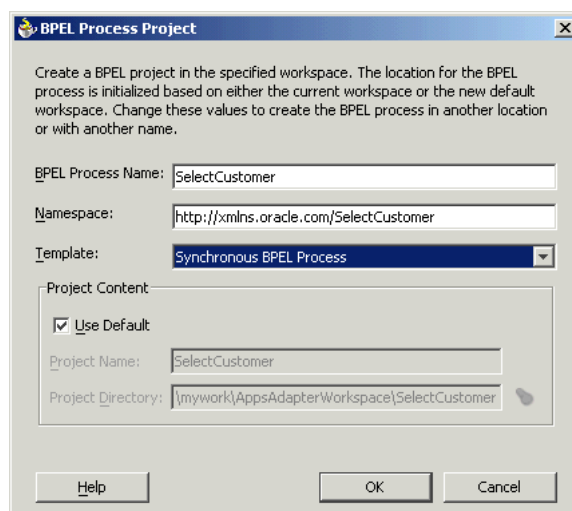
2.2.2.2.1 Creating a New BPEL Project

To create a new BPEL project:

1. Open JDeveloper BPEL Designer.
2. From the **File** menu, select **New**. The New Gallery dialog box is displayed.
3. Select **All Items** from the **Filter By** list. This displays a list of available categories.
4. Expand the **General** node, and then select **Projects**.
5. Select **BPEL Process Project** from the **Items** list, as shown in [Figure 2-25](#).

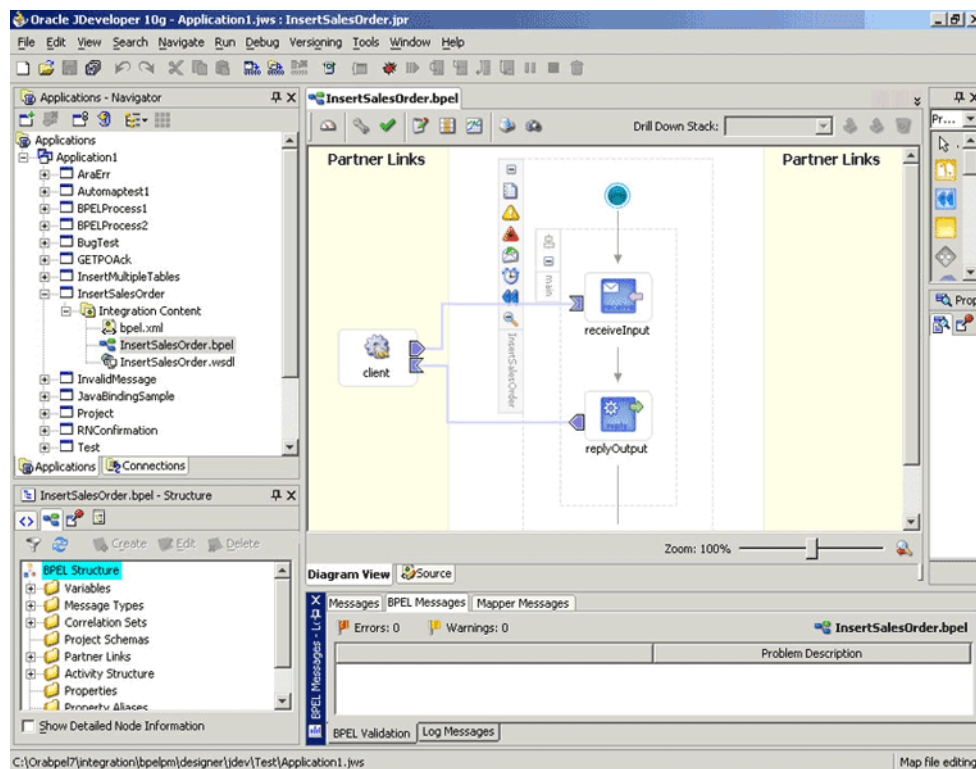
Figure 2–25 Creating a New BPEL Process Project

6. Click **OK**. The BPEL Process Project dialog box is displayed.
7. In the **BPEL Process Name** field, enter a descriptive name, for example, `SelectCustomer`.
8. From the **Template** list, select **Asynchronous BPEL Process**. Keep the default selection for Use Default in the Project Content section, as shown in [Figure 2–26](#).

Figure 2–26 Specifying a Name for the New BPEL Process Project

9. Click **OK**. A new asynchronous BPEL process with the required source files including `bpel.xml`, `SelectCustomer.xml`, and `SelectCustomer.wsdl` is created, as shown in [Figure 2–27](#).

Figure 2–27 New BPEL Process

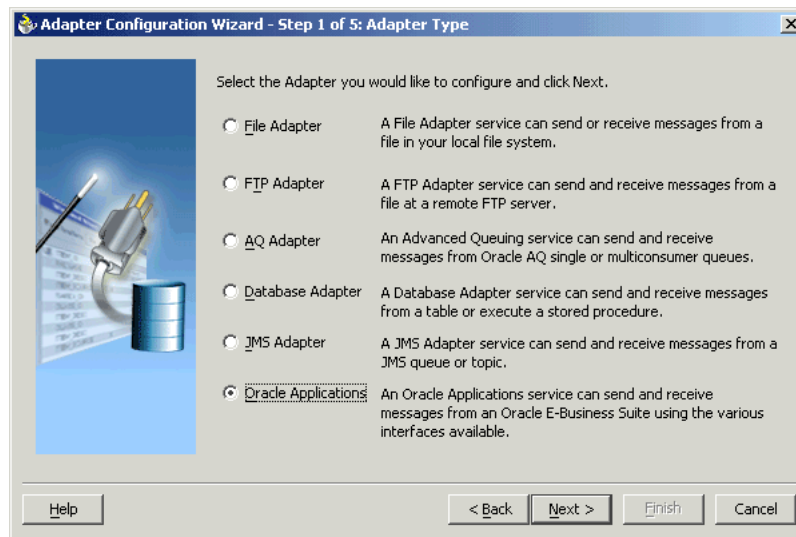


2.2.2.2.2 Adding a Partner Link

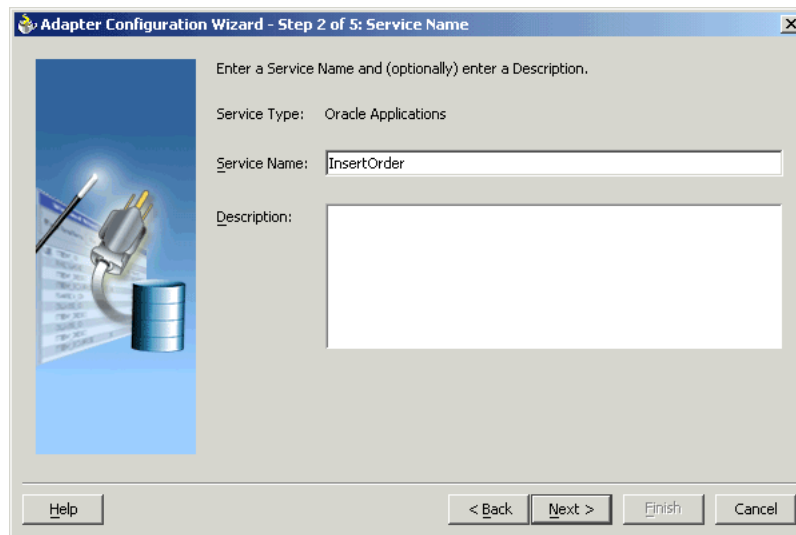
This section describes how to create an OracleAS adapter for the application service by adding a partner link to the BPEL process. A BPEL partner link defines the link name, type, and the role of the BPEL process that interacts with the partner service.

To add a partner link:

1. Drag and drop **PartnerLink** into the border area of the process diagram. The Create Partner Link dialog box is displayed.
2. Click **Define Adapter Service** in the WSDL Settings section. The Adapter Configuration Wizard is displayed.
3. Click **Next**. The Adapter Type dialog box is displayed.
4. Select **Oracle Applications** to specify the adapter you want to configure, as shown in Figure 2–28.

Figure 2–28 Selecting OracleAS Adapter for Oracle Applications

5. Click **Next**. The Service Name dialog box is displayed.
6. Enter the following information:
 1. In the **Service Name** field, enter a service name.
 2. In the **Description** field, enter a description for the service. This is an optional field. The Service Name dialog box is displayed, as shown in [Figure 2–29](#).

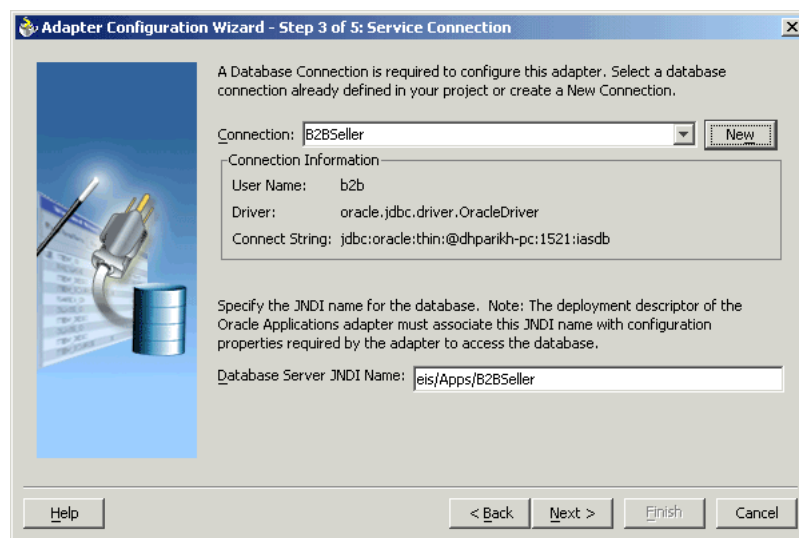
Figure 2–29 Specifying the Service Name

7. Click **Next**. The Service Connection dialog box is displayed.
8. Enter the Java Naming and Directory Interface (JNDI) name in the **Database Server JNDI Name** field. The JNDI name acts as a placeholder for the connection used when your service is deployed to the BPEL server. This enables you to use different databases for development and later production.

See Also: *Oracle Application Server Adapter Concepts* for understanding JNDI concepts

Figure 2–30 shows how to create a new database connection.

Figure 2–30 Creating a New Database Connection

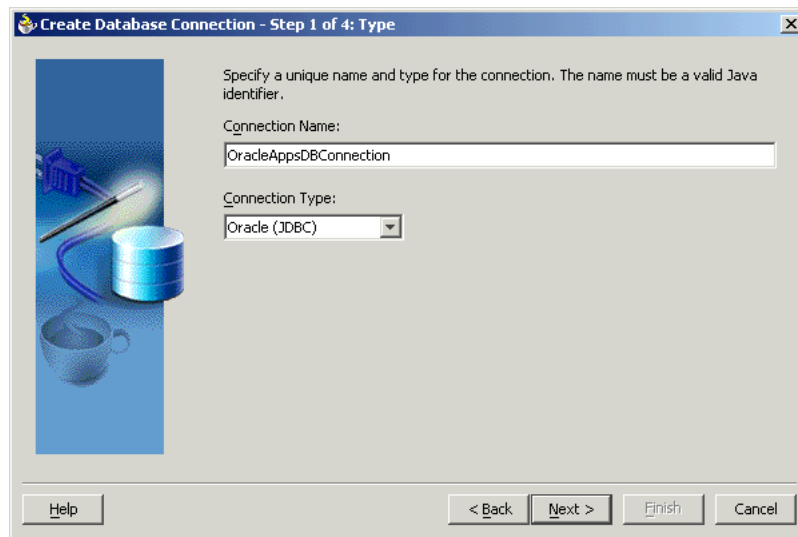


9. Click **New** to define a database connection. The Create Database Connection Wizard is displayed.

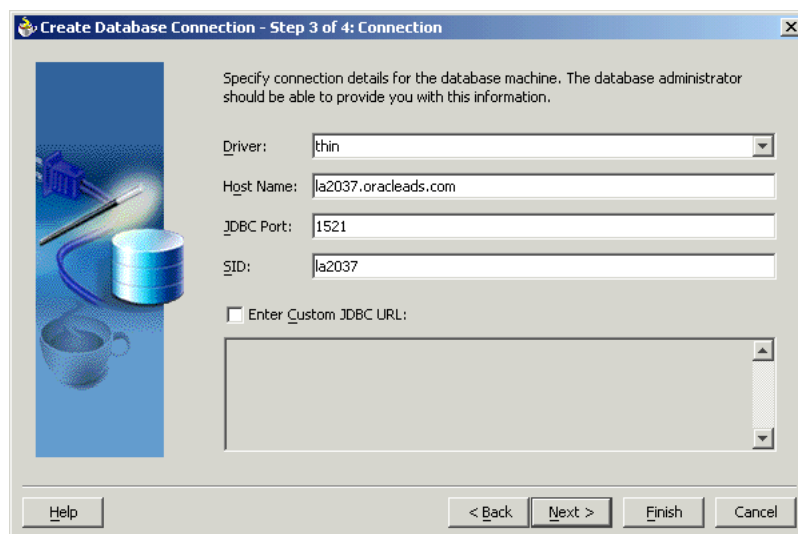
Note: You need to connect to the database where Oracle Applications is running.

10. Enter the following information in the Type dialog box:
 1. In the **Connection Name** field, specify a unique name for the database connection.
 2. From the **Connection Type** list, select the type of connection for your database connection.

The Type dialog box is displayed in Figure 2–31.

Figure 2–31 Specifying the Name and Type of Connection

11. Click **Next**. The Authentication dialog box is displayed.
 12. Enter information in the following fields:
 - a. In the **UserName** field, specify a unique name for the database connection.
 - b. In the **Password** field, specify a password for the database connection.
 13. Click **Next**. The Connection dialog box is displayed.
 14. Enter information in the following fields:
 - a. From the **Driver** list, select **Thin**.
 - b. In the **Host Name** field, specify the host name for the database connection.
 - c. In the **JDBC Port** field, specify the port number for the database connection.
 - d. In the **SID** field, specify a unique SID value for the database connection.
- The Connection dialog box is displayed in [Figure 2–32](#).

Figure 2–32 Specifying the New Database Connection Information

15. Click **Next**. The Test dialog box is displayed.
16. Click **Test Connection** to determine whether the specified information establishes a connection with the database.
17. Click **Next**. The Service Connection dialog box is displayed, providing a summary of the database connection.
18. Click **Finish** to complete the process of creating a new database connection.

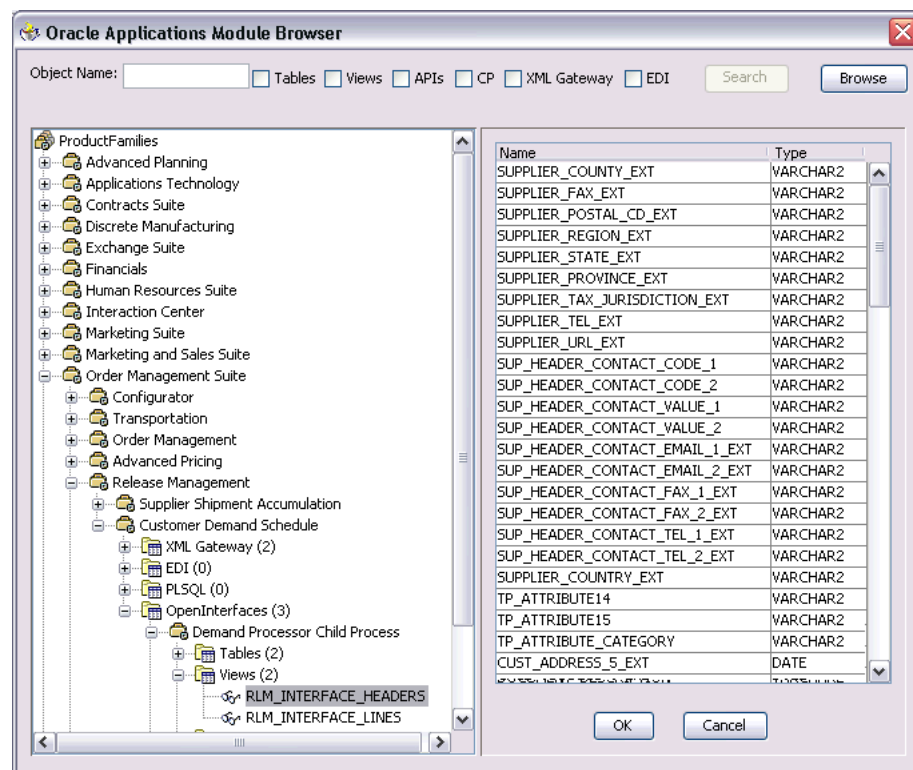
Once you have completed creating a new connection for the service, you can add a view by browsing through the maps available in Oracle Applications.

19. Click **Next** in the Service Connection dialog box. The Operation dialog box is displayed.

Note: If you are connecting to a pre-11.5.10 Oracle Applications instance, then you would be required to select the interface to Oracle Applications data. Select **Tables/Views/APIs/Programs** to proceed.

20. Click **Get Object** to open the Oracle Applications Module Browser. [Figure 2–33](#) shows the Oracle Applications Module Browser.

Figure 2–33 Selecting a View from the Oracle Applications Module Browser



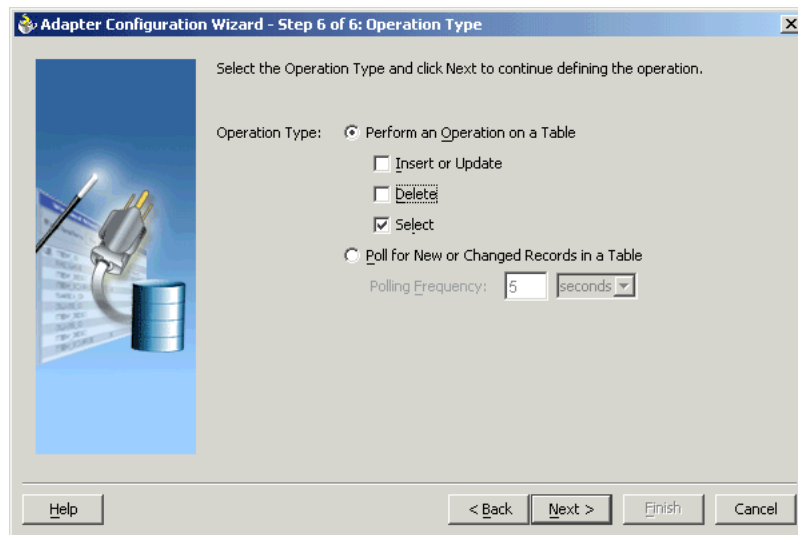
21. Select a view, and then click **OK**.

Note: You can also search for a view by entering the name of the program in the **Object Name** field. Select the **Views** check box, and then click **Search**.

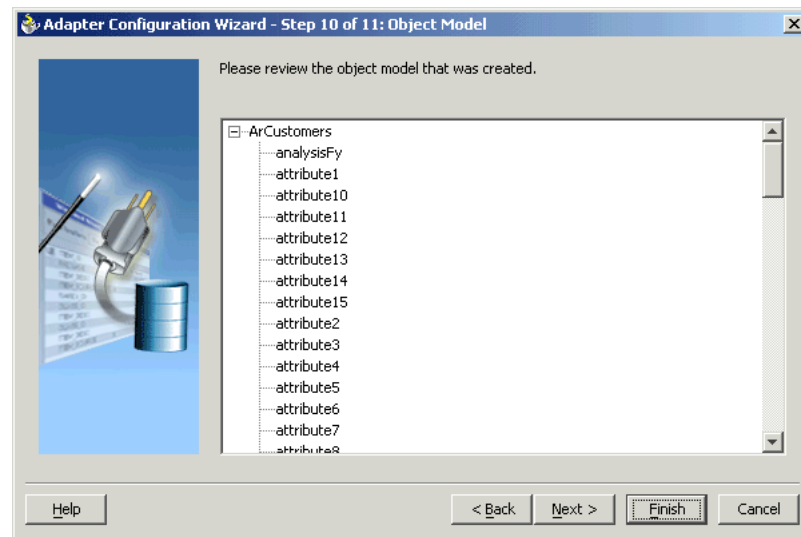
22. The view is added to the Operation Objects section. Repeat steps 2 and 3 to add more tables.
23. Click **Next** in the Operation dialog box. The Operation Type dialog box is displayed.
24. Select **Perform an Operation on a Table**, and then select **Select**, as shown in [Figure 2–34](#).

Note: You can perform only the *Select* operation on views.

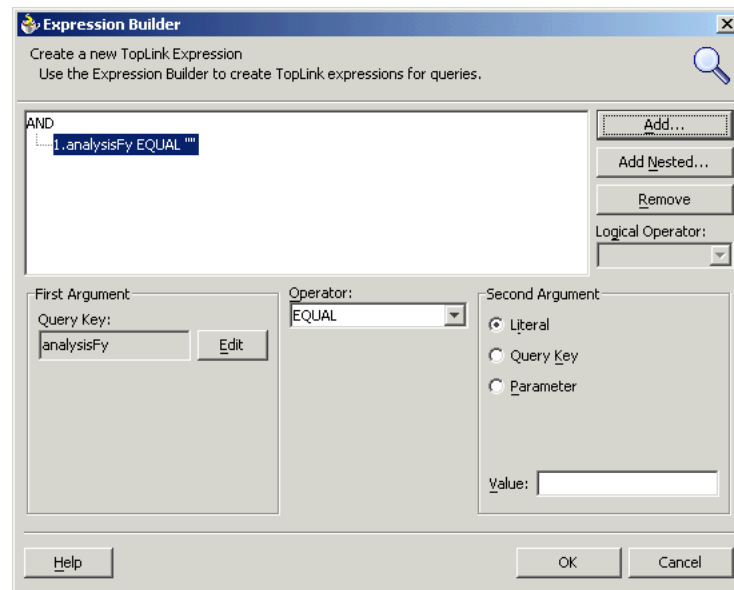
Figure 2–34 *Selecting the Type of Operation*



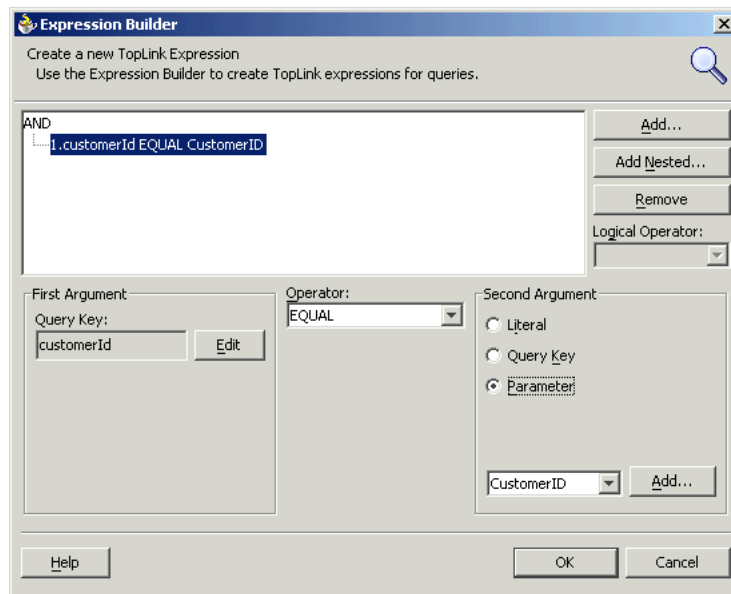
25. Click **Next**. The Select Table screen is displayed.
26. Select the required database table, and then click **Next**. The Define Primary Keys screen is displayed.
27. Select the primary key fields. You can also select multiple fields to define primary keys.
28. Click **Next**. The Relationships screen is displayed.
29. Click **Next**. The Object Model dialog box is displayed, which shows the object models that are created from the imported table definitions, including any relationships that you may have defined. [Figure 2–35](#) shows the Object Model dialog box.

Figure 2–35 Available Object Models

30. Click **Next**. The Define WHERE Clause dialog box is displayed. If your service contains a SELECT query, then you can customize the WHERE clause of the SELECT statement.
31. Click **Add** to add a new parameter. The Parameter Name dialog box is displayed.
32. Enter a name for the new parameter, and then click **OK**.
33. Click **Edit** to create an expression. The **Expression Builder** is displayed.
34. Click **Add** to create a TopLink expression, as shown in [Figure 2–36](#).

Figure 2–36 Creating an Expression

35. Click **Edit**. The Choose screen is displayed.
36. Select the required attribute, and then click **OK**.
37. Select **Parameter** in the Second Argument section, as shown in [Figure 2–37](#).

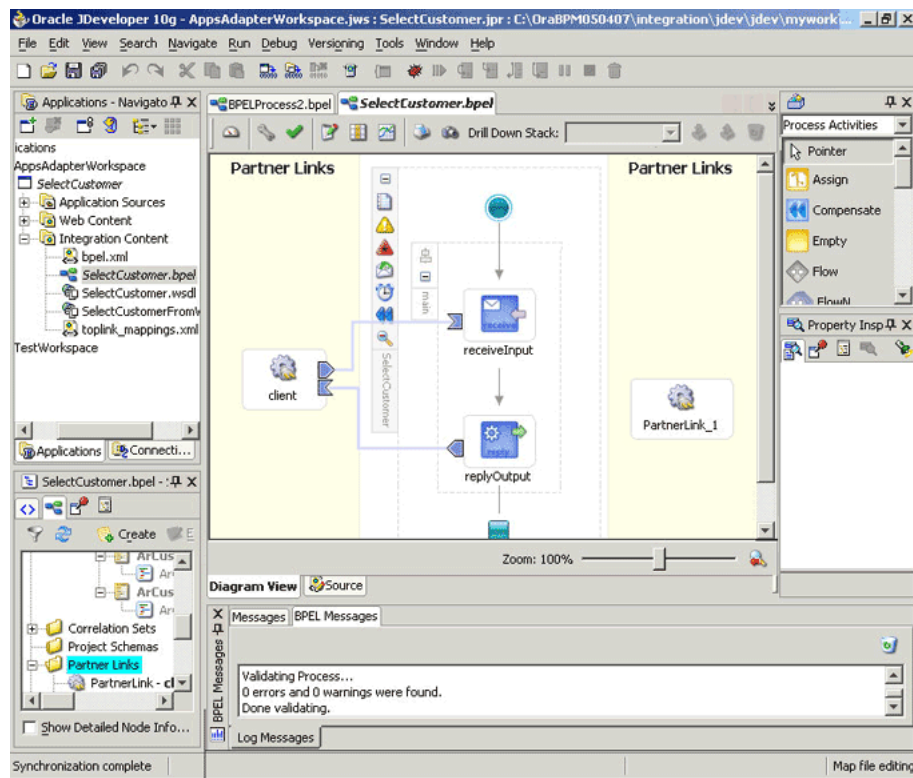
Figure 2–37 Selecting the Required Attributes

See Also: “Understanding descriptors - Building Expressions in the Oracle TopLink documentation for more information about configuring expressions using the Expression Builder, refer to the following information:

- Oracle TopLink page on OTN
<http://www.oracle.com/technology/products/ias/toplink/index.html>
 - Oracle TopLink Documentation
http://download.oracle.com/docs/cd/B14099_04/web.htm#toplink
-

38. Click **OK**. The Define WHERE Clause dialog box is displayed with the selected values.
39. Click **Next**. The Finish screen is displayed.
40. Click **Finish**. The Create Partner Link dialog box is displayed.
41. Click **OK**. The new BPEL process is created, as shown in [Figure 2–38](#).

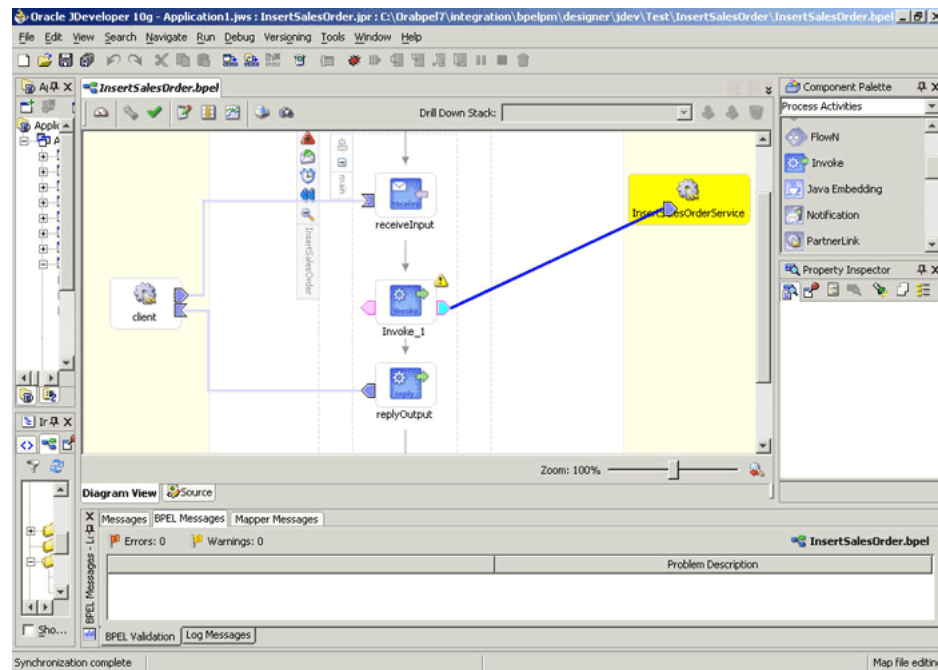
Figure 2–38 New BPEL Process



After adding and configuring the partner link, the next task is to configure the BPEL process.

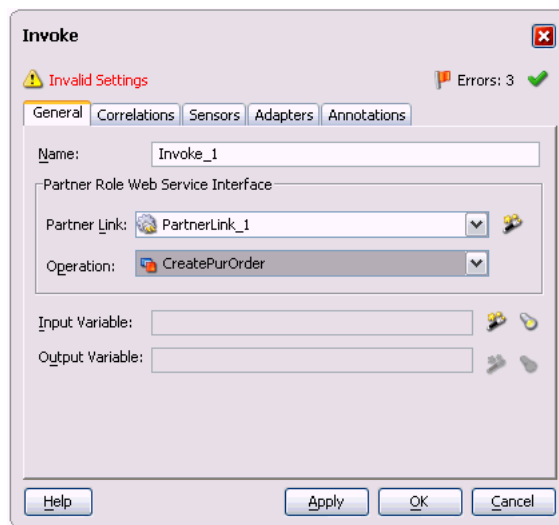
2.2.2.2.3 Configuring the Invoke Activity

1. Drag **Invoke** from the palette and drop it at the location where you want to insert the invoke activity in your BPEL process, as shown in [Figure 2–39](#).

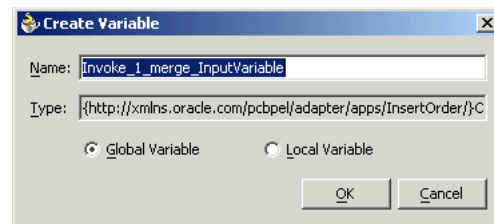
Figure 2–39 Dragging an Invoke Activity

The Edit Invoke dialog box is displayed.

2. Double-click **Invoke** in the process map to open the Invoke dialog box. The General tab is selected by default. [Figure 2–40](#) shows the Invoke dialog box.

Figure 2–40 Configuring the Invoke Activity

3. In the **Partner Link** box, select the partner link to invoke. This is the partner link that you configured in the previous section.
4. Click the **Create** icon next to the Input Variable field. Enter a descriptive name for the variable in the Create Variable dialog box that appears. You can also accept the default name. Click **OK**. [Figure 2–41](#) shows the Create Variable dialog box.

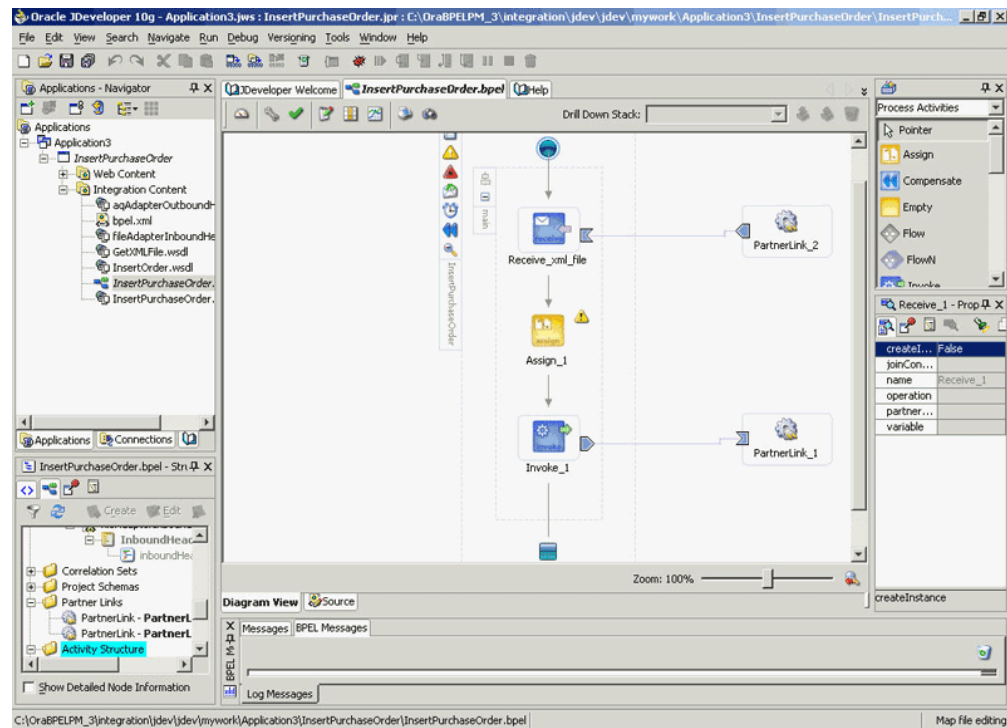
Figure 2–41 Creating a Variable

5. In the Invoke dialog box, click **Apply**, and then click **OK**. The BPEL process is configured.

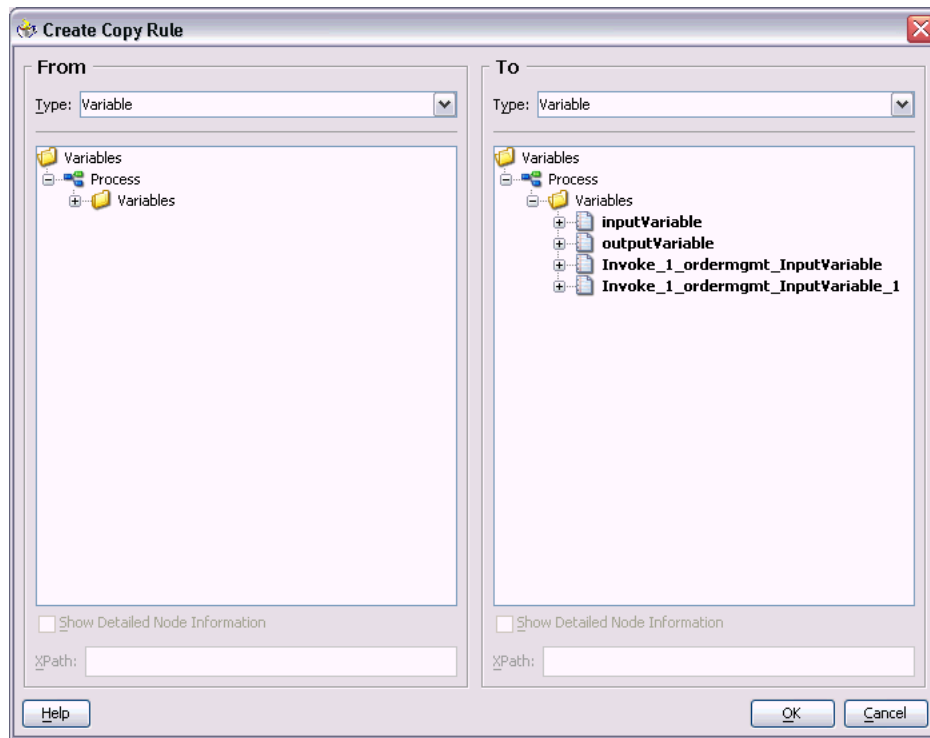
2.2.2.2.4 Configuring the Assign Activity

The next task is to add an Assign activity to the process map. This is used to provide values to the input variables. The following steps discuss this task:

1. Drag and drop the Assign activity to the process map. The Assign activity must be dropped in between the Receive and Invoke activities. [Figure 2–42](#) shows the Assign activity after it has been added to the process map.

Figure 2–42 Adding the Assign Activity

2. You must now configure the Assign activity. Double-click the **Assign** activity in the process map.
3. The Assign dialog box is displayed. The Copy Rules tab is displayed by default. Click **Create**.
4. The Create Copy Rule dialog box is displayed. In the To group, expand the Variables node by clicking the plus sign next to it, and then select **Expression** from the From group to assign values to the input variables. [Figure 2–43](#) shows assigning values to the input variables.

Figure 2–43 Assigning Values to the Input Variables

5. After assigning values to the input variables, click **OK**.
6. Select **Make** from the Run menu or press Ctrl+F9 to compile the BPEL process to check for errors. The compilation result is displayed.

2.2.3 Run-Time Step for Views

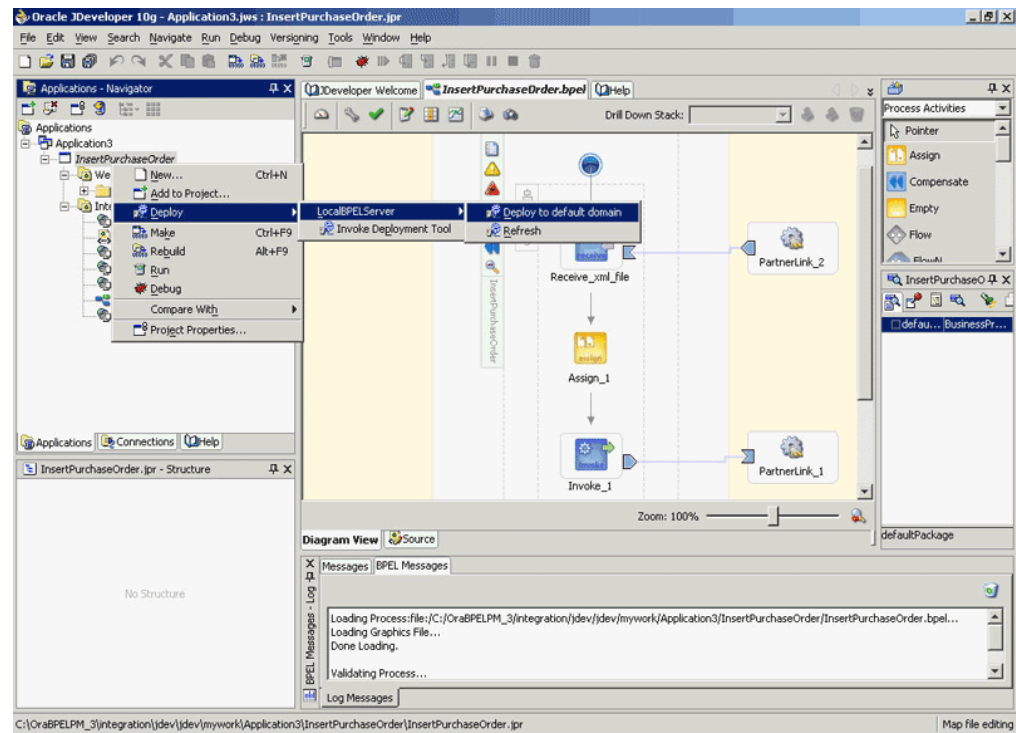
After designing the BPEL process, the next steps are to deploy, run and monitor it. This section discusses the following:

- [Deploying the BPEL Process](#)
- [Testing the BPEL Process](#)

2.2.3.1 Deploying the BPEL Process

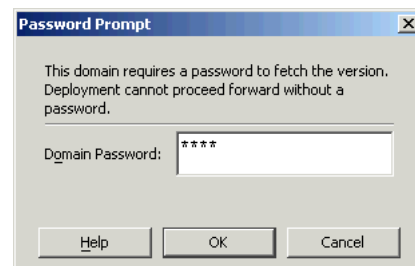
You need to deploy the BPEL process before you can run it. The BPEL process is first compiled, and then deployed to the BPEL server. The following steps discuss deploying the BPEL process to a BPEL server:

1. Select the BPEL project in the Applications window.
2. Right-click the project name. Select **Deploy** from the menu that appears.
3. Select **Local BPEL Server** followed by **Deploy to Default Domain**, if you are deploying the process on the local BPEL server. [Figure 2–44](#) illustrates deploying a BPEL process to a local BPEL server.

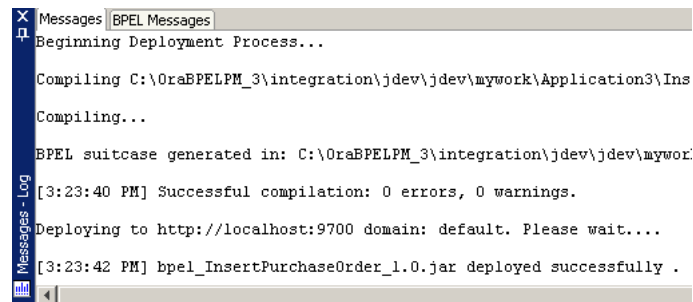
Figure 2–44 Deploying the BPEL Process

Note: You can select **Invoke Deployment Tool** if you want to deploy to a different BPEL server.

4. The Password Prompt dialog box appears. Enter the password for the default domain in the **Domain Password** field. Click **OK**. [Figure 2–45](#) shows the Password Prompt dialog box.

Figure 2–45 Specifying the Domain Password

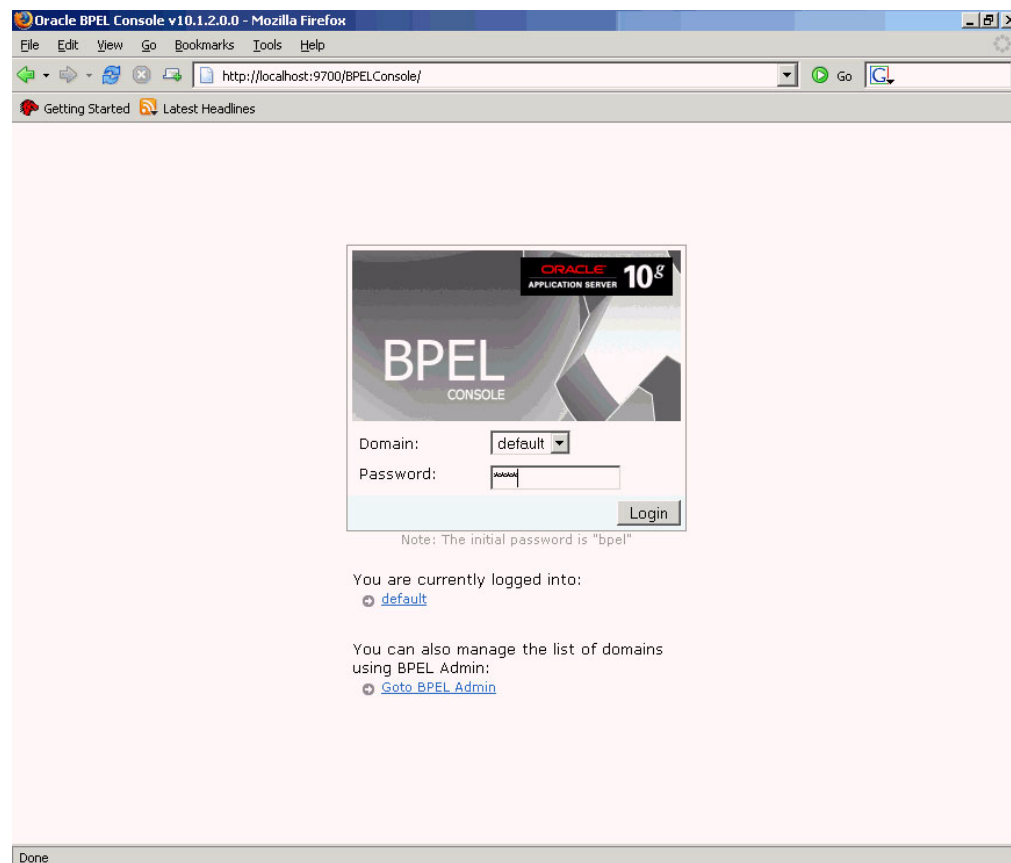
5. The BPEL process is compiled and deployed. You can check the progress of the compilation in the Messages window. [Figure 2–46](#) shows the Messages window.

Figure 2–46 Messages Window

2.2.3.2 Testing the BPEL Process

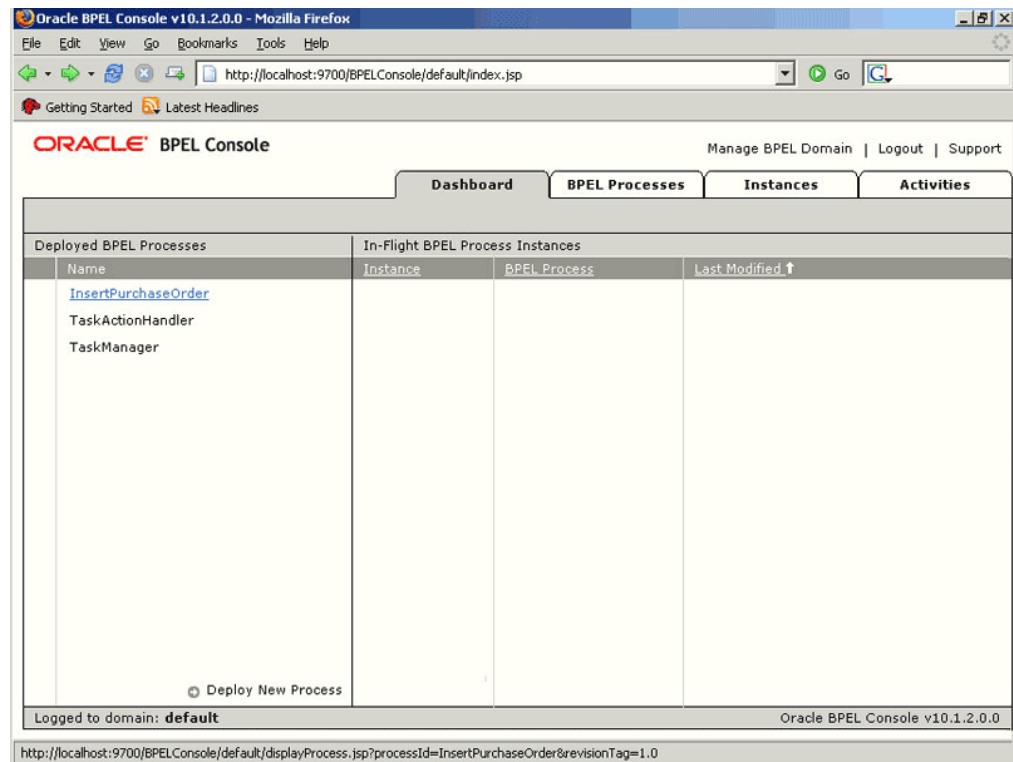
Once the BPEL process is deployed, it can be seen in the BPEL console. You can manage and monitor the process from the BPEL console. You can also test the process and the integration interface by manually initiating the process. The following steps discuss manually initiating and monitoring the BPEL process:

1. To open the BPEL Console, click **Start**, and then select **Programs**. In the Programs menu, select **Oracle - ORACLE_HOME, Oracle BPEL Process Manager 10.1.2**, and then select **BPEL Console**.
2. The BPEL Console login screen is displayed. Select **Default** in the **Domain** box. Enter the password for the default domain in the **Password** field, and then click **Login**. Figure 2–47 shows the BPEL Console login screen.

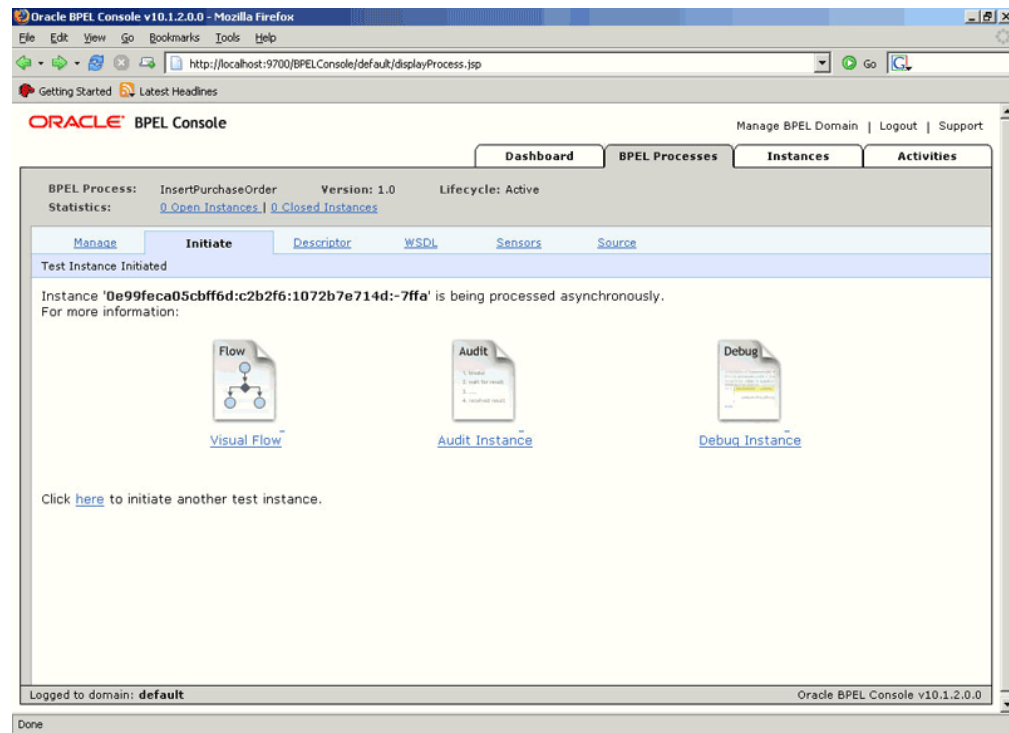
Figure 2–47 BPEL Console Login Screen

- Oracle BPEL console is displayed. The list of deployed processes is shown under Deployed BPEL Processes. Figure 2–48 shows the BPEL Console screen.

Figure 2–48 Deployed BPEL Processes



- Click the BPEL process that you want to initiate. The Initiate page is displayed. Enter the input values required by the process.
- Click **Post XML Message** to initiate the process.
- The BPEL process is now initiated. You can check the process flow by clicking the **Visual Flow** icon. Figure 2–49 shows the BPEL Console Initiate page.

Figure 2–49 BPEL Console Initiate Page

7. The audit trail provides information about the steps that have been executed. You can check the audit trail by clicking the **Audit Instance** icon.

OracleAS Adapter for Oracle Applications use Application programming interfaces (APIs) to insert and update data into Oracle Applications.

This chapter contains the following sections:

- [Overview of APIs](#)
- [Design-Time Steps](#)
- [Run-Time Steps](#)

3.1 Overview of APIs

OracleAS Adapter for Oracle Applications use APIs to insert and update data in Oracle Applications. APIs are stored procedures that enable you to insert and update data in Oracle Applications. For example, by using APIs, you can insert a customer record in Oracle Applications.

3.2 Design-Time Steps

This section describes how to configure the OracleAS Adapter for Oracle Applications to use APIs. It includes the following topics:

- [Prerequisites to Configure APIs](#)
- [Configuring OracleAS Adapter for Oracle Applications](#)

3.2.1 Prerequisites to Configure APIs

OracleAS adapter for Oracle Applications is deployed using the BPEL Process Manager (PM) in Oracle JDeveloper. The BPEL PM creates the WSDL interfaces for the API.

You need to populate certain variables in the BPEL PM in order to provide context information for Oracle Applications. The context information required for an API transaction includes the *username* and *responsibility* of an Oracle Applications user that has sufficient privileges to run the program. The default value passed for the username is SYSADMIN. The default value passed for responsibility is SYSTEM ADMINISTRATOR.

You can change the default values specified in the generated WSDL for the username and responsibility. This is a static way of changing the context information. These values would apply to all invocations of the deployed business process. However, if you need to provide different context information for different invocations of the business process, then you can dynamically populate the header variable with values

for username and responsibility. The context information can be specified by configuring an Assign activity before the Invoke activity in the BPEL PM.

3.2.2 Configuring OracleAS Adapter for Oracle Applications

This section describes the tasks required to configure OracleAS Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper. It contains the following topics:

- [Creating a New BPEL Project](#)
- [Adding a New Partner Link](#)
- [Describing Wrapper APIs](#)
- [Describing Parameters With DEFAULT Clause](#)
- [Configuring the Invoke Activity](#)
- [Configuring the Transform Activity](#)

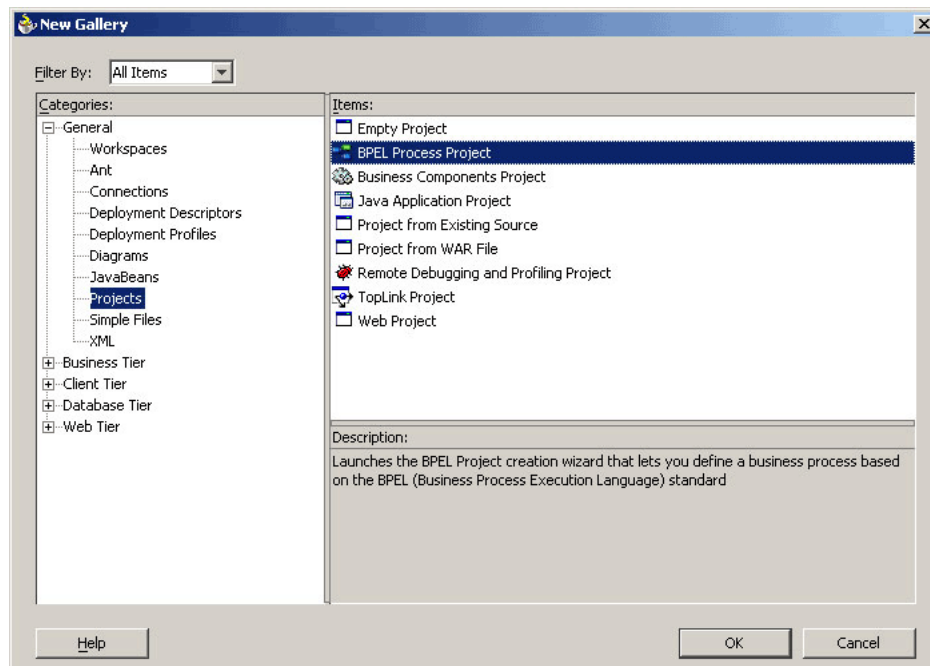
3.2.2.1 Creating a New BPEL Project

The first configuration task is to create a new BPEL project. This section describes how to create a new BPEL project.

To create a new BPEL project:

1. Open BPEL Designer.
2. From the **File** menu, select **New**. The New Gallery dialog box is displayed.
3. Select **All Items** from the **Filter By** list. This displays a list of available categories.
4. Expand the **General** node, and then select **Projects**.
5. Select **BPEL Process Project** from the **Items** list, as shown in [Figure 3–1](#).

Figure 3–1 *Creating a New BPEL Process Project*



6. Click **OK**. The BPEL Process Project dialog box is displayed.
7. In the **BPEL Process Name** field, enter a descriptive name. For example, InsertShipNotice.
8. From the **Template** list, select **Empty BPEL Process**. Keep the default selection for **Use Default** in the Project Content section, as shown in [Figure 3-2](#).

Figure 3-2 Specifying a Name for the New BPEL Process Project

BPEL Process Project

Create a BPEL project in the specified workspace. The location for the BPEL process is initialized based on either the current workspace or the new default workspace. Change these values to create the BPEL process in another location or with another name.

BPEL Process Name:

Namespace:


Template:

Project Content

☒ **Use Default**

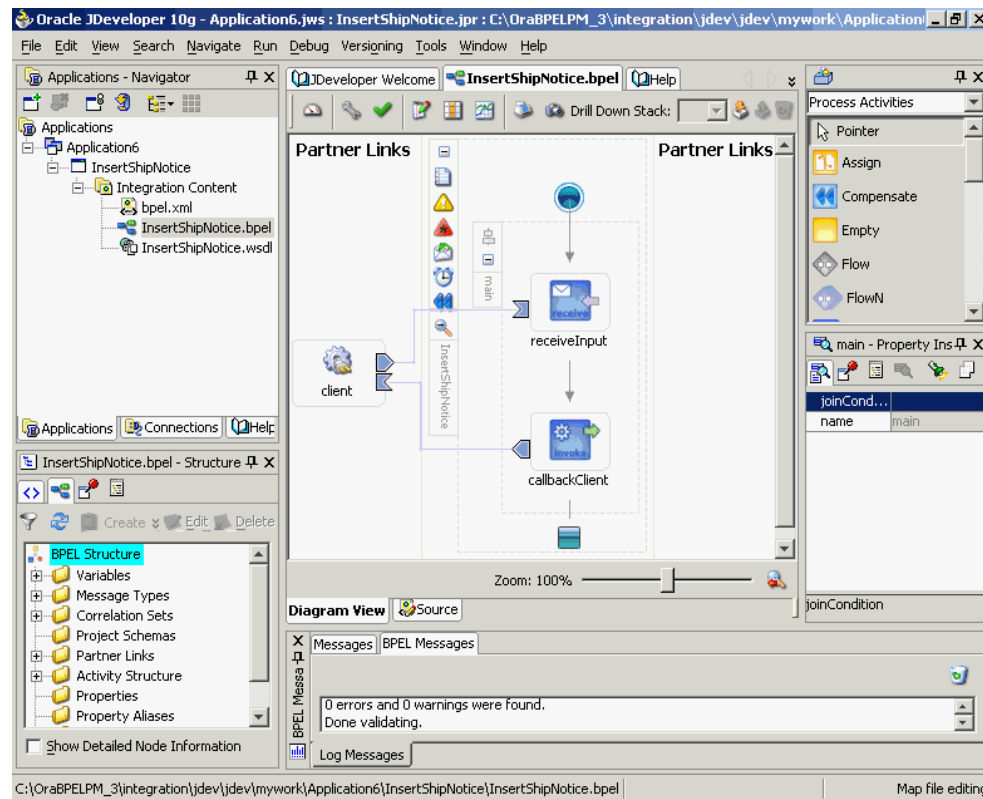
Workspace Name:

Project Name:

Workspace Directory: 

Help **OK** **Cancel**

- Click **OK**. An empty BPEL process, with the necessary source files including `bpel.xml`, `InsertShipNotice.xml`, and `InsertShipNotice.wsdl` is created. [Figure 3-3](#) shows the new BPEL process.

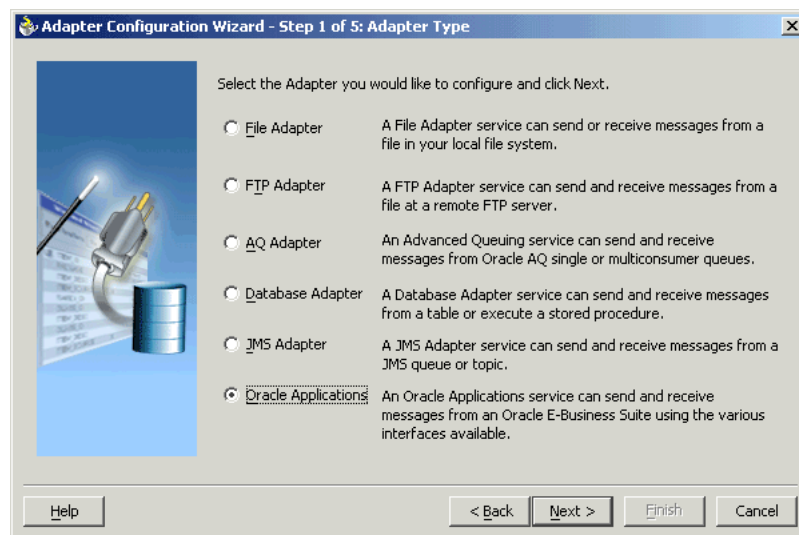
Figure 3–3 New BPEL Process

3.2.2.2 Adding a New Partner Link

The next task is to add a partner link to the BPEL process. This section describes how to create an OracleAS adapter for the application service by adding a partner link to your BPEL process. A BPEL partner link defines the link name, type, and the role of the BPEL process that interacts with the partner service.

To add a partner link:

1. Drag and drop **PartnerLink** into the border area of the process diagram. The Create Partner Link dialog box is displayed.
2. Click **Define Adapter Service** in the WSDL Settings section. The Adapter Configuration Wizard is displayed.
3. Click **Next**. The Adapter Type dialog box is displayed.
4. Select **Oracle Applications** to specify the adapter you want to configure, as shown in Figure 3–4.

Figure 3–4 Selecting OracleAS Adapter for Oracle Applications

5. Click **Next**. The Service Name dialog box is displayed.
6. Enter the following information:
 - a. In the **Service Name** field, enter a service name.
 - b. In the **Description** field, enter a description for the service. This is an optional field. The Service Name dialog box is displayed, as shown in [Figure 3–5](#).

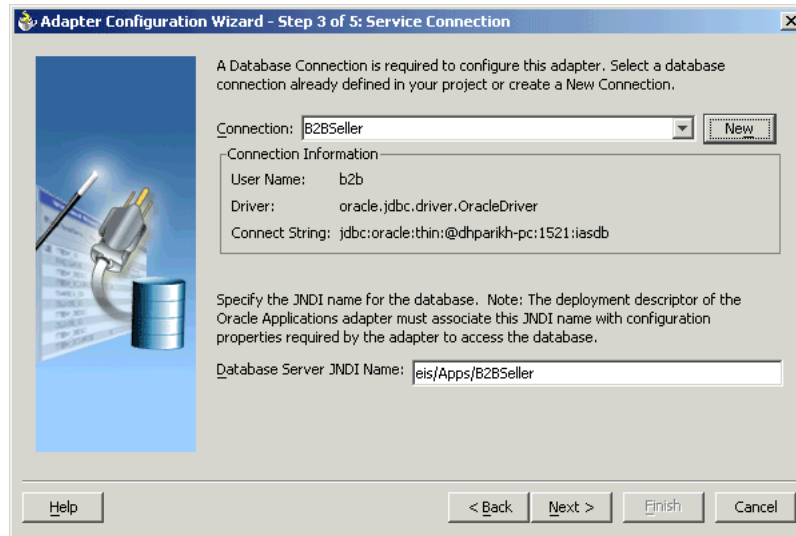
Figure 3–5 Specifying the Service Name and Description

7. Click **Next**. The Service Connection dialog box is displayed.
8. Enter the JNDI (Java Naming and Directory Interface) name in the **Database Server JNDI Name** field. The JNDI name acts as a placeholder for the connection used when your service is deployed to the BPEL server. This enables you to use different databases for development and later for production.

See Also: *Oracle Application Server Adapter Concepts* for understanding JNDI concepts.

Figure 3–6 shows how to create a new database connection.

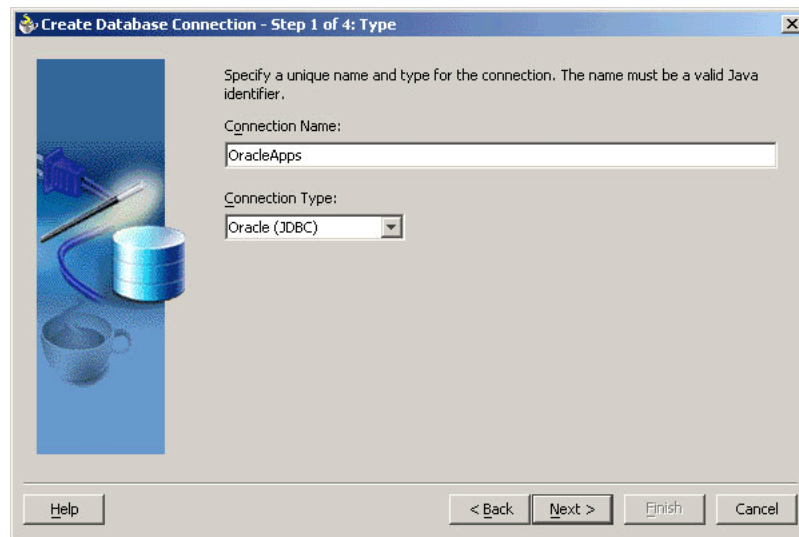
Figure 3–6 Creating a New Database Connection



9. Click **New** to define a database connection. The Create Database Connection Wizard is displayed.

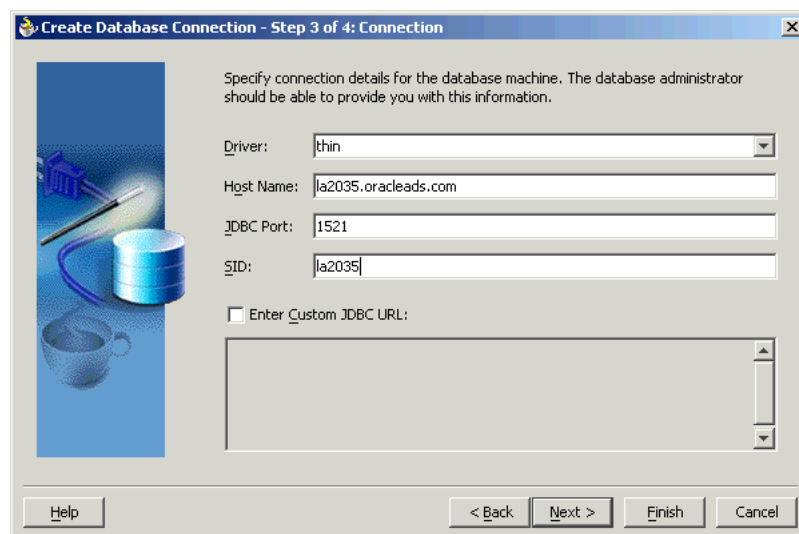
Note: You need to connect to the database where Oracle Applications is running.

10. Click **Next**. The Type dialog box is displayed, as shown in Figure 3–7.
11. Enter the following information in the Type dialog box:
 - a. In the **Connection Name** field, specify a unique name for the database connection.
 - b. From the **Connection Type** list, select the type of connection for your database.

Figure 3–7 Specifying a Name and Type for the Database Connection

12. Click **Next**. The Authentication dialog box is displayed.
13. Enter the following information:
 - a. In the **UserName** field, specify a unique name for the database connection.
 - b. In the **Password** field, specify a password for the database connection.
14. Click **Next**. The Connection dialog box is displayed.
15. Enter the following information:
 - a. From the **Driver** list, select **Thin**.
 - b. In the **Host Name** field, specify the host name for the database connection.
 - c. In the **JDBC Port** field, specify the port number for the database connection.
 - d. In the **SID** field, specify the unique SID value for the database connection.

The Connection dialog box is displayed, as shown in [Figure 3–8](#).

Figure 3–8 Specifying Connection Details for the New Database

16. Click **Next**. The Test dialog box is displayed.
17. Click **Test Connection** to determine whether the specified information establishes a connection with the database.
18. Click **Next**. The Service Connection dialog box is displayed, providing a summary of your database connection.
19. Click **Finish** to complete the process of creating a new database connection.

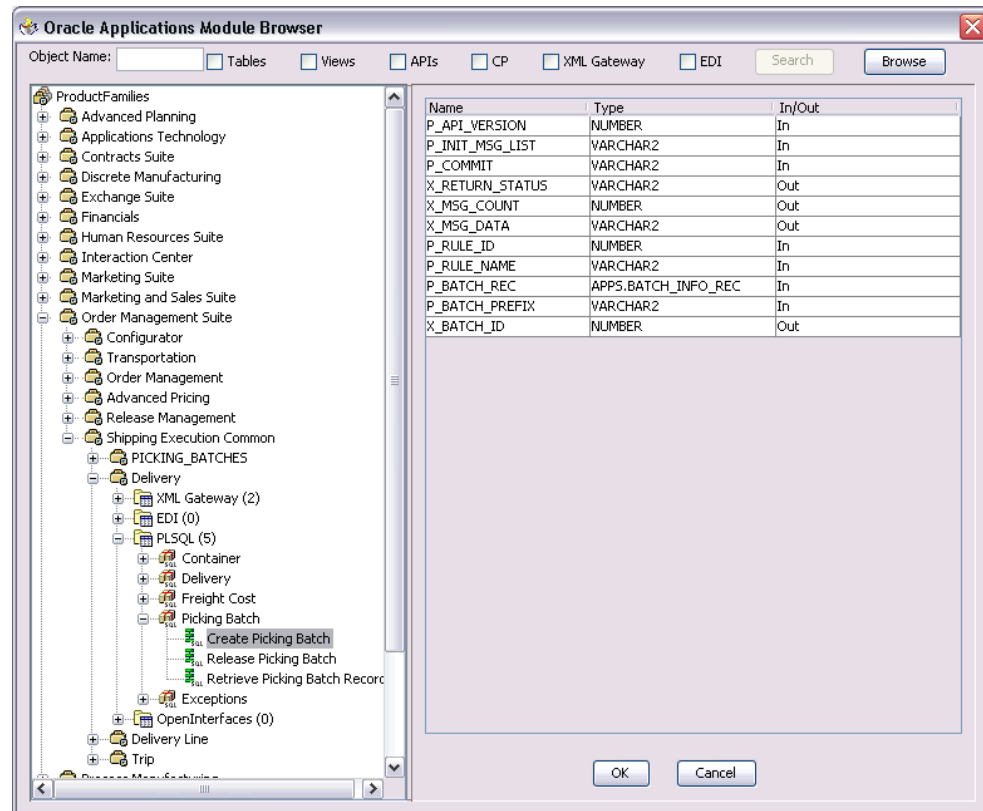
Once you have completed creating a new connection for the service, you can add an API by browsing through the list of APIs available in Oracle Applications.
20. Click **Next**. The Application Interface dialog box is displayed, as shown in [Figure 3–9](#).

Note: If you are connecting to a pre-11.5.10 Oracle Applications instance, then you would be required to select the interface to Oracle Applications data. Select **Tables/Views/APIs/Concurrent Programs** to proceed.

Figure 3–9 Adding the Database Objects



21. Click **Get Object** to open the Oracle Applications Module Browser. [Figure 3–10](#) shows the Oracle Applications Module Browser.

Figure 3–10 Specifying the API

Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. For example, the Marketing Suite or the Order Management Suite are product families in Oracle Applications. The product families contain the individual products. For example, the Order Management Suite contains the Shipping Execution Common product. The product contains the business entities associated with the product. For example, the Shipping Execution Common product contains the Delivery entity.

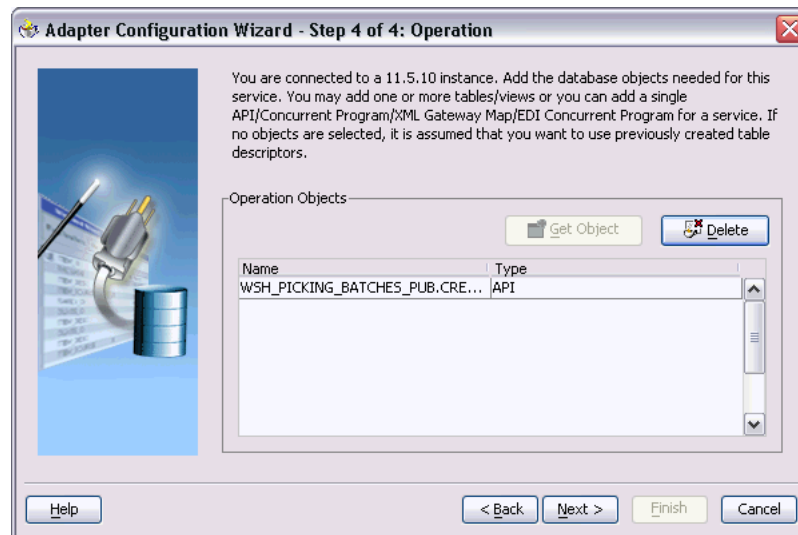
Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide. APIs can be found under the PLSQL category.

Note: In the 11.5.10 instance the Module Browser displays only the public APIs that are exposed to the integration repository. Whereas, in the pre-11.5.10 instance the Module Browser displays all the APIs that are available in the package.

22. Select the required API. The signature of that given API is displayed.
23. click **OK**. You can select only one API for each adapter service. [Figure 3–11](#) shows that the API has been added.

Note: Use the Search option to quickly find the required objects. Enter the required database object name in the **Object Name** field and select **APIs**. And then, click **Search** to retrieve the required database objects. When searching for an API, enter the package name, and *not* the procedure name. In contrast, when using the DB Adapter Wizard, the user *must* enter the name of the API while searching.

Figure 3–11 Adding the API



24. Click **Next**, and then click **Finish** to complete the process of configuring OracleAS Adapter for Oracle Applications. The wizard generates the WSDL file corresponding to the XML schema. This WSDL file is now available for the partner link.

25. Click **OK**. The partner link is created with the required WSDL settings.

After adding and configuring the partner link, the next task is to configure the BPEL process.

3.2.2.3 Describing Wrapper APIs

The Adapter Configuration wizard generates a wrapper API when an API has arguments the data types of which are: PL/SQL Boolean, PL/SQL Table, or PL/SQL Record. For generating the wrapper API, Oracle JPublisher is automatically invoked in the background. When a wrapper API is created, besides the WSDL and XSD files, two SQL files are created: one for creating the wrapper API, and necessary datatypes and another for deleting it. The two SQL files are saved in the same directory where the WSDL and XSD files are stored, and are available in the Project view.

The following is a sample code of an API that would require a wrapper to be generated:

```
package pkg is
  type rec is record (...);
  type tbl is table of .. index by ..;
  procedure proc(r rec, t tbl, b boolean);
end;
```


If the preceding API is selected in the wizard, then a wrapper API will be created, and loaded into the database. This wrapper API will be used instead of the originally selected API. For this reason, the content of the WSDL and XSD files represent the wrapper procedure, not the procedure originally selected.

The following are the types that will be created for the wrapper API:

- Object type for PL/SQL RECORD
- Nested table of the given type for PL/SQL TABLE. For example, the nested table of NUMBER.
- INTEGER substituted for PL/SQL BOOLEAN

The generated SQL file that creates the wrapper API also creates the required schema objects. The types of the wrapper APIs parameters will be that of the new schema object types. The wrapper package will contain conversion APIs to convert between the base PL/SQL type and the new schema object types.

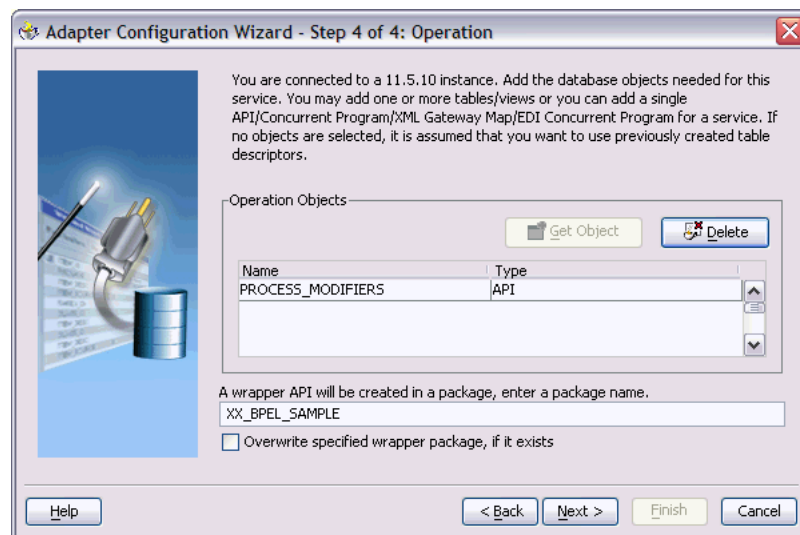
Note: The SQLJUTL package contains the BOOL2INT and INT2BOOL conversion functions used for PL/SQL BOOLEAN arguments whose data types have been changed to INTEGER.

The wrapper API is created in a package. This package is named `XX_BPEL_SERVICE_NAME`, as shown in [Figure 3-12](#). `SERVICE_NAME` is the name of the service that you entered in step 6 of [Adding a New Partner Link](#). If this package already exists, then the wizard prompts for a different package name, or to select a checkbox, to overwrite the existing package. Overwriting an existing package causes all APIs in the specified package to be lost. When the wizard creates a package for the wrapper, only one API, that is, the wrapper API, is contained in it.

Note: Despite specifying to overwrite an existing package, if the wrapper API already exists in the specified package, the wizard will not re-create the wrapper API, as it would take some time. This means no SQL files will be created, Oracle JPublisher will not be run, and the WSDL and XSD files will be for the existing wrapper API. The Finish page of the wizard will indicate that these actions will take place, but it is possible that they will not, depending on whether the wrapper already exists.

The name of the wrapper API depends on whether the API that was originally selected is in a package or not. If the original API is a root-level API, that is, it does not belong in a package, then the name of the wrapper API will be, `TOPLEVEL$ORIGINAL_API_NAME`. If the originally selected API is in a package, then the name of the wrapper API will be, `ORIGINAL_PACKAGE_NAME$ORIGINAL_API_NAME`.

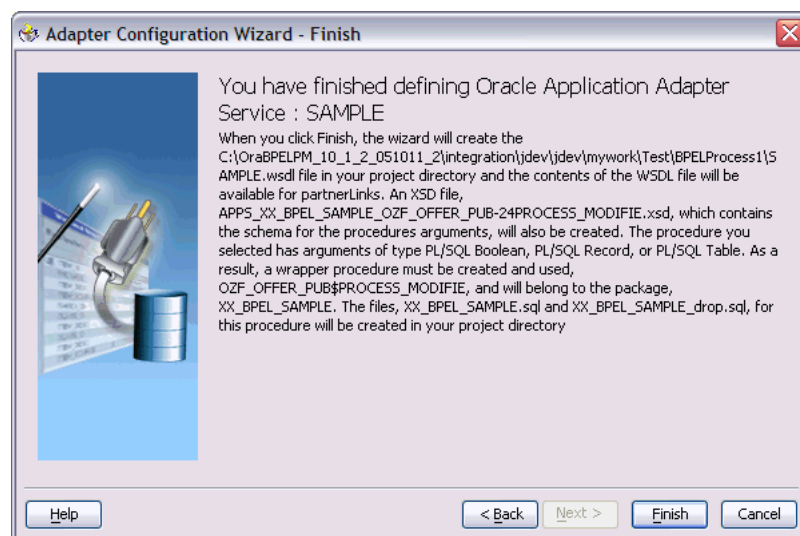
Wrapper APIs follow the naming convention of Oracle JPublisher. For example, if the original API was called `CREATE_EMPLOYEE` and was a root-level API, then the wrapper API would be named, `TOPLEVEL$CREATE_EMPLOYEE`. If the original API is in a package called `EMPLOYEE`, then the wrapper API would be named, `EMPLOYEE$CREATE_EMPLOYEE`.

Figure 3–12 Entering a Package Name for the Wrapper API

Note: The package name for the wrapper has a limit of 30 characters, and the wrapper API name has a limit of 29 characters. Thus, if the package name and the wrapper API names are longer than the maximum limit, then they will be truncated accordingly.

The Finish page is different when a wrapper API needs to be created, as shown in [Figure 3–13](#). The Finish page informs you that a wrapper API is needed, and in addition, lists the name of the wrapper package, wrapper API, and the SQL files that will be created.

Note: When a wrapper API needs to be created, it may take a while before the wizard completes. However, the processing time for subsequent APIs in the same package would be much shorter.

Figure 3–13 The Finish Page

Note: The REF CURSOR type is not supported out of the box. However, for detailed steps to generate an adapter service for an API which takes REF CURSOR type, refer to: http://otndnld.oracle.co.jp/document/products/as10g/1012/doc_v3/integrate.1012/B14448-01/html/adptr_db.htm. Refer to Support for REF CURSOR under Advance Topics.

Overloaded APIs are not supported in the 11.5.10 instance.

3.2.2.4 Describing Parameters With DEFAULT Clause

You can declare parameters of a stored procedure with a default clause, that when, in the absence of the parameter from the invocation of the procedure, will supply a default value for that parameter. For example:

```
PROCEDURE addEmployee (name VARCHAR2, country VARCHAR2 DEFAULT 'US')
```

This procedure can be invoked in the following two different ways:

1. addEmployee ('John Smith')// country => 'US'
2. addEmployee ('John Smith', 'France')// country => 'France'

You can omit elements for parameters with default values in the instance XML. The procedure will be invoked without these parameters, allowing their default values to be used, as shown in the following example:

Input:

```
<db:InputParameters xmlns:db="...">
  <name>John Smith</name>
</db:InputParameters>
```

Runtime Invocation:

```
BEGIN addEmployee (name=>?); END; // country => 'US'
or
```

Input:

```
<db:InputParameters xmlns:db="...">
  <name>John Smith</name>
  <country>France</country>
</db:InputParameters>
```

Runtime Invocation:

```
BEGIN addEmployee (name=>?, country=>?); END; // country => 'France'
```

The element in the XSD for parameters with a default clause is annotated with a special tag to indicate that the parameter has a default clause, as shown in the following example.

```
<element name="country" ... db:default="true" .../>
```

This new functionality allows elements for parameters without a default clause also to be omitted in the instance XML. In these cases, the parameter is still included in the invocation of the stored procedure. A value of NULL is bound by default. The following is an example, where the country parameter did not have a default clause:

```
PROCEDURE addEmployee (name VARCHAR2, country VARCHAR2)
```

In prior release, BPEL PM 10.1.2, elements for both parameters were required in the instance XML. If an element was omitted, then it was presumed to have a default clause so the parameter would not be included in the invocation of the procedure. In this case, the missing parameter would result in a PL/SQL error stating that an incorrect number of arguments were passed to the procedure.

In the current BPEL PM release, the missing parameter will be included in the invocation of the procedure. A NULL value will be bound, as shown in the following example:

```
Input:
    <db:InputParameters xmlns:db="...">
    <name>John Smith</name>
    </db:InputParameters>

Runtime Invocation:
    BEGIN addEmployee (name =>?, country=>?); END; // country => NULL
```

Even though the element for country was not provided in the instance XML, it still appears in the call to the procedure. In this case, country will be NULL.

3.2.2.4.1 Describing DEFAULT Clause Handling in Wrapper Procedures

If a procedure contains a special type requiring a wrapper to be generated, then the default clauses on any of the original parameters will *not* be carried over to the wrapper, as shown in the following example:

```
PROCEDURE needsWrapper(isTrue BOOLEAN, value NUMBER DEFAULT 0)
```

Assuming that the procedure in the preceding example was defined at the top level, outside of a package, then the generated wrapper will appear, as shown in the following example:

```
TOPLEVEL$NEEDSWRAPPER (isTrue INTEGER, value NUMBER)
```

In the preceding example, the BOOLEAN type has been replaced by INTEGER. The default clause on the value parameter is missing. In the current release, parameters of generated wrapper procedures will never have a default clause, even if these parameters did in the original procedure. If the element is missing in the instance XML, instead of defaulting to 0, then the value of the parameter will be NULL, as shown in the following example:

```
Input:
    <db:InputParameters xmlns:db="...">
    <isTrue>1</isTrue>
    </db:InputParameters>

Runtime Invocation:
    BEGIN TOPLEVEL$NEEDSWRAPPER (isTrue =>?, value =>?); END; // value => NULL
```

To fix this, you can edit the generated SQL file, restoring the default clauses. You should then, run the SQL file to reload the wrapper definitions into the database schema. In addition, you should modify the generated XSD.

Following are the steps to fix the default clause with the wrapper generated for `needsWrapper()`:

1. Change the signature in the following manner in the generated wrapper SQL file, *from* `TOPLEVEL$NEEDSWRAPPER (isTrue INTEGER, value NUMBER)` *to* `TOPLEVEL$NEEDSWRAPPER (isTrue INTEGER, value NUMBER DEFAULT 0)`
2. Reload the modified wrapper SQL file mentioned in the preceding example into the appropriate database schema. For BOOLEAN parameters with DEFAULT clause, you need to map as follows:

```
DEFAULT TRUE(base) to DEFAULT 1 (wrapper)
DEFAULT FALSE (base) to DEFAULT 0 (wrapper)
```

For example, if the base stored procedure is PROCEDURE needsWrapper(isTrue BOOLEAN TRUE, value NUMBER DEFAULT 0), then the generated wrapper would be, TOPLEVEL\$NEEDSWRAPPER (isTrue INTEGER, value NUMBER). You should manually fix the store procedure to be, TOPLEVEL\$NEEDSWRAPPER (isTrue INTEGER DEFAULT 1, value NUMBER DEFAULT 0)

3. If a parameter has a default clause, then its corresponding element in the XSD must have an extra attribute, db:default="true". For example, if a parameter has a default clause, TOPLEVEL\$NEEDSWRAPPER(isTrue INTEGER DEFAULT 1, value NUMBER DEFAULT 0), then the elements in the XSD for isTrue and value need to have the following new attributes:

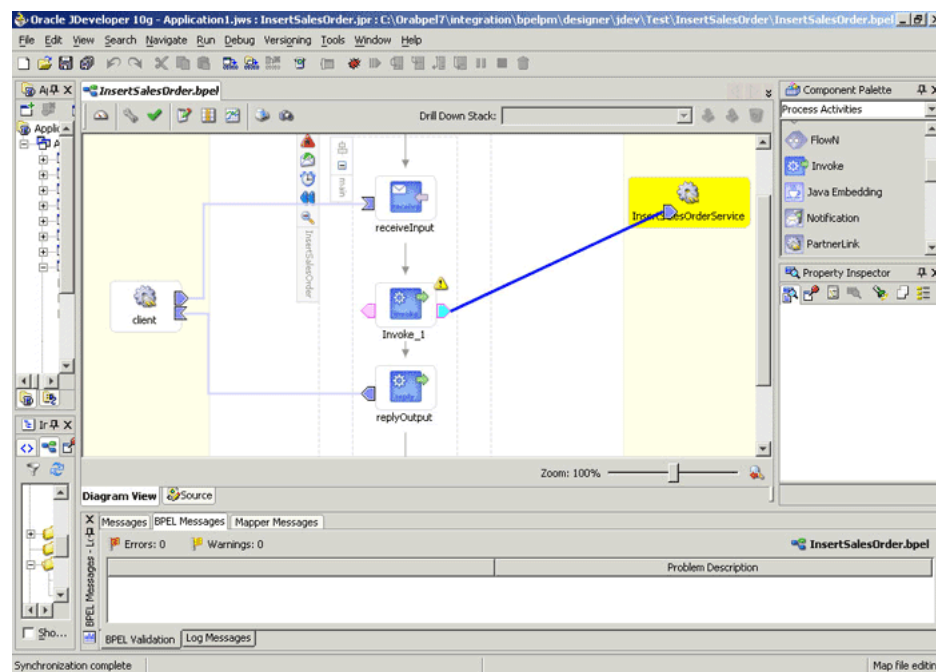
```
<element name="ISTRUE" ... db:default="true" .../>
<element name="VALUE" ... db:default="true" .../>
```

3.2.2.5 Configuring the Invoke Activity

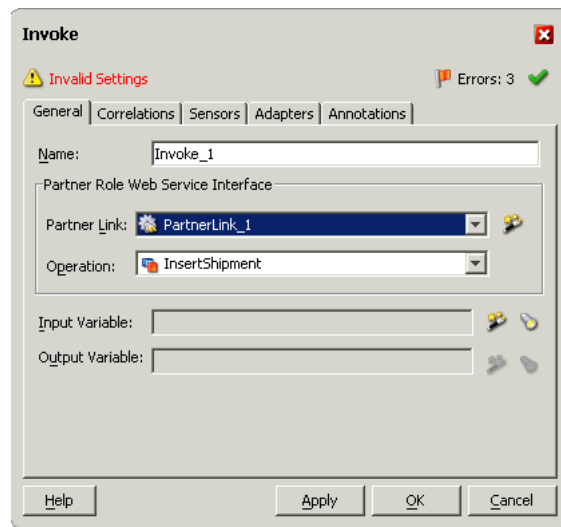
To configure the Invoke activity:

1. Drag **Invoke** from the Component palette and drop it at the location where you want to insert the invoke activity in your BPEL process, as shown in [Figure 3–14](#).

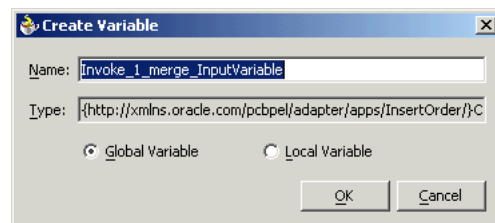
Figure 3–14 Dragging the Invoke Activity



2. Double-click **Invoke** in the process map to open the Invoke dialog box. The General tab is selected by default. [Figure 3–15](#) shows the Invoke dialog box.

Figure 3–15 Configuring the Invoke Activity

3. In the **Partner Link** box, select the partner link to invoke. This is the partner link that you configured in the previous section. The **Operation** is automatically selected.
4. Click the **Create** icon next to the Input Variable field. Enter a descriptive name for the variable in the Create Variable dialog box that appears. You can also accept the default name. Click **OK**. [Figure 3–16](#) shows the Create Variable dialog box.

Figure 3–16 Creating a Variable

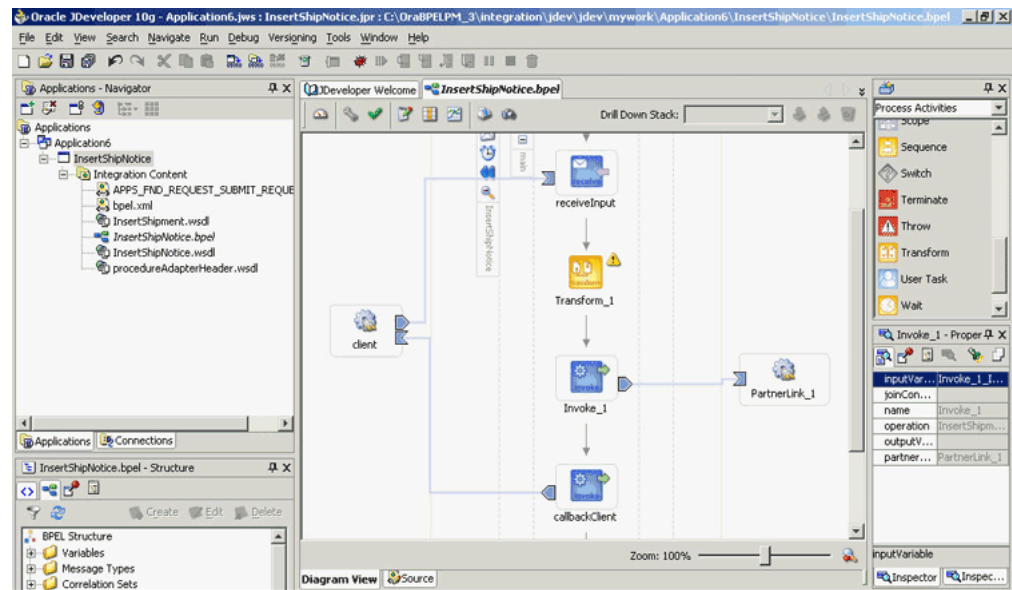
5. Click the **Create** icon next to the Output Variable field. Enter a descriptive name for the variable in the Create Variable dialog box that appears. You can also accept the default name. Click **OK**.
6. In the Invoke dialog box, click **Apply**, and then click **OK**. The invoke activity is configured.

3.2.2.6 Configuring the Transform Activity

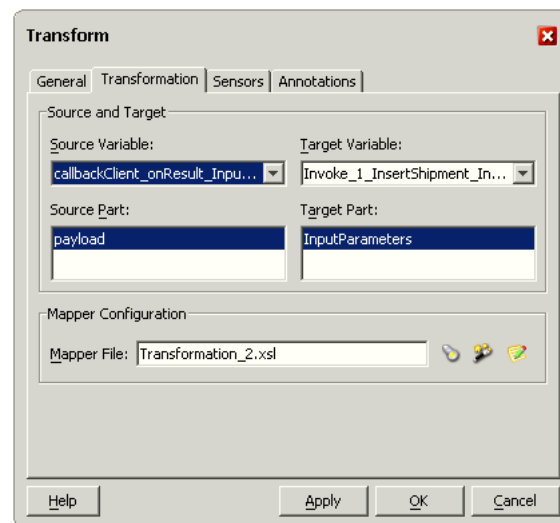
The Transform activity can be used to configure the parameters for the input and output variables. The Transform activity can also be used if variable values need to be transformed before updating them in Oracle Applications.

The following steps discuss configuring the Transform activity:

1. Drag and drop **Transform** into the process map window. The **Transform** activity should be placed in between **Receive** and **Invoke**. [Figure 3–17](#) shows the process map window after the Transform activity has been added.

Figure 3–17 Adding the Transform Activity

2. Double-click **Transform** in the process map to open the Transform dialog box. The Transformation tab is selected by default. Figure 3–18 shows the Transform dialog box.

Figure 3–18 Configuring the Transform Activity

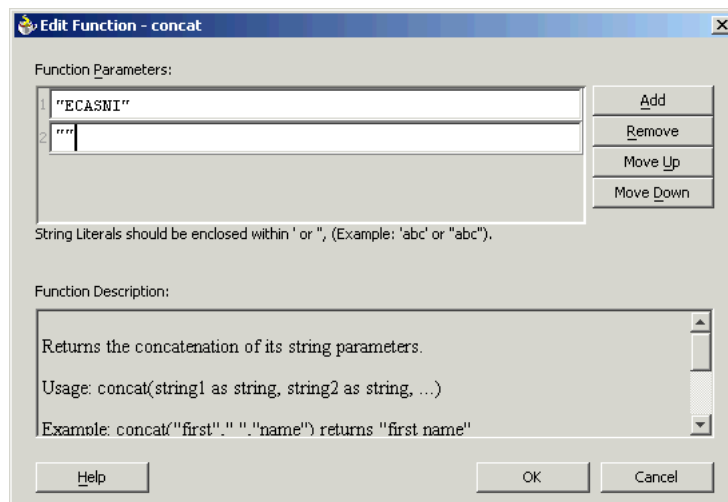
3. Select the **Source Variable** and **Target Variable** from the respective boxes. Elements are mapped from the **Source Variable** to the **Target Variable**.
4. Select the **Source Part** of the variable from which to map elements. For example, the source part may be a payload schema consisting of a purchase order request.
5. Select the **Target Part** of the variable to which to map elements. For example, the target part may be a payload schema consisting of a purchase order acknowledgment.

6. Click the **Create** icon next to the Mapper File field to create a new transformation mapping file. Mapper File specifies the file in which you create the mappings using the XSLT Mapper Transformation tool.
7. The transformation mapping file is displayed. The **Design** view is displayed by default.
8. You can define the parameter values in the **Design** view. Drag a string function to the Design area. Connect the function to the appropriate parameter for which you want to define a value.

Note: You can use an input parameter value from the source variable, transform it using a string function, and use it as the input parameter value for the target variable.

9. Double-click the icon for the function. The Edit Function dialog box is displayed. [Figure 3–19](#) shows the Edit Function dialog box.

Figure 3–19 Supplying the Function Parameters



10. Repeat steps 8 and 9 for all the parameters that you need to supply.

3.3 Run-Time Steps

After designing the BPEL process, the next step is to deploy, run and monitor it. This section discusses the following:

- [Deploying the BPEL Process](#)
- [Testing the BPEL Process](#)

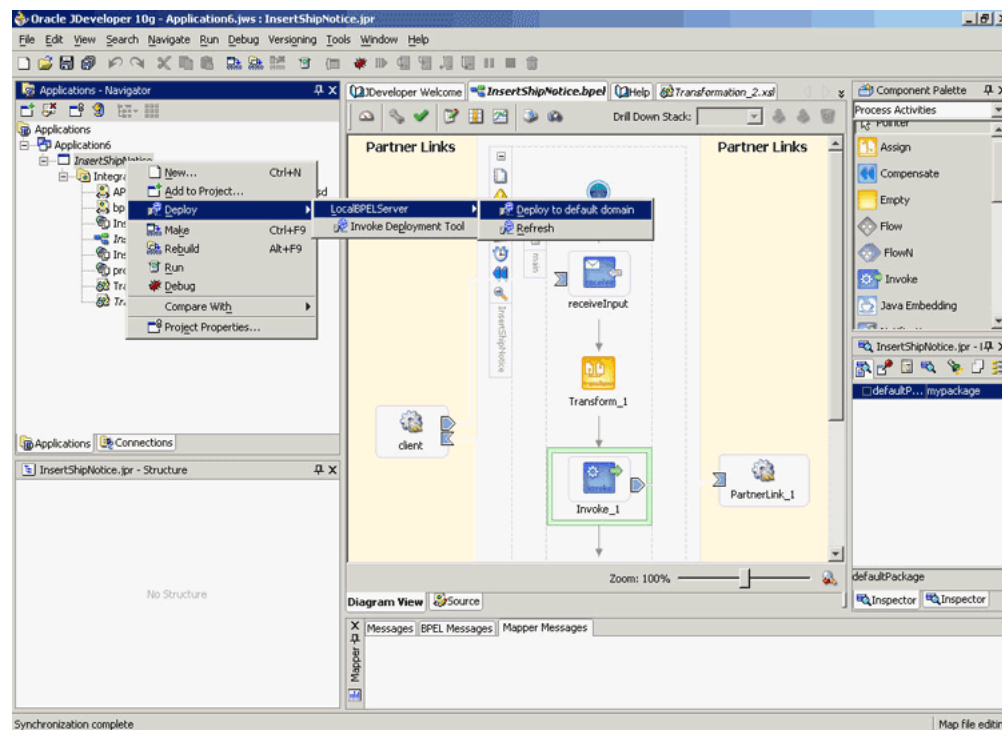
3.3.1 Deploying the BPEL Process

You must deploy the BPEL process before you can run it. The BPEL process is first compiled, and then deployed to the BPEL server. The following steps discuss deploying the BPEL process to a BPEL server:

1. Select the BPEL project in the Applications window.
2. Right-click the project name, and then select **Deploy** from the menu that appears.

3. Select **Local BPEL Server** followed by **Deploy to Default Domain**, if you are deploying the process on the local BPEL server. [Figure 3–20](#) illustrates deploying a BPEL process to a local BPEL server.

Figure 3–20 Deploying the BPEL Process



4. The Password Prompt dialog box appears. Enter the password for the default domain in the **Domain Password** field, and then click **OK**. [Figure 3–21](#) shows the Password Prompt dialog box.

Figure 3–21 Specifying the Domain Password



5. The BPEL process is compiled and deployed. You can check the progress in the Messages window. [Figure 3–22](#) shows the Messages window.

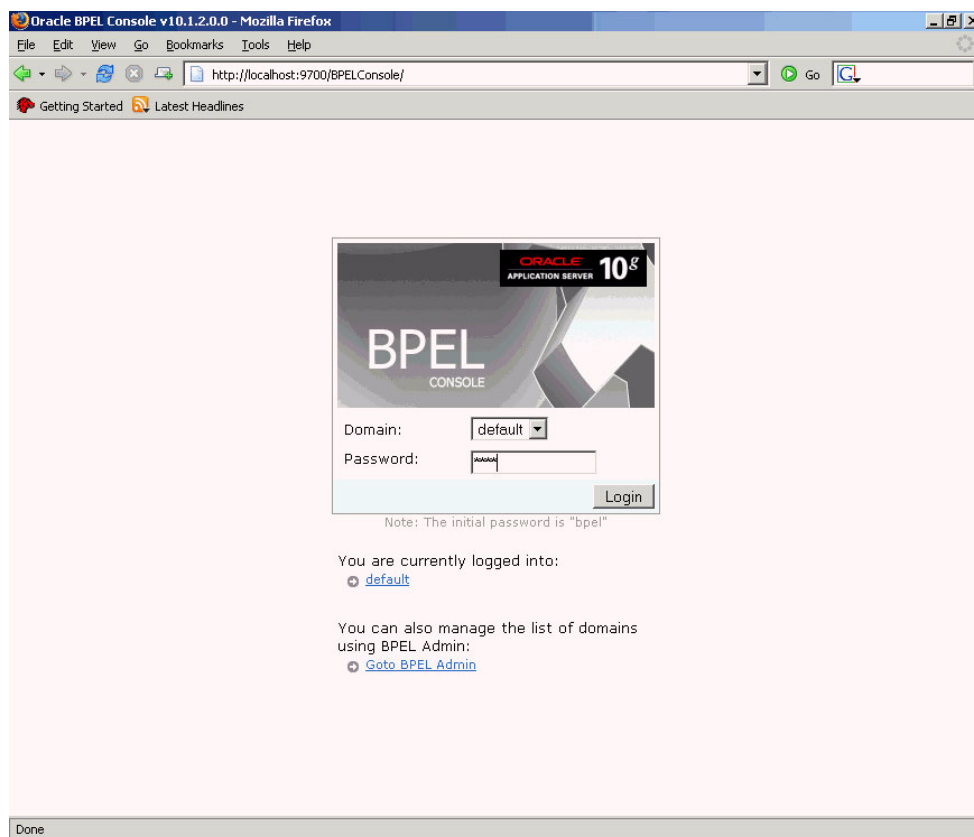
Figure 3–22 Messages Window

3.3.2 Testing the BPEL Process

Once the BPEL process is deployed, it can be seen in the BPEL console. You can manage and monitor the process from the BPEL console. You can also test the process and the integration interface by manually initiating the process. The following steps discuss manually initiating and monitoring the BPEL process:

1. To open the BPEL console, click **Start**, and then choose **Programs**. In the Programs menu, select **Oracle - ORACLE_HOME**, **Oracle BPEL Process Manager 10.1.2**, and then select **BPEL Console**.

The BPEL console login screen is displayed, as shown in [Figure 3–23](#).

Figure 3–23 BPEL Console Login Screen

2. Select **Default** in the **Domain** box. Enter the password for the default domain in the **Password** field, and then click **Login**.

Oracle BPEL console is displayed.

3. A list of deployed processes is shown under Deployed BPEL Processes. [Figure 3–24](#) shows the BPEL console screen.

Figure 3–24 Deployed BPEL Processes

The screenshot shows the Oracle BPEL Console v10.1.2.0.0 interface. The top navigation bar includes links for Manage BPEL Domain, Logout, and Support. The main content area has tabs for Dashboard, BPEL Processes, Instances, and Activities. The BPEL Processes tab is selected, showing a table of deployed processes and a section for recently completed instances.

Deployed BPEL Processes		In-Flight BPEL Process Instances		
Name	Instance	BPEL Process	Last Modified ↑	
InsertPurchaseOrder				
InsertShipNotice				
TaskActionHandler				
TaskManager				

Recently Completed BPEL Process Instances ([More...](#))

✓ 9 : Instance #9 of InsertPurchaseOrder	InsertPurchaseOrder (v. 1.0)	11/3/05 5:16:08 PM
✓ 8 : Instance #8 of InsertPurchaseOrder	InsertPurchaseOrder (v. 1.0)	11/3/05 12:32:59 PM
✓ 7 : Instance #7 of InsertPurchaseOrder	InsertPurchaseOrder (v. 1.0)	11/2/05 3:17:24 PM

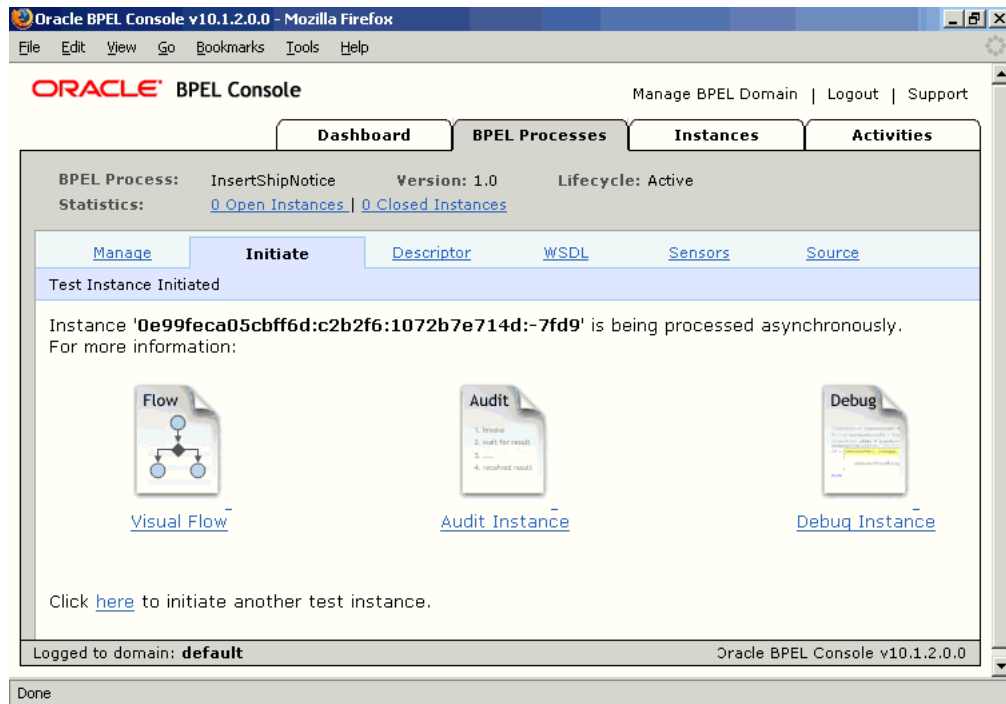
Deploy New Process

Logged to domain: **default** Oracle BPEL Console v10.1.2.0.0

http://localhost:9700/BPELConsole/default/displayProcess.jsp?processId=InsertShipNotice&revisionTag=1.0

4. Click the BPEL process that you want to initiate. The Initiate page is displayed. Enter the input string required by the process.
5. Click **Post XML Message** to initiate the process.
6. The BPEL process is now initiated. You can check the process flow by clicking the **Visual Flow** icon. shows the BPEL Console Initiate page.

Figure 3–25 BPEL Console Initiate Page



Note: To confirm that the records have been written into the Oracle Applications, you can write SQL `SELECT` statements and fetch the results showing the latest records inserted into Oracle Applications. Alternatively, you can go to the specific module in Oracle Applications and verify the appropriate changes in the records.

Using Concurrent Programs

OracleAS Adapter for Oracle Applications use concurrent programs to move data from interface tables to base tables.

This chapter includes the following sections:

- [Overview of Concurrent Programs](#)
- [Design-Time Steps](#)
- [Run-Time Steps](#)

4.1 Overview of Concurrent Programs

A concurrent program is an instance of an execution file, along with parameter definitions and incompatibilities. Concurrent programs use concurrent program executable to locate the correct execution file. Several concurrent programs may use the same execution file to perform their specific tasks, each having different parameter defaults and incompatibilities.

A concurrent program executable links either an execution file or the method used to execute it, or both with a defined concurrent program. Under concurrent processing, an execution method may be a program written in a standard language, a reporting tool, or an operating system language.

In the Oracle Applications 11.5.10 release only the concurrent programs, which have *PL/SQL stored procedure* as the execution method, would be supported.

4.2 Design-Time Steps

This section describes how to configure the OracleAS Adapter for Oracle Applications to use concurrent programs. It includes the following topics:

- [Prerequisites to Configuring OracleAS Adapter for Oracle Applications](#)
- [Configuring OracleAS Adapter for Oracle Applications](#)

4.2.1 Prerequisites to Configuring OracleAS Adapter for Oracle Applications

OracleAS adapter for Oracle Applications is deployed using the BPEL Process Manager (PM) in Oracle JDeveloper. The BPEL PM creates the WSDL interfaces for the concurrent programs

You must populate certain variables in the BPEL PM in order to provide context information for Oracle Applications. The context information required for a concurrent program includes the *username* and *responsibility* of an Oracle Applications user that

has sufficient privileges to run the program. The default value passed for the username is SYSADMIN. The default value passed for responsibility is SYSTEM ADMINISTRATOR.

You can change the default values specified in the generated WSDL for the username and responsibility. This is a static way of changing the context information. These values would apply to all invocations of the deployed business process. However, if you must provide different context information for different invocations of the business process, then you can dynamically populate the header variable with values for username and responsibility. The context information can be specified by configuring an Assign activity before the Invoke activity in the BPEL PM.

4.2.2 Configuring OracleAS Adapter for Oracle Applications

This section describes the tasks required to configure OracleAS Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper.

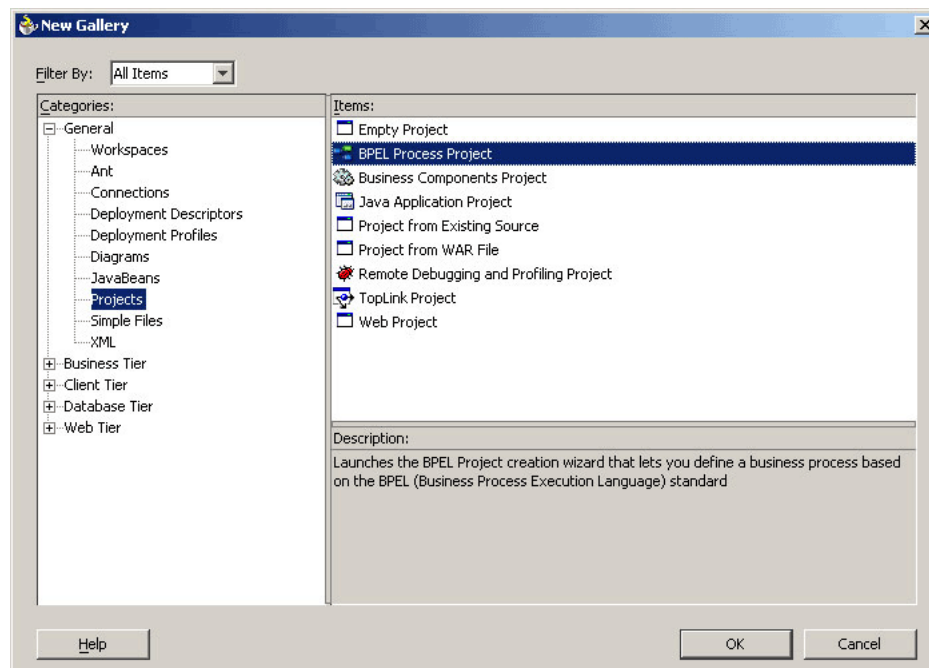
This section comprises the following:

- [Creating a New BPEL Project](#)
- [Adding a Partner Link](#)
- [Configuring the Invoke Activity](#)
- [Configuring the Transform Activity](#)

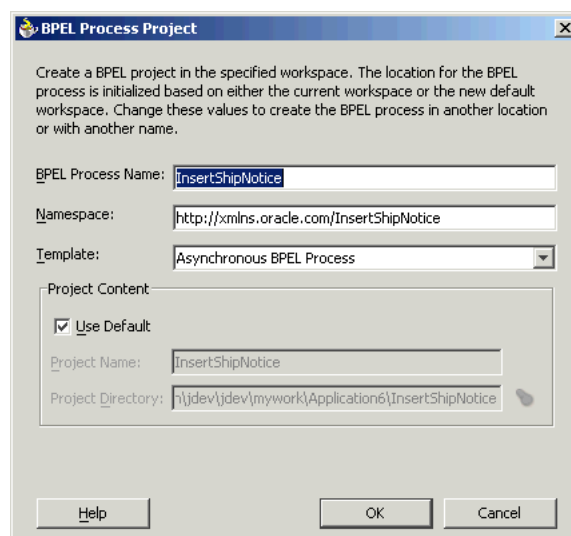
4.2.2.1 Creating a New BPEL Project

The first configuration task is to create a new BPEL project. Use the following steps to create a new BPEL project:

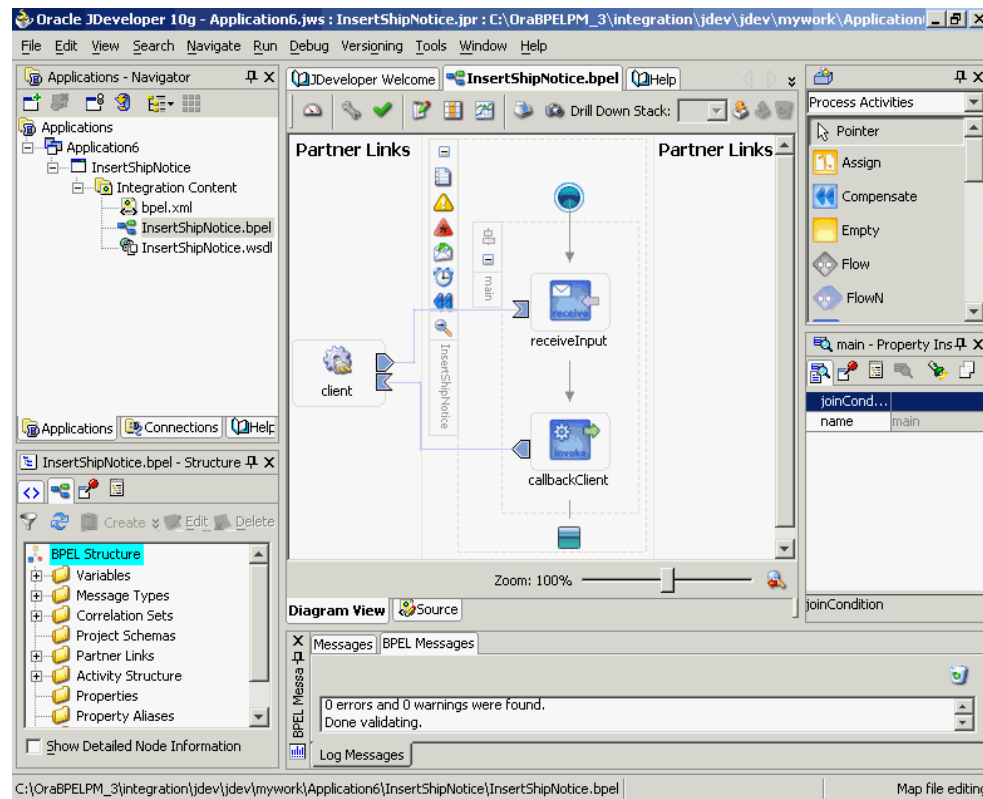
1. Open JDeveloper BPEL Designer.
2. From the **File** menu, select **New**. The New Gallery dialog box is displayed.
3. Select **All Items** from the **Filter By** box. This displays a list of available categories.
4. Expand the **General** node, and then select **Projects**.
5. Select **BPEL Process Project** from the **Items** group, as shown in [Figure 4–1](#).

Figure 4–1 Creating a New BPEL Process Project

6. Click **OK**. The BPEL Process Project dialog box is displayed.
7. In the **BPEL Process Name** field, enter a descriptive name. For example, `InsertShipNotice`.
8. From the **Template** box, select **Asynchronous BPEL Process**. Keep the default selection for **Use Default** under Project Content, as shown in [Figure 4–2](#).

Figure 4–2 Specifying a Name for the New BPEL Process Project

9. Click **OK**. A new BPEL process, with the required source files including `bpel.xml`, `InsertShipNotice.bpel`, and `InsertShipNotice.wsdl` is created, as shown in [Figure 4–3](#).

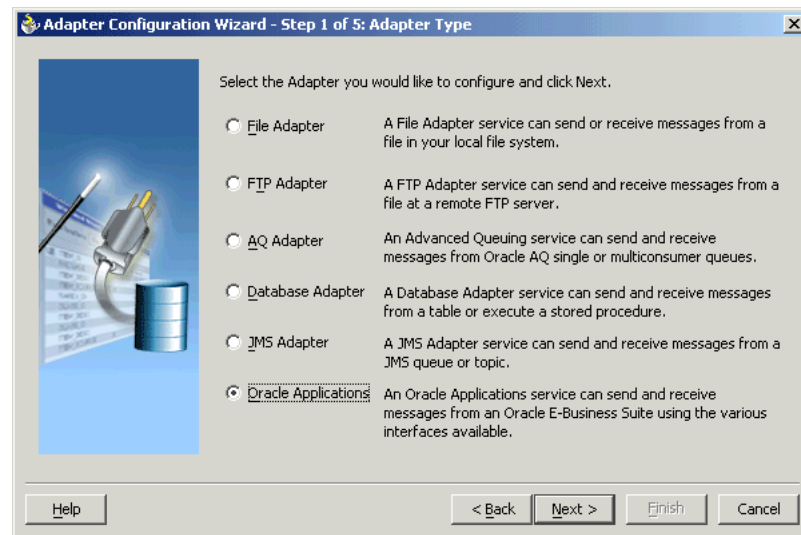
Figure 4–3 New BPEL Process Project

4.2.2.2 Adding a Partner Link

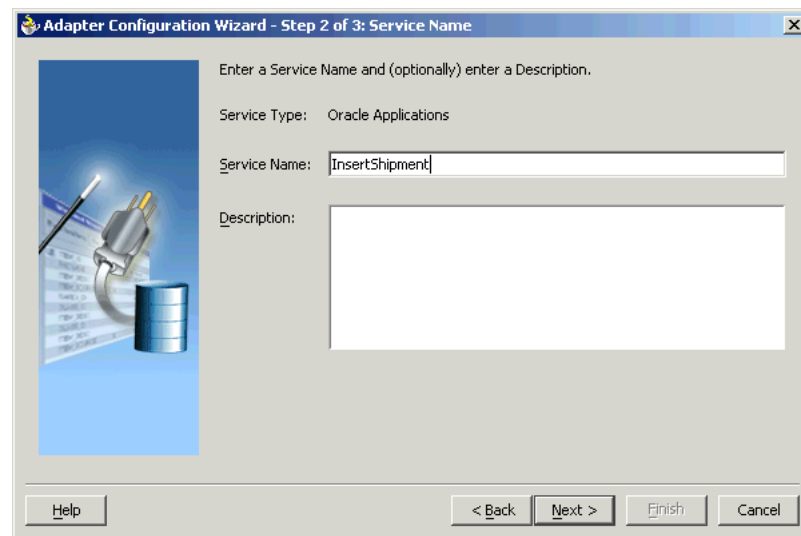
The next task is to add a partner link to the BPEL process. A partner link defines the link name, type, and the role of the BPEL process that interacts with the partner service.

Use the following steps to add a partner link:

1. Drag and drop **PartnerLink**, from the Component Palette, into the border area of the process diagram. The Create Partner Link dialog box is displayed.
2. Click the **Define Adapter Service** icon in WSDL Settings. The Adapter Configuration Wizard is displayed.
3. Click **Next**. The Adapter Type dialog box is displayed.
4. Select **Oracle Applications**, as shown in [Figure 4–4](#), and then click **Next**.

Figure 4–4 Selecting OracleAS Adapter for Oracle Applications

5. The Service Name dialog box is displayed, as shown in [Figure 4–5](#). Enter the following information:
 - a. In the **Service Name** field, enter a service name.
 - b. In the **Description** field, enter a description for the service. This is an optional field. Click **Next**.

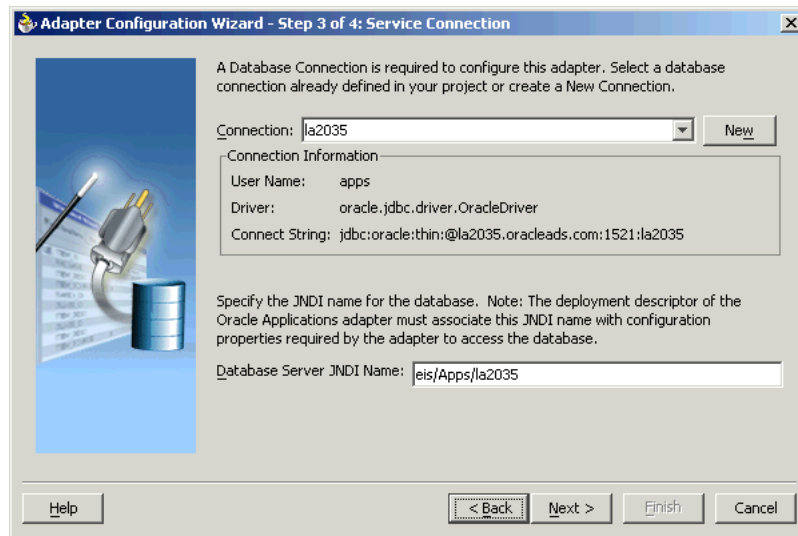
Figure 4–5 Specifying the Service Name

6. The Service Connection dialog box is displayed. Enter the Java Naming and Directory Interface (JNDI) name in the **Database Server JNDI Name** field. The JNDI name acts as a placeholder for the connection used when your service is deployed to the BPEL server. This enables you to use different databases for development and later production.

See Also: *Oracle Application Server Adapter Concepts* for understanding JNDI concepts

Figure 4–6 shows how to create a new database connection.

Figure 4–6 Creating a New Database Connection

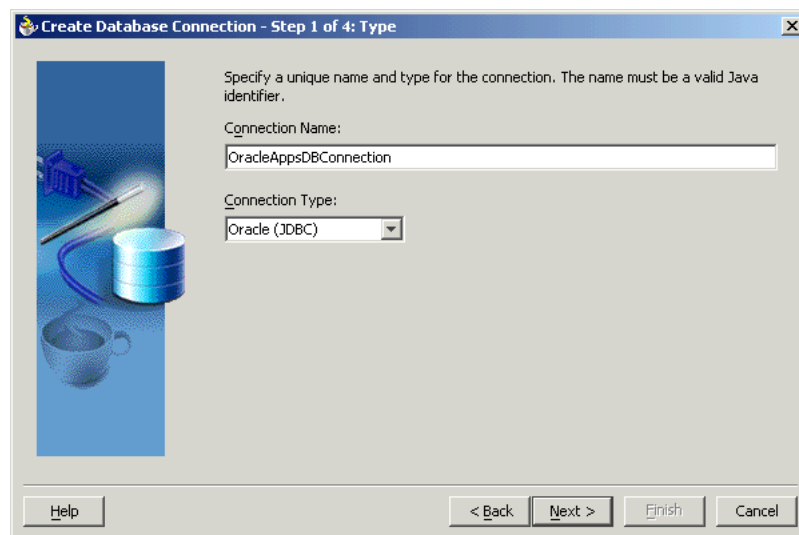


7. Click **New** to define a database connection. The Create Database Connection Wizard is displayed.

Note: You must connect to the database where Oracle Applications is running.

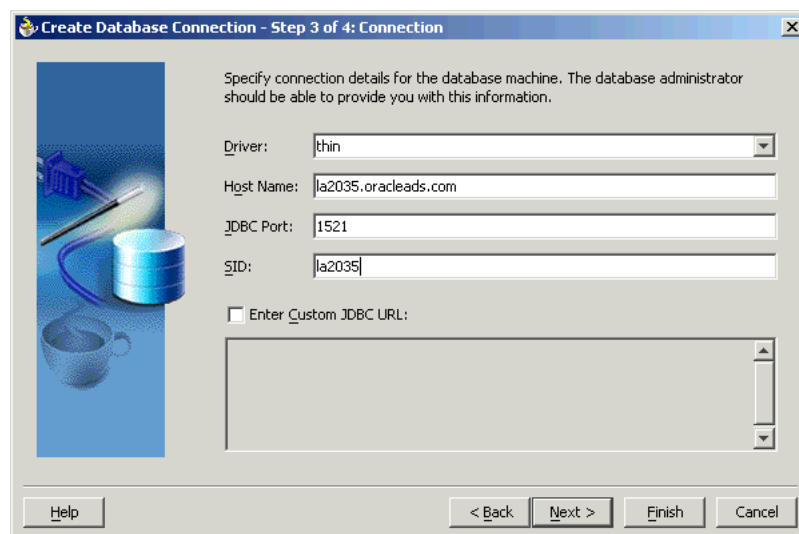
8. Enter the following information in the Type dialog box:
 - a. In the **Connection Name** field, specify a unique name for the database connection.
 - b. From the **Connection Type** box, select the type of connection for your database connection.

Figure 4–7 shows the Type dialog box.

Figure 4–7 Specifying the Connection Name and Type of Connection

9. Click **Next**. The Authentication dialog box is displayed.
10. Enter information in the following fields:
 - a. In the **UserName** field, specify a unique name for the database connection.
 - b. In the **Password** field, specify a password for the database connection.
11. Click **Next**. The Connection dialog box is displayed.
12. Enter information in the following fields:
 - a. From the **Driver** list, select **Thin**.
 - b. In the **Host Name** field, specify the host name for the database connection.
 - c. In the **JDBC Port** field, specify the port number for the database connection.
 - d. In the **SID** field, specify a unique SID value for the database connection.

Figure 4–8 shows the Connection dialog box.

Figure 4–8 Specifying the New Database Connection Information

13. Click **Next**. The Test dialog box is displayed.
14. Click **Test Connection** to determine whether the specified information establishes a connection with the database.
15. Click **Next**. The Service Connection dialog box is displayed, providing a summary of the database connection.
16. Click **Finish** to complete the process of creating a new database connection.

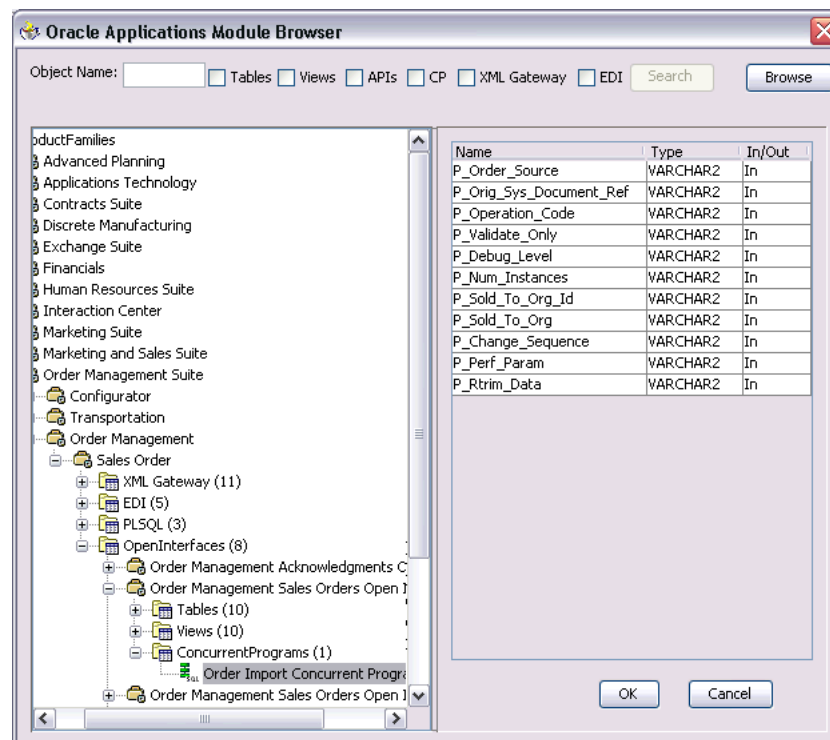
Once you have completed creating a new connection for the service, you can add a concurrent program by browsing through the list of concurrent programs available in Oracle Applications.

17. Click **Next** in the Service Connection dialog box. The Operation dialog box is displayed.

Note: If you are connecting to a pre-11.5.10 Oracle Applications instance, then you would be required to select the interface to Oracle Applications data. Select **Tables/Views/APIs/Concurrent Programs** to proceed.

18. Click **Get Object** to open the Oracle Applications Module Browser. [Figure 4–9](#) shows the Oracle Applications Module Browser.

Figure 4–9 Selecting a Concurrent Program from the Module Browser



Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. For example, Applications Technology or Order Management Suite are product families in Oracle Applications. The product families contain the individual products. For example, Order Management Suite contains the Order Management product. The individual products contain the

business entities associated with the product. For example, the Order Management product contains the Sales Order business entity.

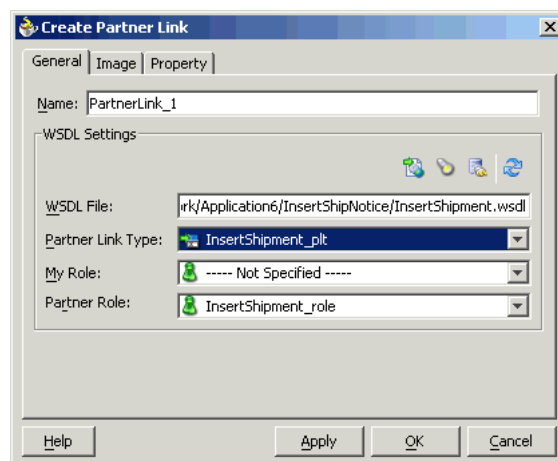
Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide. concurrent programs can be found under the OpenIntefaces category.

19. Select a concurrent program, and then click **OK**. You can select only one concurrent program for each adapter service.

Note: You can also search for a concurrent program by entering the name of the program in the **Object Name** field. Select the **CP** check box, and then click **Search**.

20. The concurrent program is added to Operation Objects. Click **Next** in the Operation dialog box.
21. Click **Finish**. When you click **Finish**, the wizard generates the WSDL file corresponding to the selected interface. This WSDL file is now available for the partner link. [Figure 4–10](#) shows the Create Partner Link dialog box after the WSDL file has been generated.

Figure 4–10 *Completing the Partner Link Configuration*



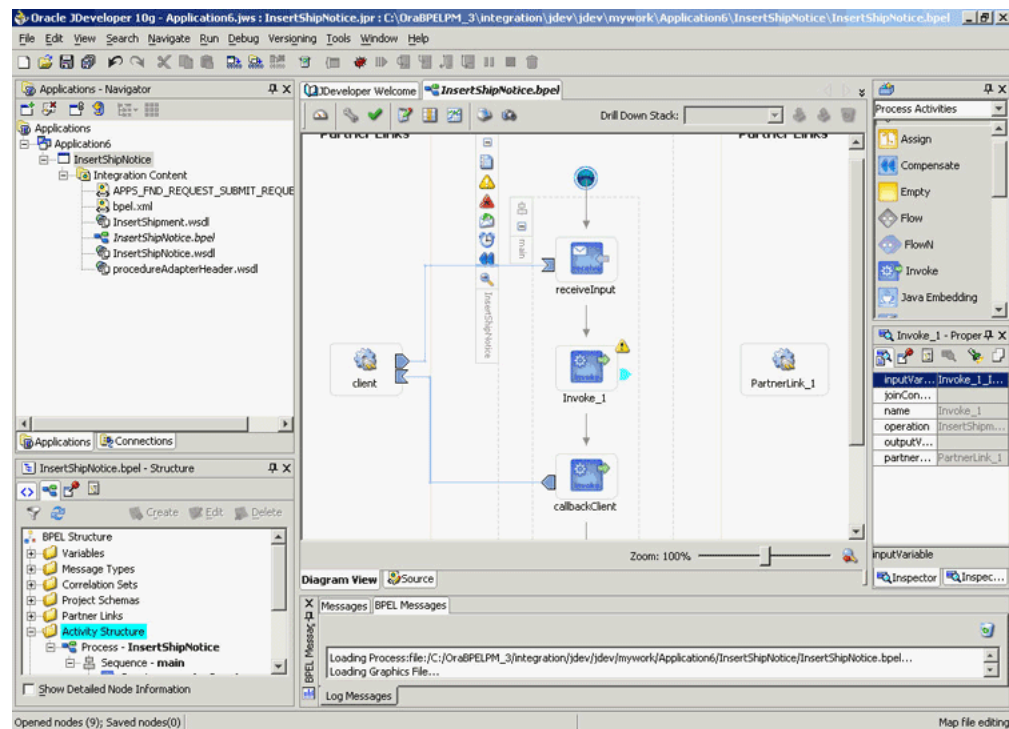
22. Click **OK**. The partner link is created with the required WSDL settings.

4.2.2.3 Configuring the Invoke Activity

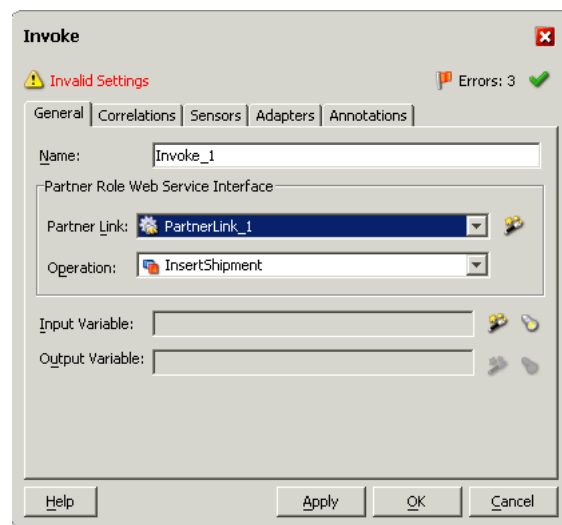
After adding and configuring the partner link, the next task is to configure the BPEL process. This task includes configuring the Invoke activity, and the Transform activity.

The following steps discuss configuring the Invoke activity:

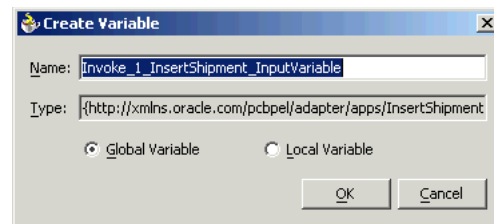
1. Drag and drop **Invoke** into the process map window. [Figure 4–11](#) shows the **Invoke** activity after it has been added to the process map.

Figure 4–11 Adding the Invoke Activity

2. Double-click **Invoke** in the process map to open the Invoke dialog box. The **General** tab is selected by default. [Figure 4–12](#) shows the Invoke dialog box.

Figure 4–12 Configuring the Invoke Activity

3. In the **Partner Link** box, select the partner link to invoke. This is the partner link that you configured in the previous section. The **Operation** is automatically selected, depending on the concurrent program that you chose when configuring the partner link.
4. Click the **Create** icon next to the **Input Variable** field. Enter a descriptive name for the variable in the Create Variable dialog box that appears. You can also accept the default name. Click **OK**. [Figure 4–13](#) shows the Create Variable dialog box.

Figure 4–13 Creating the Input Variable

5. Click the **Create** icon next to the **Output Variable** field. Enter a descriptive name for the variable in the Create Variable dialog box that appears. You can also accept the default name. Click **OK**.
6. In the Invoke dialog box, click **Apply**, and then click **OK**.

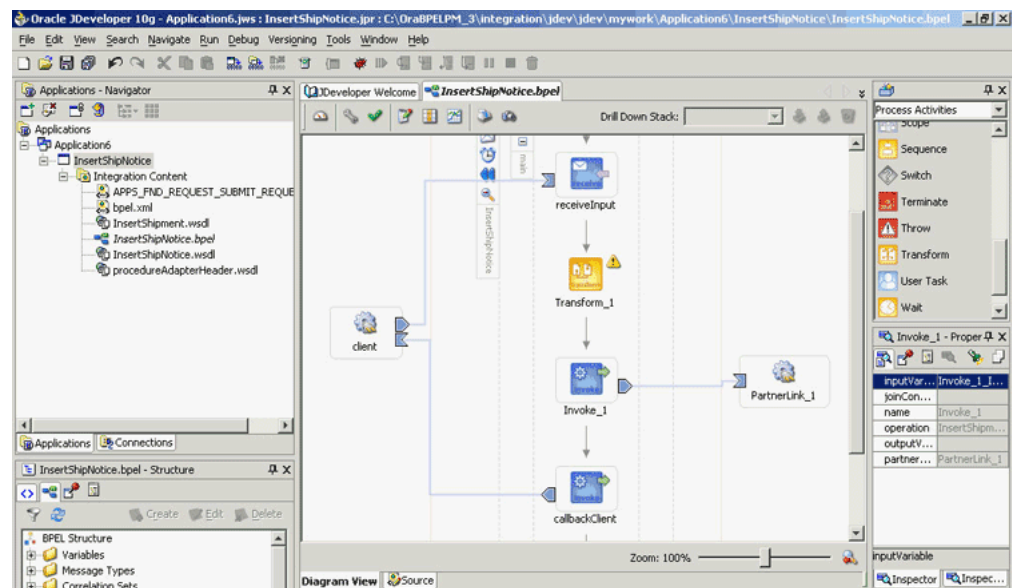
Note: You can define a **Input Header Variable** under the **Adapters** tab of the Invoke dialog box. This variable can be used to provide context information for Oracle Applications.

4.2.2.4 Configuring the Transform Activity

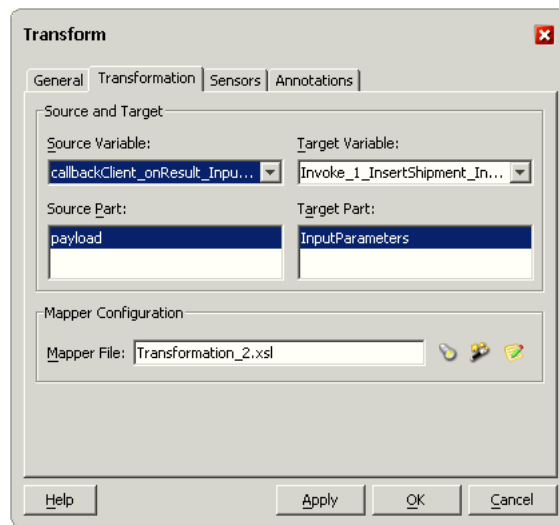
The Transform activity can be used to configure the parameters for the input and output variable. The Transform activity can also be used if variable values need to be transformed before updating them in Oracle Applications.

The following steps discuss configuring the Transform activity:

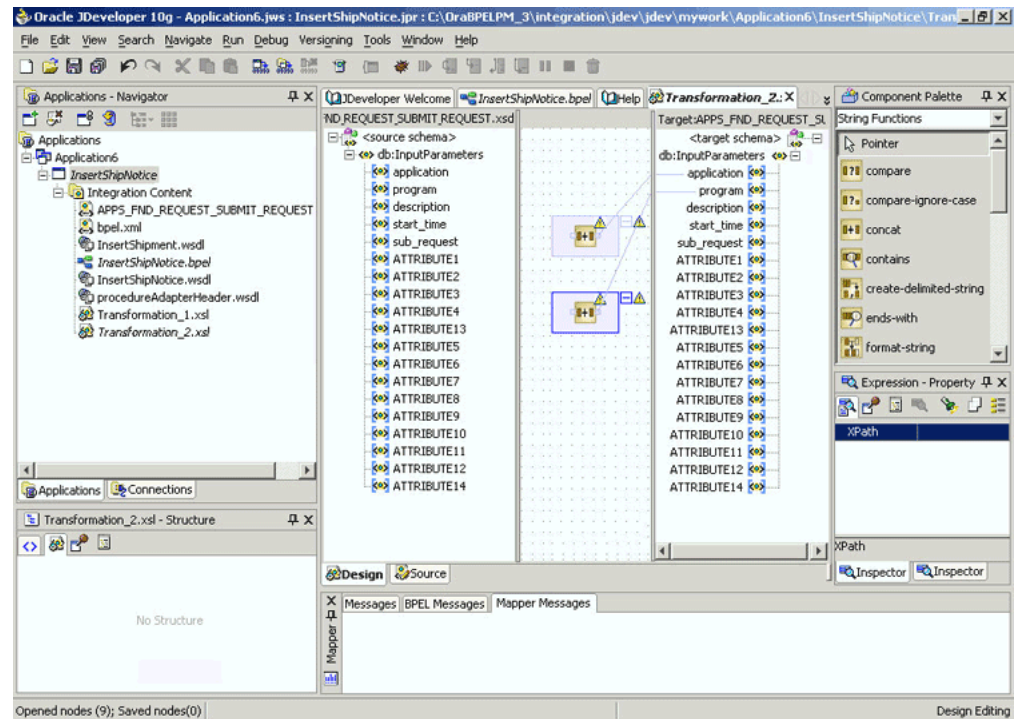
1. Drag and drop **Transform** into the process map window. The **Transform** activity should be placed in between **Receive** and **Invoke**. Figure 4–14 shows the process map window after the **Transform** activity has been added.

Figure 4–14 Adding the Transform Activity

2. Double-click **Transform** in the process map to open the Transform dialog box. The **Transformation** tab is selected by default. Figure 4–15 shows the Transform dialog box.

Figure 4–15 Configuring the Transform Activity

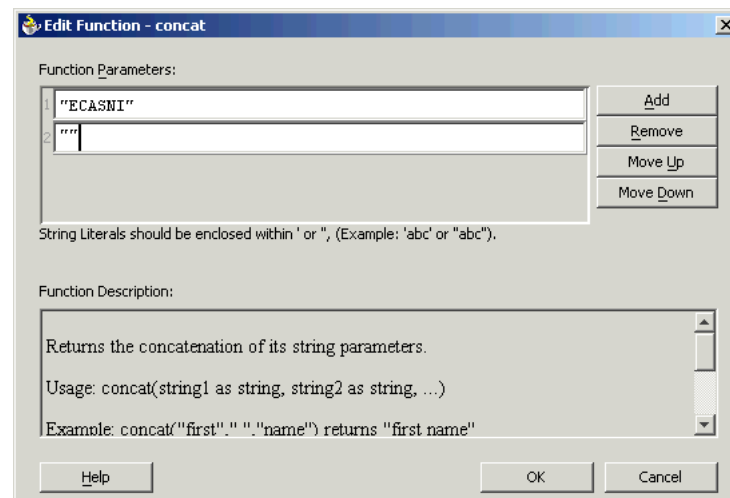
3. Select the **Source Variable** and **Target Variable** from the respective boxes. Elements are mapped from the **Source Variable** to the **Target Variable**.
4. Select the **Source Part** of the variable from which to map elements. For example, the source part may be a payload schema consisting of a purchase order request.
5. Select the **Target Part** of the variable to which to map elements. For example, the target part may be a payload schema consisting of a purchase order acknowledgment.
6. **Mapper File** specifies the file in which you create the mappings using the XSLT Mapper Transformation tool. Click the **Create** icon next to the **Mapper File** field to create a new transformation mapping file.
The transformation mapping file is displayed.
7. The **Design** view is displayed by default. [Figure 4–16](#) shows the **Design** view for the transformation mapping file.

Figure 4–16 Editing the Transformation Mapping File

8. You can define the parameter values in the **Design** view. Drag a string function to the **Design** area. Connect the function to the appropriate parameter for which you want to define a value.

Note: You can use an input parameter value from the source variable, transform it using a string function, and then use it as the input parameter value for the target variable.

9. Double-click the icon for the function. The Edit Function dialog box is displayed. [Figure 4–17](#) shows the Edit Function dialog box.

Figure 4–17 Supplying the Function Parameters

10. Repeat steps 8 and 9 for all the parameters that you must supply.

Note: You can use either the Transform activity or the Assign activity to configure the variables. If the number of mappings to be configured is less, say three or four, then you can use the Assign activity. The Transform activity is more suitable for situations where the number of mappings is large or where variable transformation is required.

4.3 Run-Time Steps

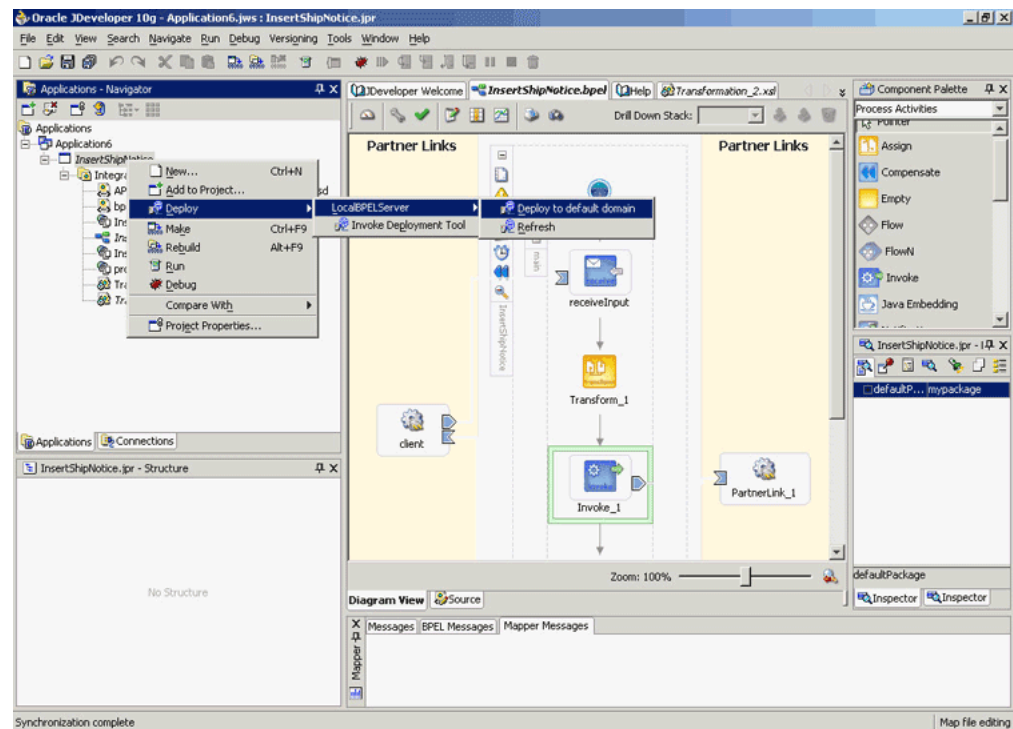
After designing the BPEL process, the next step is to deploy, run and monitor it. This section discusses the following:

- [Deploying the BPEL Process](#)
- [Testing the BPEL Process](#)
- [Verifying Records in Oracle Applications](#)

4.3.1 Deploying the BPEL Process

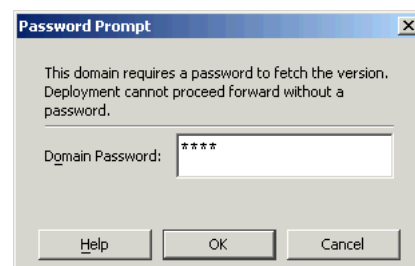
You must deploy the BPEL process before you can run it. The BPEL process is first compiled, and then deployed to the BPEL server. The following steps discuss deploying the BPEL process to a BPEL server:

1. Select the BPEL project in the Applications window.
2. Right-click the project name, and then select **Deploy** from the menu that appears.
3. Select **Local BPEL Server** followed by **Deploy to Default Domain**, if you are deploying the process on the local BPEL server. [Figure 4–18](#) illustrates deploying a BPEL process to a local BPEL server.

Figure 4–18 Deploying the BPEL Process

Note: You can select **Invoke Deployment Tool** if you want to deploy to a different BPEL server.

4. The Password Prompt dialog box appears. Enter the password for the default domain in the **Domain Password** field, and then click **OK**. Figure 4–19 shows the Password Prompt dialog box.

Figure 4–19 Specifying the Domain Password

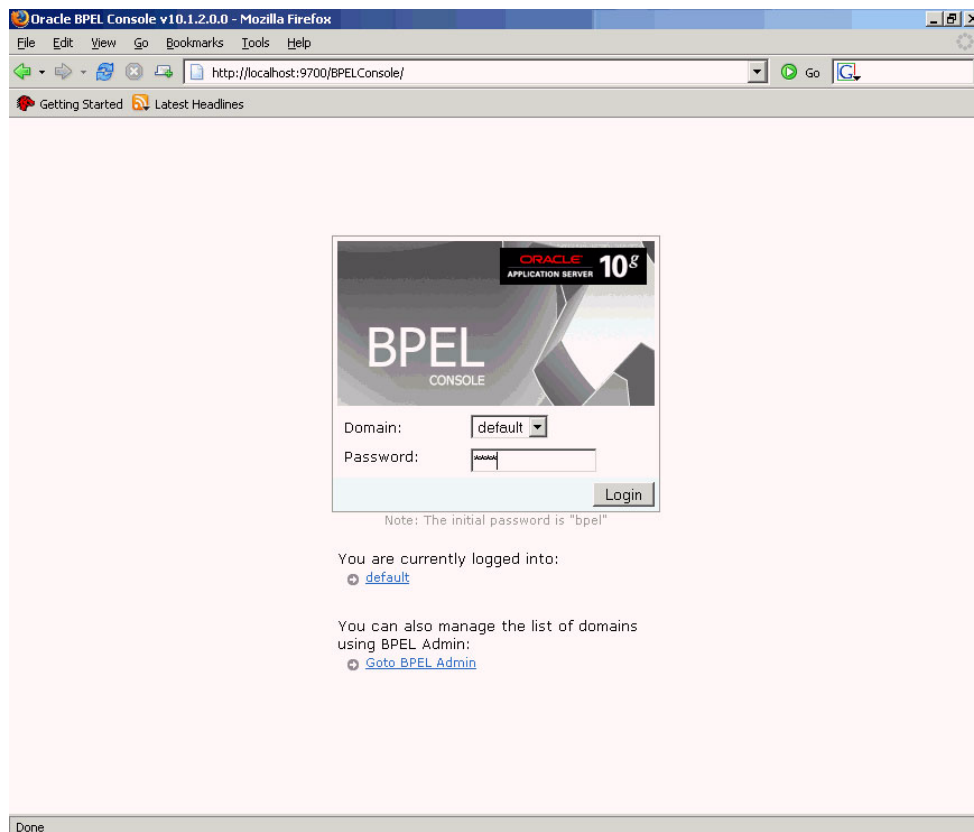
5. The BPEL process is compiled and deployed. You can check the progress in the Messages window. Figure 4–20 shows the Messages window.

Figure 4–20 Messages Window

4.3.2 Testing the BPEL Process

Once the BPEL process is deployed, it can be seen in the BPEL console. You can manage and monitor the process from the BPEL console. You can also test the process and the integration interface by manually initiating the process. The following steps discuss manually initiating and monitoring the BPEL process:

1. To open the BPEL console, click **Start**, and then choose **Programs**. In the Programs menu, select **Oracle - ORACLE_HOME, Oracle BPEL Process Manager 10.1.2**, and then click **BPEL Console**.
2. The BPEL console login screen is displayed. Select **Default** in the **Domain** box. Enter the password for the default domain in the **Password** field, and then click **Login**. Figure 4–21 shows the BPEL console login screen.

Figure 4–21 BPEL Console Login Screen

- Oracle BPEL console is displayed. The list of deployed processes is shown under Deployed BPEL Processes. Figure 4–22 shows the BPEL console screen.

Figure 4–22 Deployed BPEL Processes

Oracle BPEL Console v10.1.2.0.0 - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

Getting Started Latest Headlines

ORACLE BPEL Console Manage BPEL Domain | Logout | Support

Dashboard **BPEL Processes** **Instances** **Activities**

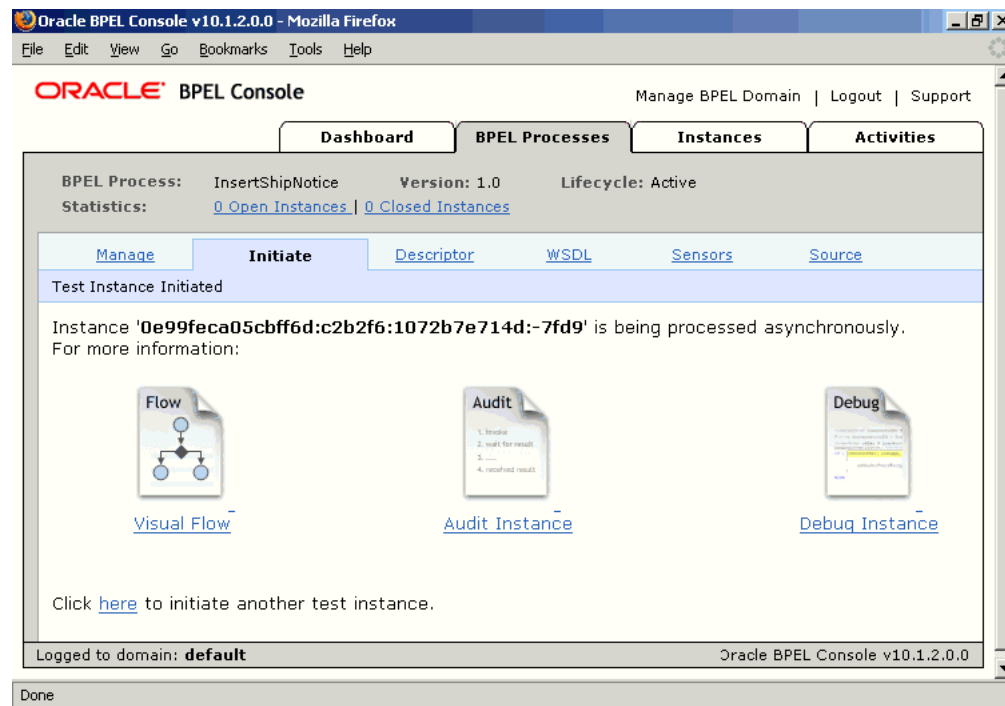
Deployed BPEL Processes		In-Flight BPEL Process Instances		
Name	Instance	BPEL Process	Last Modified ↑	
InsertPurchaseOrder				
InsertShipNotice				
TaskActionHandler				
TaskManager				
Recently Completed BPEL Process Instances (More...)				
		✓ 9 : Instance #9 of InsertPurchaseOrder	InsertPurchaseOrder (v. 1.0)	11/3/05 5:16:08 PM
		✓ 8 : Instance #8 of InsertPurchaseOrder	InsertPurchaseOrder (v. 1.0)	11/3/05 12:32:59 PM
		✓ 7 : Instance #7 of InsertPurchaseOrder	InsertPurchaseOrder (v. 1.0)	11/2/05 3:17:24 PM

Deploy New Process

Logged to domain: **default** Oracle BPEL Console v10.1.2.0.0

<http://localhost:9700/BPELConsole/default/displayProcess.jsp?processId=InsertShipNotice&revisionTag=1.0>

- Click the BPEL process that you want to initiate. The Initiate page is displayed. Enter the input string required by the process.
- Click **Post XML Message** to initiate the process.
- The BPEL process is now initiated. You can check the process flow by clicking the **Visual Flow** icon. Figure 4–23 shows the BPEL console Initiate page.

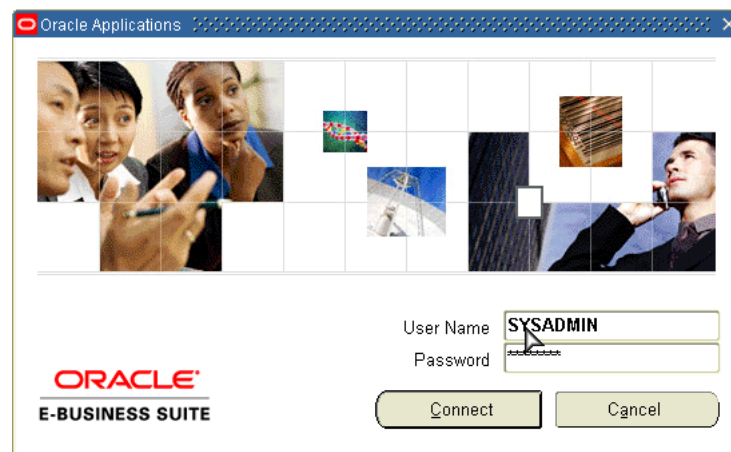
Figure 4–23 BPEL Console Initiate Page

7. The audit trail provides information about the steps that have been executed. The audit trail also records the Request ID that is returned for the transaction. You can check the audit trail by clicking the **Audit Instance** icon.

4.3.3 Verifying Records in Oracle Applications

You can use the following steps to track the process in Oracle Applications:

1. Log in to Oracle Applications as the System Administrator. [Figure 4–24](#) shows the Oracle Applications login screen.

Figure 4–24 Oracle Applications Login Screen

2. Select **Requests** from the **View** menu.

3. Search for the Request by entering the Request Id that you got from the audit trail, and then click **Find**. [Figure 4–25](#) shows the Find Requests dialog box.

Figure 4–25 Find Requests Dialog Box

4. The Request details are displayed. You can check for details such as the Phase and Status of the request.
5. If the Status of the Request is complete, then you can also query the appropriate table in Oracle Applications to search for the relevant records that have been inserted. [Figure 4–26](#) shows a simple SQL query on an Oracle Applications table.

Figure 4–26 Querying Oracle Applications for a Record

```
SQL> select      distinct(shipment_num) from    rcv_headers_interface where s
shipment_num
=          'S1722427';

SHIPMENT_NUM
-----
S1722427

SQL> █
```

Using e-Commerce Gateway

Oracle e-Commerce Gateway provides a common, standards-based approach for Electronic Data Interchange (EDI) integration between Oracle Applications and third party applications. EDI is the computer to computer exchange of business documents in a standard format. The EDI format is commonly used for e-commerce transactions between businesses.

This chapter includes the following sections:

- [Overview of e-Commerce Gateway Integration](#)
- [Design-Time Steps](#)
- [Run-Time Steps](#)

5.1 Overview of e-Commerce Gateway Integration

Oracle e-Commerce Gateway is the EDI integration enabler for Oracle Applications. It provides a single integration framework for you to conduct e-business using EDI standards with everyone in your global supply chain. Oracle e-Commerce Gateway provides an application integration infrastructure that is flexible enough to accommodate the integration requirements of any and all applications that must integrate with Oracle Applications. This allows for seamless flow of information in an ever expanding trading partner base.

Oracle e-Commerce Gateway includes pre-built transactions of key business documents that can be implemented simply by defining a trading partner and enabling the transaction in test or production mode. You can implement a single transaction, a group of transactions, or a business flow. You can implement the pre-built transactions as is or configure them to meet your specific industry needs.

Oracle e-Commerce Gateway uses a metadata driven approach to dynamically generate outbound and consume inbound flat files based on user defined trading partner, mapping, transformation, and data validation rules. You can change a rule by changing the metadata stored in the repository. The updated rule takes effect at run-time without any code modifications.

See Also: *Oracle e-Commerce Gateway User's Guide* for detailed information on Oracle e-Commerce Gateway. This guide is a part of the Oracle Applications documentation library. Oracle Applications documentation can be accessed from the following link:

<http://www.oracle.com/technology/documentation/applications.html>

OracleAS Adapter for Applications can be configured to use e-Commerce Gateway to interact with third party applications. e-Commerce Gateway, like XML Gateway, is primarily used for Business-to-Business (B2B) integration. While XML transactions are mostly based on a single transaction and are event based, EDI transactions are more batch oriented.

5.2 Design-Time Steps

OracleAS adapter for Oracle Applications is deployed using the BPEL Process Manager (PM) in Oracle JDeveloper. The BPEL PM creates the WSDL interfaces for the e-Commerce Gateway.

This section describes configuring the OracleAS Adapter for Oracle Applications to use e-Commerce Gateway. It includes the following topics:

- [Prerequisites to Configuring OracleAS Adapter for Oracle Applications](#)
- [Configuring OracleAS Adapter for Oracle Applications](#)

5.2.1 Prerequisites to Configuring OracleAS Adapter for Oracle Applications

You need to populate certain variables in the BPEL PM in order to provide context information for Oracle Applications. The context information required for an EDI transaction includes the *username* and *responsibility* of an Oracle Applications user that has sufficient privileges to run the program. The default value passed for the username is SYSADMIN. The default value passed for responsibility is SYSTEM ADMINISTRATOR.

You can change the default values specified in the generated WSDL for the username and responsibility. This is a static way of changing the context information. These values would apply to all invocations of the deployed business process. However, if you need to provide different context information for different invocations of the business process, then you can dynamically populate the header variable with values for username and responsibility. The context information can be specified by configuring an Assign activity before the Invoke activity in the BPEL PM.

5.2.2 Configuring OracleAS Adapter for Oracle Applications

This section describes the tasks required to configure OracleAS Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper.

This section comprises the following:

- [Creating a New BPEL Project](#)
- [Adding a Partner Link](#)
- [Configuring the Invoke Activity](#)
- [Configuring the Transform Activity](#)

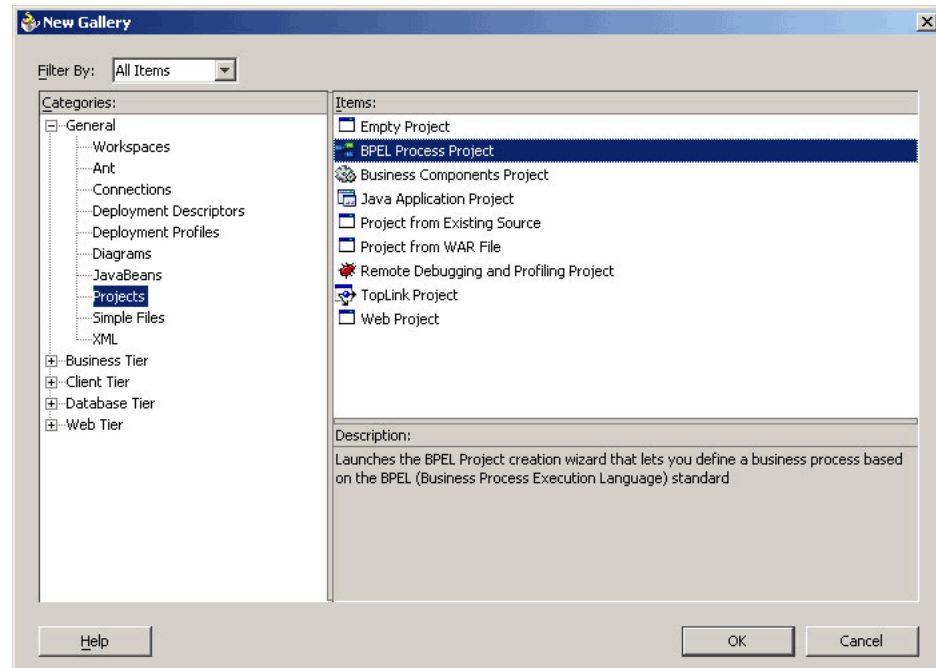
5.2.2.1 Creating a New BPEL Project

The first configuration task is to create a new BPEL project. Use the following steps to create a new BPEL project:

1. Open JDeveloper BPEL Designer.
2. From the **File** menu, select **New**. The New Gallery dialog box is displayed.
3. Select **All Items** from the **Filter By** box. This displays a list of available categories.

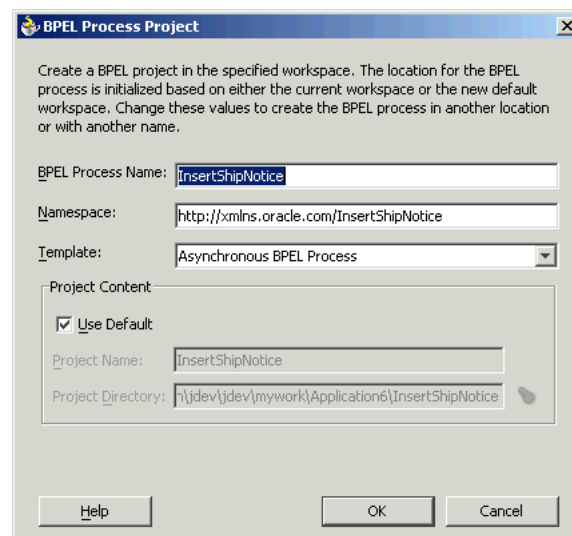
4. Expand the **General** node, and then select **Projects**.
5. Select **BPEL Process Project** from the **Items** group, as shown in [Figure 5–1](#).

Figure 5–1 Creating a New BPEL Process Project



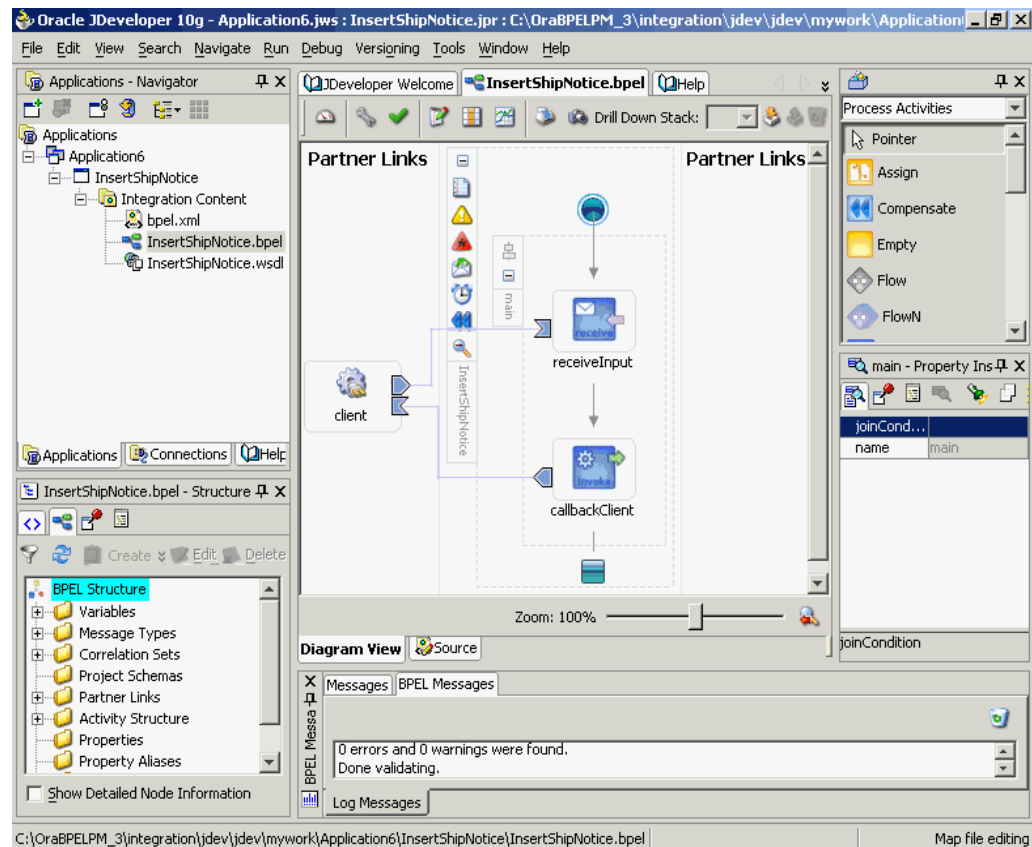
6. Click **OK**. The BPEL Process Project dialog box is displayed.
7. In the **BPEL Process Name** field, enter a descriptive name. For example, `InsertShipNotice`.
8. From the **Template** box, select **Asynchronous BPEL Process**. Keep the default selection for **Use Default** under Project Content, as shown in [Figure 5–2](#).

Figure 5–2 Specifying a Name for the New BPEL Process Project



- Click **OK**. A new BPEL process, with the required source files including `bpel.xml`, `InsertShipNotice.bpel`, and `InsertShipNotice.wsdl` is created, as shown in [Figure 5-3](#).

Figure 5-3 New BPEL Process Project

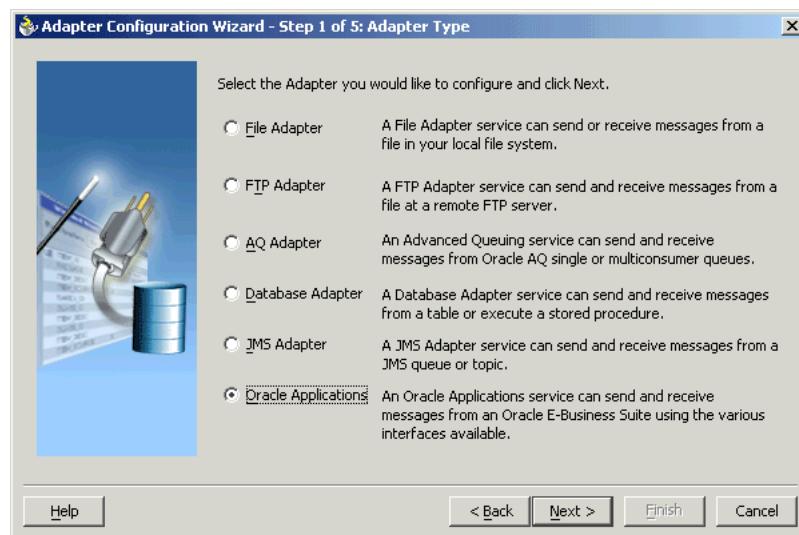


5.2.2.2 Adding a Partner Link

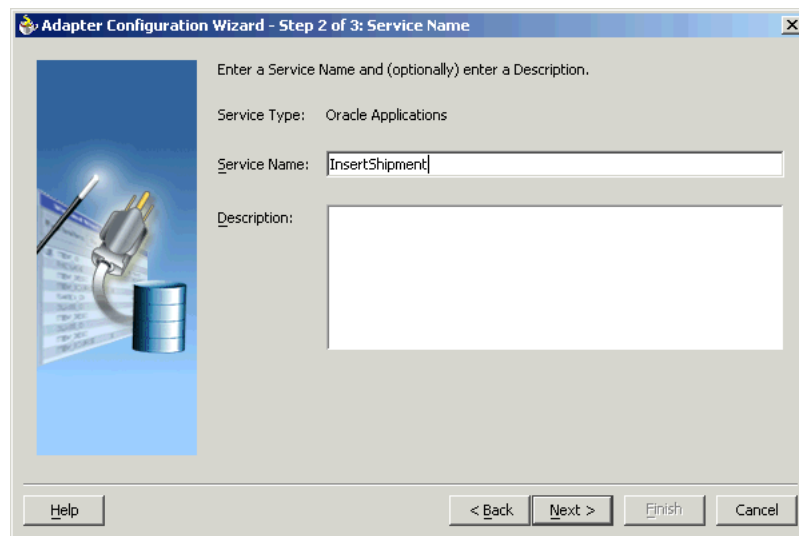
The next task is to add a partner link to the BPEL process. A partner link defines the link name, type, and the role of the BPEL process that interacts with the partner service.

Use the following steps to add a partner link:

- Drag and drop **PartnerLink**, from the Component Palette, into the border area of the process diagram. The Create Partner Link dialog box is displayed.
- Click the **Define Adapter Service** icon in WSDL Settings. The Adapter Configuration Wizard is displayed.
- Click **Next**. The Adapter Type dialog box is displayed.
- Select **Oracle Applications**, as shown in [Figure 5-4](#). Click **Next**.

Figure 5–4 Selecting OracleAS Adapter for Oracle Applications

5. The Service Name dialog box is displayed, as shown in [Figure 5–5](#). Enter the following information:
 - a. In the **Service Name** field, enter a service name.
 - b. In the **Description** field, enter a description for the service. This is an optional field. Click **Next**.

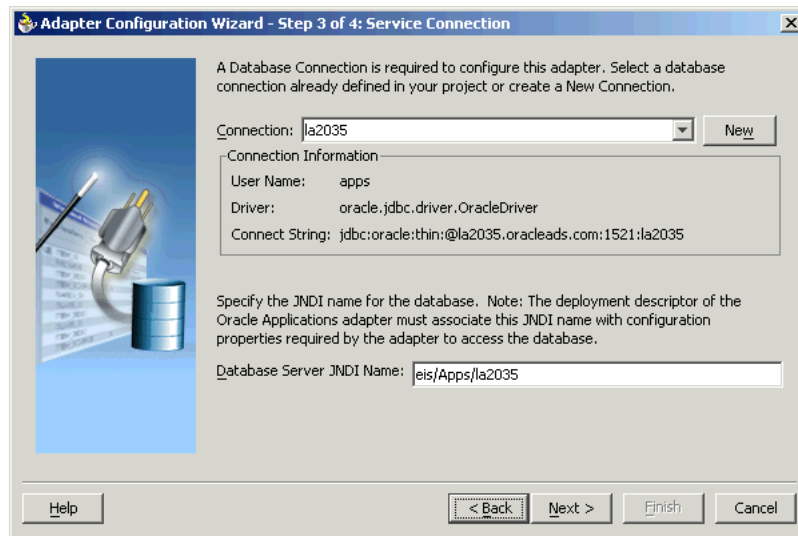
Figure 5–5 Specifying the Service Name

6. The Service Connection dialog box is displayed. Enter the Java Naming and Directory Interface (JNDI) name in the **Database Server JNDI Name** field. The JNDI name acts as a placeholder for the connection used when your service is deployed to the BPEL server. This enables you to use different databases for development and later production.

See Also: *Oracle Application Server Adapter Concepts* for understanding JNDI concepts

Figure 5–6 shows how to create a new database connection.

Figure 5–6 Creating a New Database Connection



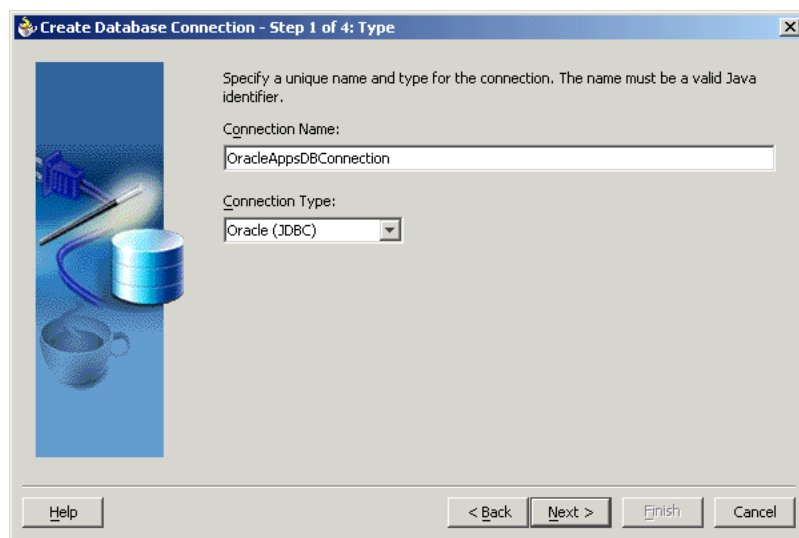
7. Click **New** to define a database connection. The Create Database Connection Wizard is displayed.

Note: You need to connect to the database where Oracle Applications is running.

8. Enter the following information in the Type dialog box:
 - a. In the **Connection Name** field, specify a unique name for the database connection.
 - b. From the **Connection Type** box, select the type of connection for your database connection.

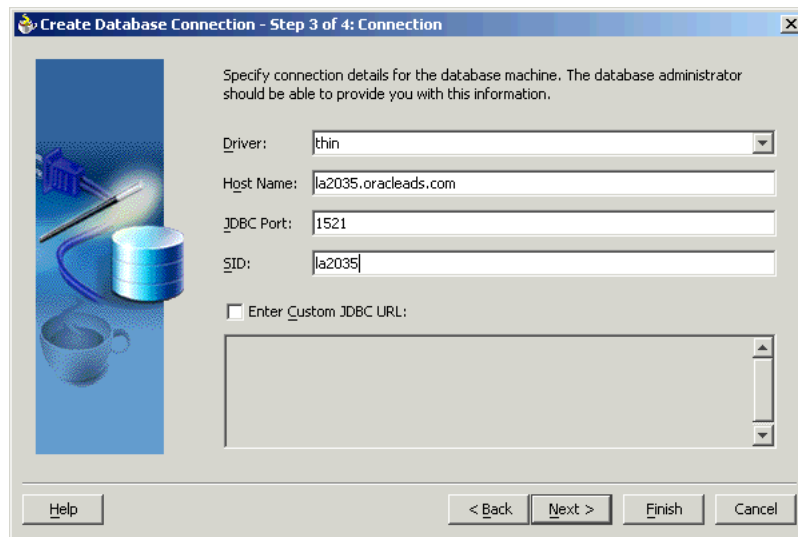
Figure 5–7 shows the Type dialog box.

Figure 5–7 Specifying the Connection Name and Type of Connection



9. Click **Next**. The Authentication dialog box is displayed.
10. Enter information in the following fields:
 - a. In the **UserName** field, specify a unique name for the database connection.
 - b. In the **Password** field, specify a password for the database connection.
11. Click **Next**. The Connection dialog box is displayed.
12. Enter information in the following fields:
 - a. From the **Driver** list, select **Thin**.
 - b. In the **Host Name** field, specify the host name for the database connection.
 - c. In the **JDBC Port** field, specify the port number for the database connection.
 - d. In the **SID** field, specify a unique SID value for the database connection.

[Figure 5–8](#) shows the Connection dialog box.

Figure 5–8 Specifying the New Database Connection Information

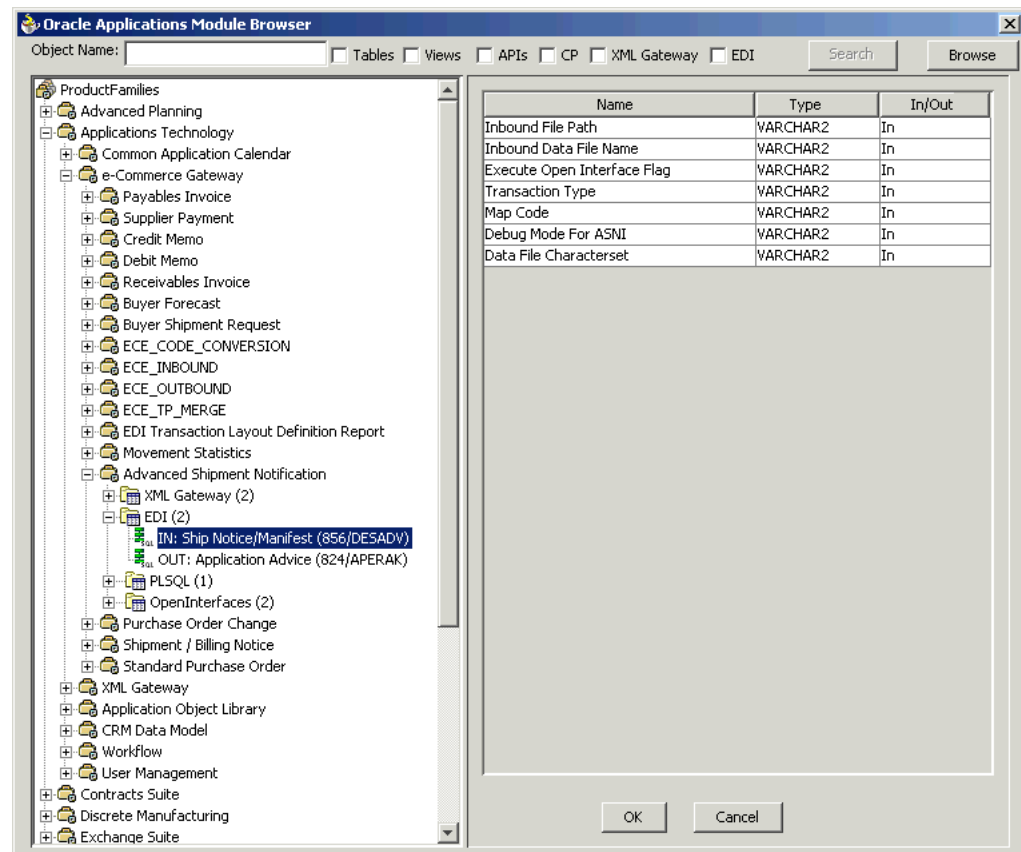
13. Click **Next**. The Test dialog is displayed.
14. Click **Test Connection** to determine whether the specified information establishes a connection with the database.
15. Click **Next**. The Service Connection dialog box is displayed, providing a summary of the database connection.
16. Click **Finish** to complete the process of creating a new database connection.

Once you have completed creating a new connection for the service, you can select an e-Commerce Gateway interface by browsing through the modules available in Oracle Applications.

1. Click **Next** in the Service Connection dialog box. The Operation dialog box is displayed.

Note: If you are connecting to a pre-11.5.10 Oracle Applications instance, then you would be required to select the interface to Oracle Applications data. Select **EDI Gateway** to proceed. Select **Inbound into Oracle Applications** or **Outbound from Oracle Applications** depending on whether data is inbound into Oracle Applications or outbound from Oracle Applications. Next, click **Get CP** to choose the EDI concurrent program from the Oracle Applications Module Browser.

2. Click **Get Object** to open the Oracle Applications Module Browser. [Figure 5–9](#) shows the Oracle Applications Module Browser.

Figure 5–9 Specifying the EDI Concurrent Program

Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. For example, Applications Technology or Order Management Suite are product families in Oracle Applications. The product families contain the individual products. For example, Applications Technology contains the e-Commerce Gateway product. The product contains the business entities associated with the product. For example, the e-Commerce Gateway product contains the Advanced Shipment Notification business entity.

Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide. EDI programs can be found under the EDI category.

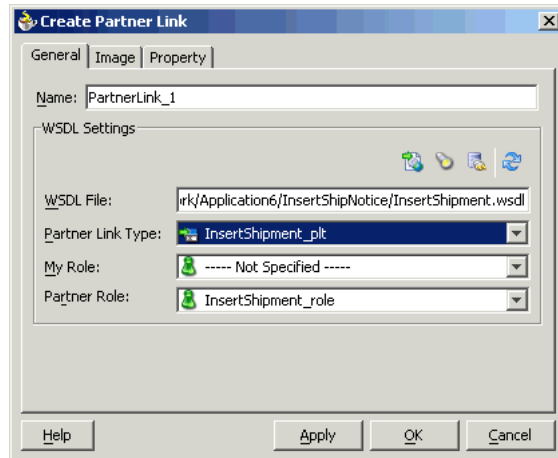
3. Select an inbound or outbound EDI program. Click **OK**. You can select only one EDI program for each adapter service.

Note: You can also search for an EDI program by entering the name of the program in the **Object Name** field. Select the **EDI** check box and click **Search**.

4. The EDI program is added to Operation Objects. Click **Next** in the Operation dialog box.
5. Click **Finish**. When you click **Finish**, the wizard generates the WSDL file corresponding to the selected interface. This WSDL file is now available for the partner link. [Figure 5–10](#) shows the Create Partner Link dialog box after the WSDL file has been generated.

Note: When you click **Finish**, two SQL files may be added to the project if a wrapper does not exist for the function. A wrapper is generated the very first time you create the e-Commerce Gateway based service. Subsequent services reuse the wrapper that is created.

Figure 5–10 *Completing the Partner Link Configuration*



6. Click **OK**. The partner link is created with the required WSDL settings.

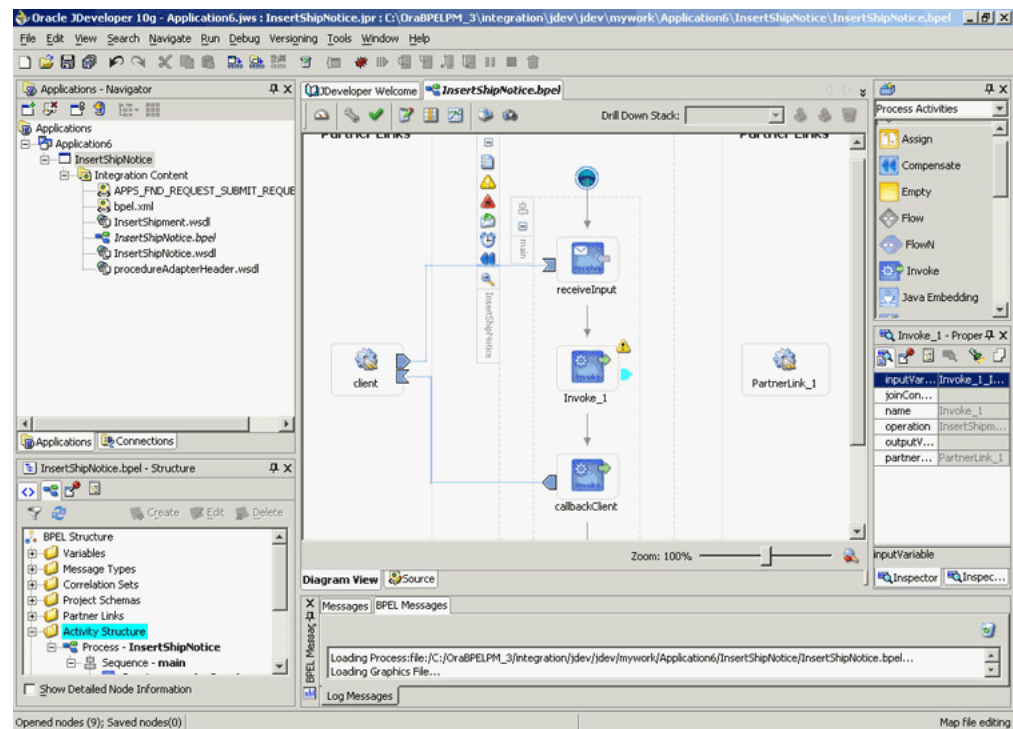
5.2.2.3 Configuring the Invoke Activity

After adding and configuring the partner link, the next task is to configure the BPEL process itself. You can start by configuring the **Invoke** process activity to invoke the EDI concurrent program.

The following steps discuss configuring the Invoke activity:

1. Drag and drop **Invoke** into the process map window. shows the **Invoke** activity after it has been added to the process map.

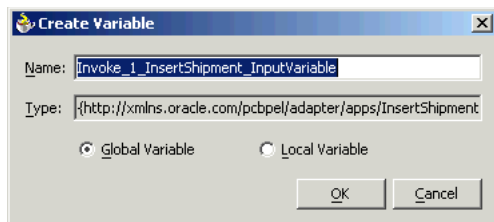
Figure 5–11 Adding the Invoke Activity



2. Double-click **Invoke** in the process map to open the Invoke dialog box. The **General** tab is selected by default. Figure 5–12 shows the Invoke dialog box.

Figure 5–12 Configuring the Invoke Activity

3. In the **Partner Link** box, select the partner link to invoke. This is the partner link that you configured in the previous section. The **Operation** is automatically selected, depending on the EDI concurrent program that you chose when configuring the partner link.
4. Click the **Create** icon next to the **Input Variable** field. Enter a descriptive name for the variable in the Create Variable dialog box that appears. You can also accept the default name. Click **OK**. Figure 5–13 shows the Create Variable dialog box.

Figure 5–13 Creating the Input Variable

5. Click the **Create** icon next to the **Output Variable** field. Enter a descriptive name for the variable in the Create Variable dialog box that appears. You can also accept the default name. Click **OK**.
6. In the Invoke dialog box, click **Apply**, and then click **OK**.

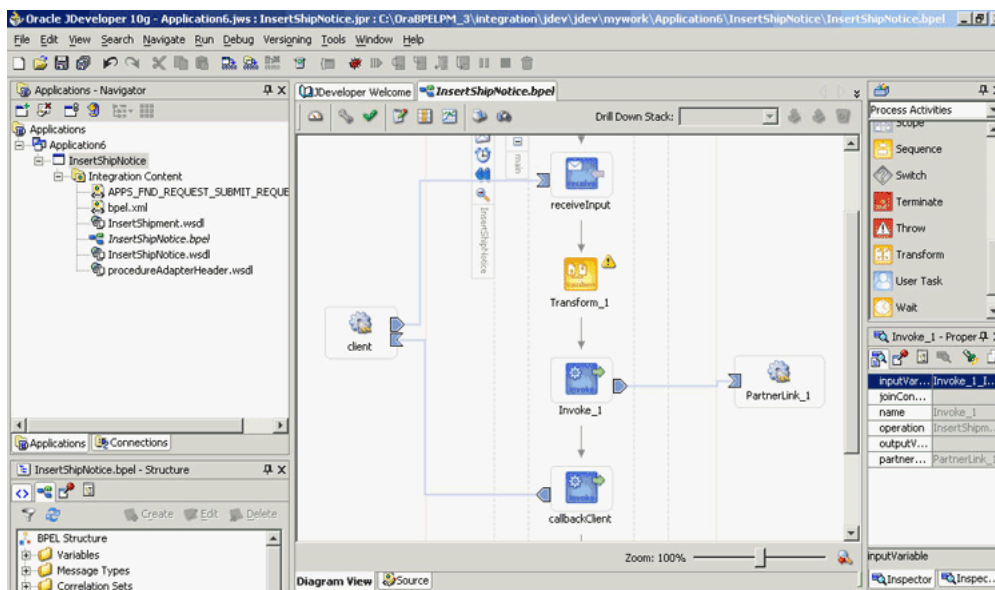
Note: You can define a **Input Header Variable** under the **Adapters** tab of the Invoke dialog box. This variable can be used to provide context information for Oracle Applications.

5.2.2.4 Configuring the Transform Activity

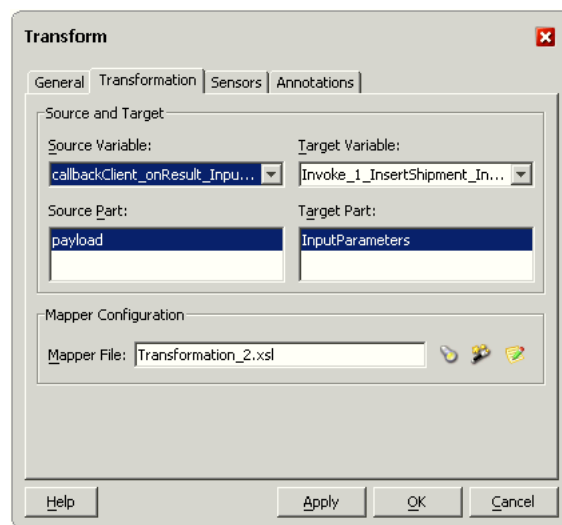
The Transform activity can be used to configure the parameters for the input and output variable. The Transform activity can also be used if variable values need to be transformed before updating them in Oracle Applications.

The following steps discuss configuring the Transform activity:

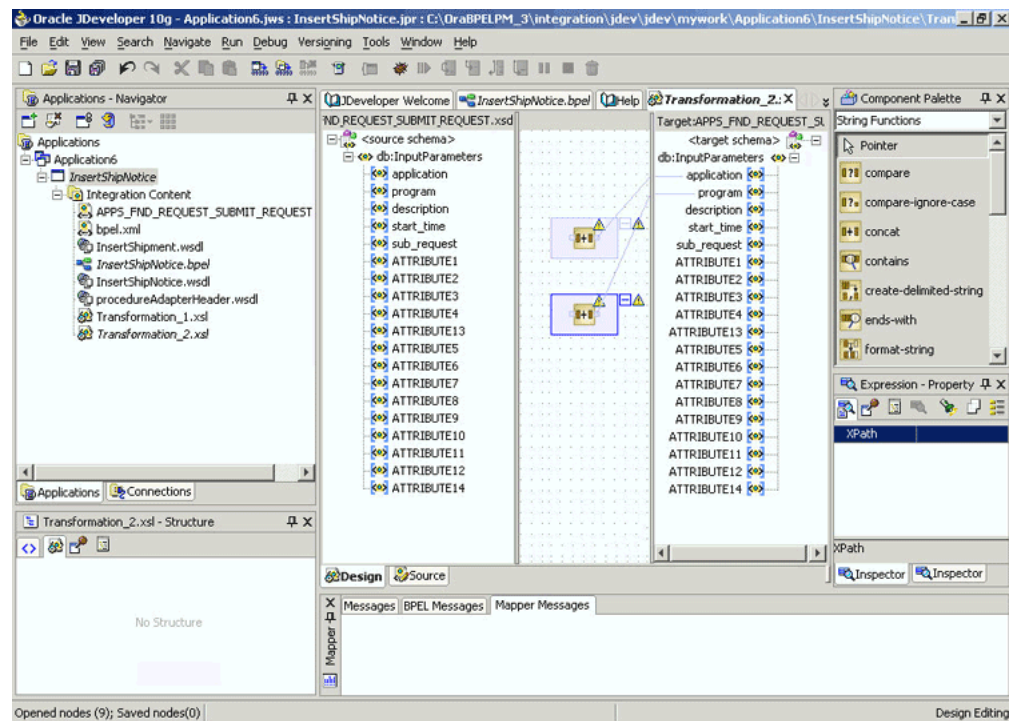
1. Drag and drop **Transform** into the process map window. The **Transform** activity should be placed in between **Receive** and **Invoke**. Figure 5–14 shows the process map window after the **Transform** activity has been added.

Figure 5–14 Adding the Transform Activity

2. Double-click **Transform** in the process map to open the Transform dialog box. The **Transformation** tab is selected by default. Figure 5–15 shows the Transform dialog box.

Figure 5–15 Configuring the Transform Activity

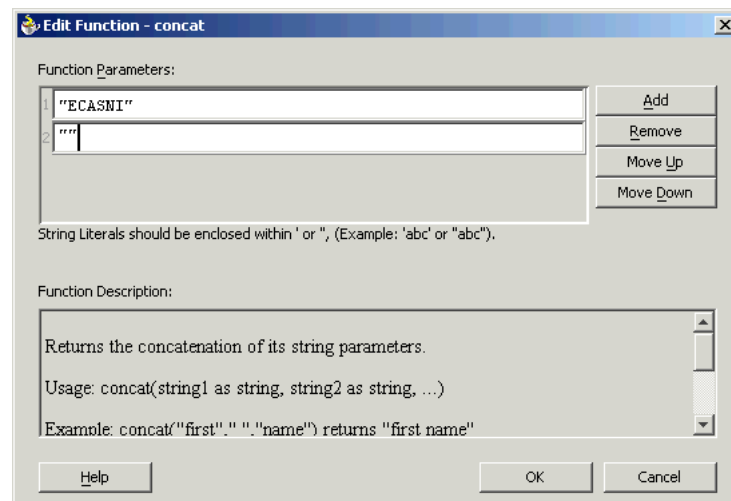
3. Select the **Source Variable** and **Target Variable** from the respective boxes. Elements are mapped from the **Source Variable** to the **Target Variable**.
4. Select the **Source Part** of the variable from which to map elements. For example, the source part may be a payload schema consisting of a purchase order request.
5. Select the **Target Part** of the variable to which to map elements. For example, the target part may be a payload schema consisting of a purchase order acknowledgment.
6. **Mapper File** specifies the file in which you create the mappings using the XSLT Mapper Transformation tool. Click the **Create** icon next to the **Mapper File** field to create a new transformation mapping file.
7. The transformation mapping file is displayed. The **Design** view is displayed by default. [Figure 5–16](#) shows the **Design** view for the transformation mapping file.

Figure 5–16 Editing the Transformation Mapping File

8. You can define the parameter values in the **Design** view. Drag a string function to the **Design** area. Connect the function to the appropriate parameter for which you want to define a value.

Note: You can use an input parameter value from the source variable, transform it using a string function, and use it as the input parameter value for the target variable.

9. Double-click the icon for the function. The Edit Function dialog box is displayed. [Figure 5–17](#) shows the Edit Function dialog box.

Figure 5–17 Supplying the Function Parameters

10. Repeat steps 8 and 9 for all the parameters that you need to supply.

Note: You can use both the Transform and Assign activities to configure the variables. If the number of mappings to be configured is less, say three or four, then you can use the Assign activity. The Transform activity is more suitable for situations where the number of mappings is large or where variable transformation is required.

5.3 Run-Time Steps

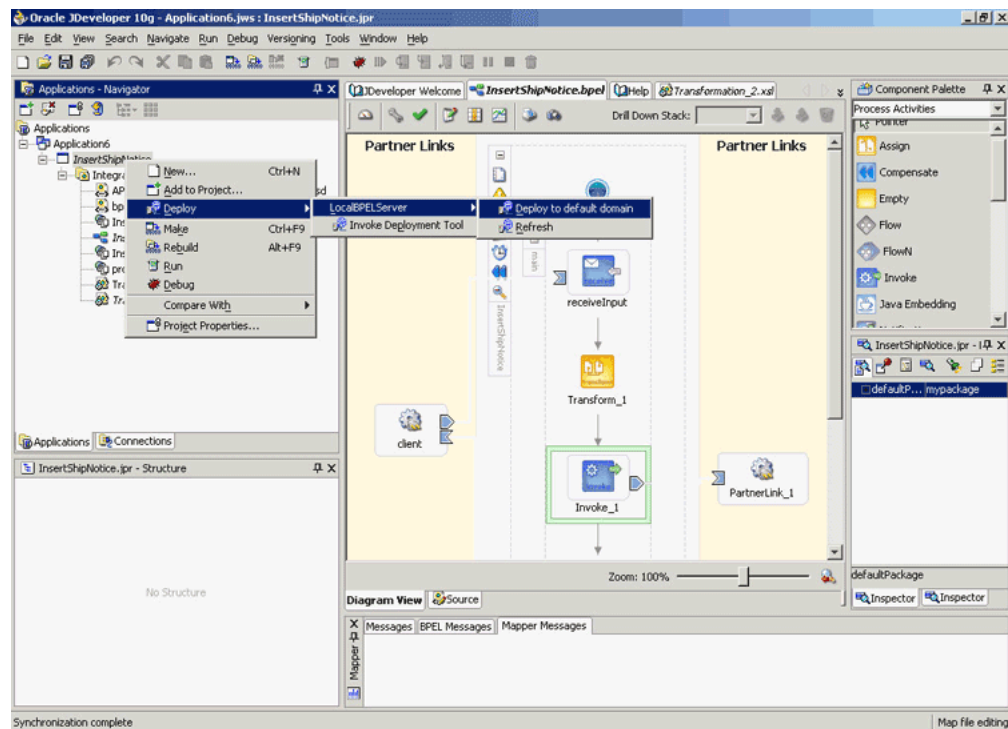
After designing the BPEL process, the next step is to deploy, run and monitor it. This section discusses the following:

- [Deploying the BPEL Process](#)
- [Testing the BPEL Process](#)
- [Verifying Records in Oracle Applications](#)

5.3.1 Deploying the BPEL Process

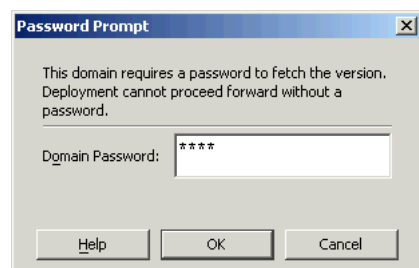
You need to deploy the BPEL process before you can run it. The BPEL process is first compiled and then deployed to the BPEL server. The following steps discuss deploying the BPEL process to a BPEL server:

1. Select the BPEL project in the Applications window.
2. Right-click the project name, and then select **Deploy** from the menu that appears.
3. Select **Local BPEL Server** followed by **Deploy to Default Domain**, if you are deploying the process on the local BPEL server. [Figure 5-18](#) illustrates deploying a BPEL process to a local BPEL server.

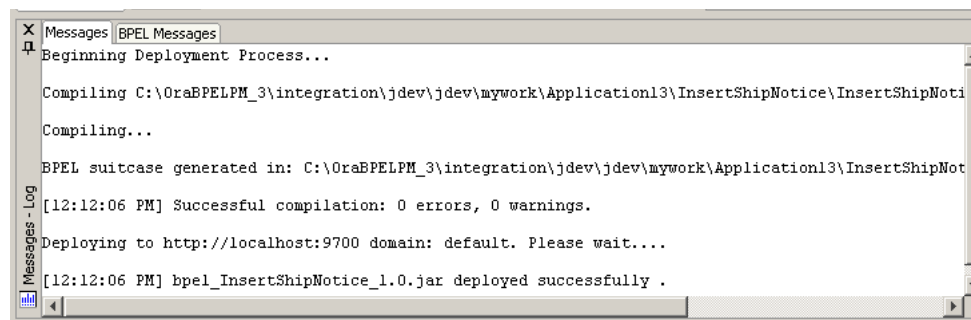
Figure 5–18 Deploying the BPEL Process

Note: You can select **Invoke Deployment Tool** if you want to deploy to a different BPEL server.

4. The Password Prompt dialog box appears. Enter the password for the default domain in the **Domain Password** field. Click **OK**. [Figure 5–19](#) shows the Password Prompt dialog box.

Figure 5–19 Specifying the Domain Password

5. The BPEL process is compiled and deployed. You can check the progress in the Messages window. [Figure 5–20](#) shows the Messages window.

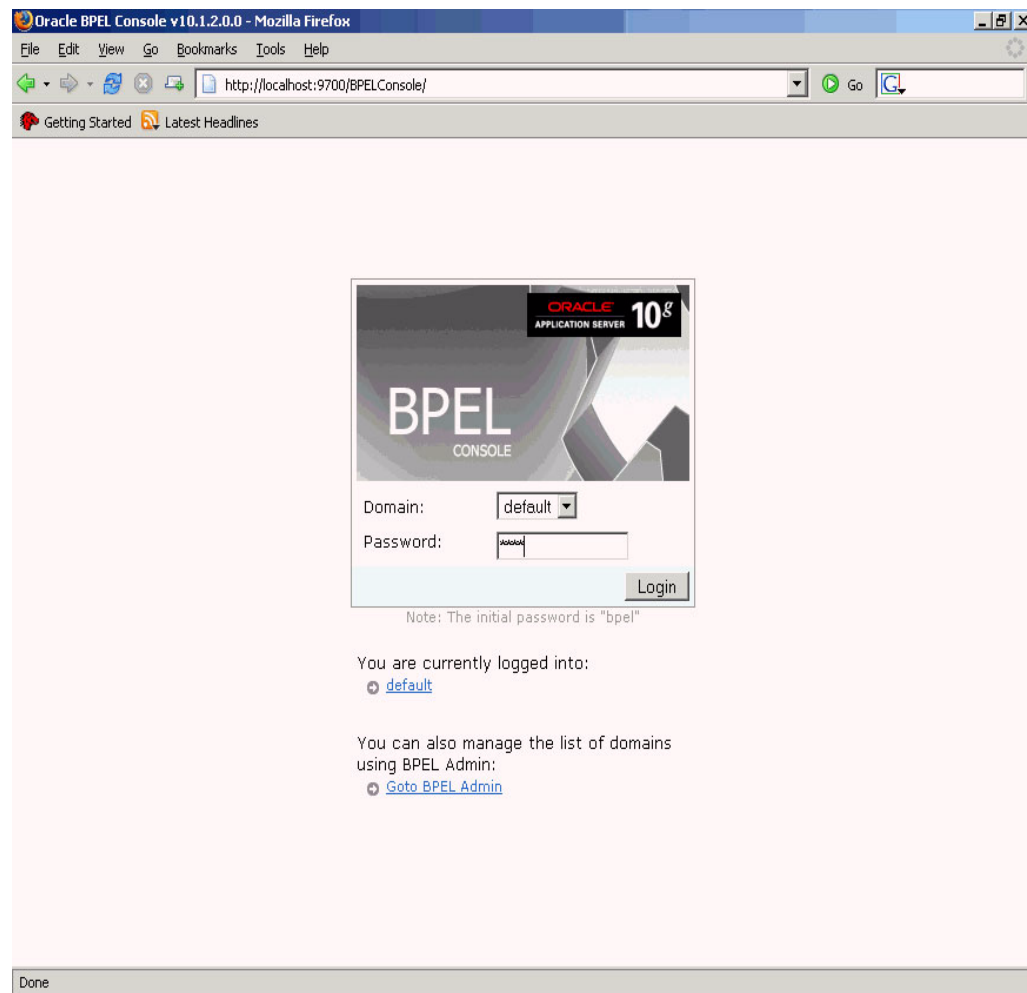
Figure 5–20 Messages Window

5.3.2 Testing the BPEL Process

Once the BPEL process is deployed, it can be seen in the BPEL Console. You can manage and monitor the process from the BPEL Console. You can also test the process and the integration interface by manually initiating the process. The following steps discuss manually initiating and monitoring the BPEL process:

1. To open the BPEL Console, click **Start**, and choose **Programs**. In the Programs menu, select **Oracle - ORACLE_HOME, Oracle BPEL Process Manager 10.1.2**, and then select **BPEL Console**.
2. The BPEL Console login screen is displayed. Select **Default** in the **Domain** box. Enter the password for the default domain in the **Password** field. Click **Login**.

[Figure 5–21](#) shows the BPEL Console login screen.

Figure 5–21 BPEL Console Login Screen

3. Oracle BPEL Console is displayed. The list of deployed processes is shown under Deployed BPEL Processes. [Figure 5–22](#) shows the BPEL Console screen.

Figure 5–22 Deployed BPEL Processes

The screenshot shows the Oracle BPEL Console v10.1.2.0.0 interface in a Mozilla Firefox browser. The console has a navigation bar with tabs: Dashboard, BPEL Processes, Instances, and Activities. The BPEL Processes tab is active, displaying a table of deployed BPEL processes and their instances.

Deployed BPEL Processes		In-Flight BPEL Process Instances	
Name	Instance	BPEL Process	Last Modified ↑
InsertPurchaseOrder			
InsertShipNotice			
TaskActionHandler			
TaskManager			
Recently Completed BPEL Process Instances (More...)			
	✓ 9 : Instance #9 of InsertPurchaseOrder	InsertPurchaseOrder (v. 1.0)	11/3/05 5:16:08 PM
	✓ 8 : Instance #8 of InsertPurchaseOrder	InsertPurchaseOrder (v. 1.0)	11/3/05 12:32:59 PM
	✓ 7 : Instance #7 of InsertPurchaseOrder	InsertPurchaseOrder (v. 1.0)	11/2/05 3:17:24 PM

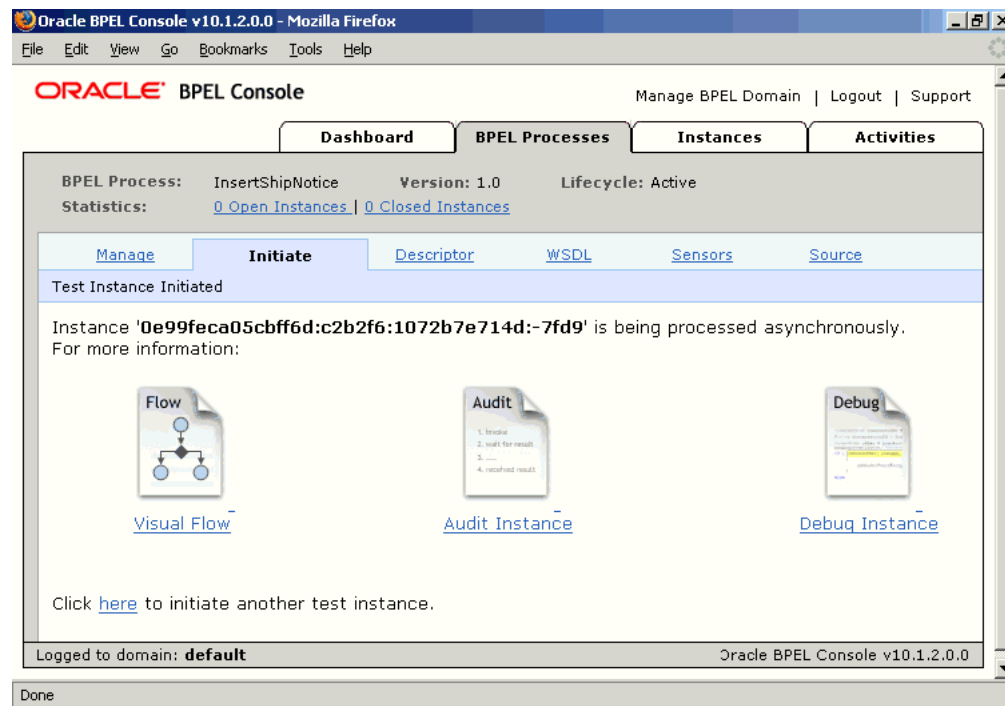
Deployed BPEL Processes: InsertPurchaseOrder, InsertShipNotice, TaskActionHandler, TaskManager

Deploy New Process

Logged to domain: **default** Oracle BPEL Console v10.1.2.0.0

http://localhost:9700/BPELConsole/default/displayProcess.jsp?processId=InsertShipNotice&revisionTag=1.0

4. Click the BPEL process that you want to initiate. The Initiate page is displayed. Enter the input string required by the process.
5. Click **Post XML Message** to initiate the process.
6. The BPEL process is now initiated. You can check the process flow by clicking the **Visual Flow** icon. Figure 5–23 shows the BPEL Console Initiate page.

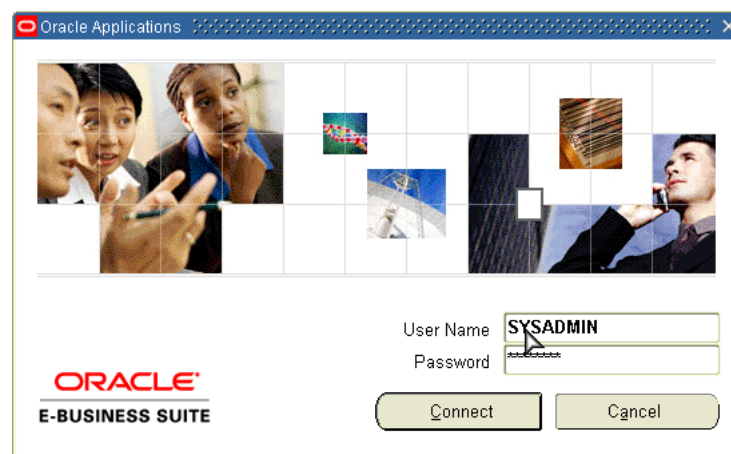
Figure 5–23 BPEL Console Initiate Page

7. The audit trail provides information on the steps that have been executed. The audit trail also records the Request ID that is returned for the transaction. You can check the audit trail by clicking the **Audit Instance** icon.

5.3.3 Verifying Records in Oracle Applications

You can use the following steps to track the process in Oracle Applications:

1. Log in to Oracle Applications as the System Administrator. [Figure 5–24](#) shows the Oracle Applications login screen.

Figure 5–24 Oracle Applications Login Screen

2. Select **Requests** from the **View** menu.

3. Search for the Request by entering the Request Id that you got from the audit trail, and then click **Find**. Figure 5–25 shows the Find Requests dialog box.

Figure 5–25 Find Requests Dialog Box

4. The Request details are displayed. You can check for details such as the Phase and Status of the request.
5. If the Status of the Request is complete, then you can also query the appropriate table in Oracle Applications to search for the relevant records that have been inserted. Figure 5–26 shows a simple SQL query on an Oracle Applications table.

Figure 5–26 Querying Oracle Applications for a Record

```
SQL> select      distinct(shipment_num) from    rcv_headers_interface where s
shipment_num
=          'S1722427';

SHIPMENT_NUM
-----
S1722427

SQL> █
```

Using XML Gateway

Oracle XML Gateway provides a common, standards-based approach for XML integration between Oracle Applications and third party applications, both inside and outside your enterprise. XML is key to an integration solution, as it standardizes the way in which data is searched, exchanged, and presented thereby enabling interoperability throughout the supply chain.

This chapter includes the following sections:

- [Overview of XML Gateway](#)
- [Design-Time Steps for XML Gateway Inbound into Oracle Applications](#)
- [Run-Time Steps for XML Gateway Inbound into Oracle Applications](#)
- [Design-Time Steps for XML Gateway Outbound from Oracle Applications](#)
- [Run-Time Steps for XML Gateway Outbound from Oracle Applications](#)

6.1 Overview of XML Gateway

Oracle XML Gateway is a set of services that allows easy integration with Oracle Applications to support XML messaging. Oracle Applications utilize the Oracle Workflow Business Event System to support event-based XML message creation and consumption.

OracleAS Adapter for Applications can be configured to use XML Gateway to interact with third party applications. The tight integration provided by open interface tables is not suitable for those scenarios where trading partners change frequently. XML Gateway is an ideal solution when you need to interact with third party applications that use open standards. Moreover, it is also suitable for scenarios where trading partners change frequently.

6.1.1 Standards-Based Messaging

As a provider of broad based business application solutions to support all industries, Oracle XML Gateway supports all Document Type Definition (DTD) based XML standards. The majority of the Oracle prebuilt messages delivered with Oracle Applications are premapped using the Open Application Group (OAG) standard. Any Oracle prebuilt message map may be remapped to your standard of choice using the XML Gateway Message Designer.

6.1.2 Integration Architecture

XML Gateway provides an application integration infrastructure that is flexible enough to accommodate the integration requirements of any application that needs to

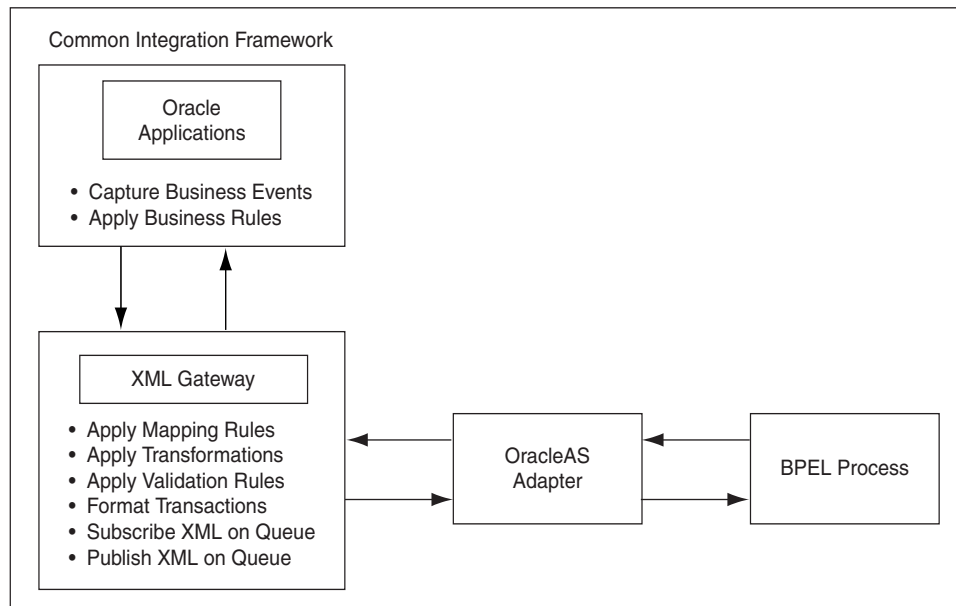
integrate with Oracle Applications. XML Gateway enables you to create an efficient and responsive supply chain that links all customers, factories, warehouses, distributors, carriers, and other trading partners. All these entities can seamlessly operate as a single enterprise.

Oracle XML Gateway supports both Business-to-Business (B2B) and Application-to-Application (A2A) initiatives. B2B initiatives include communicating business documents and participating in industry exchanges. An example of an A2A initiative is data integration with legacy and disparate systems.

XML Gateway enables bidirectional integration with Oracle Applications by allowing you to insert and retrieve data from Oracle Applications. The Oracle Applications adapter for XML Gateway supports two operations, namely, enqueue and dequeue. The enqueue operation is used to put messages in the queue. This is used to insert inbound data into Oracle Applications. The dequeue operation polls for messages from the queue. This is used to retrieve outbound data from Oracle Applications. Only one operation can be defined for each adapter service.

Figure 6–1 shows the integration architecture for XML Gateway.

Figure 6–1 XML Gateway Integration Architecture



6.1.3 Message Queues

The XML Gateway uses queues specifically at two points in the process as well as employing a general error queue. The first point is at the transport agent level between the transport agent module and the XML Gateway. The second point is at the transaction level between base Oracle Applications products and the XML Gateway.

6.1.3.1 Inbound Queues

Inbound message queues are used for XML messages inbound into Oracle Applications. Inbound message queues are positioned between the Transport Agent and the Oracle Workflow Business Event System.

The messages must be formatted according to the XML Gateway envelope message format. The envelope message format is discussed under [XML Gateway Envelope](#).

Oracle Workflow Business Event System copies the inbound messages to the proper inbound Transaction Queue.

6.1.3.2 Outbound Queues

Outbound message queues are used for XML messages outbound from Oracle Applications. The outbound Message Queue is positioned between the XML Gateway and the Transport Agent.

The XML Gateway creates XML messages, then enqueues them on this queue. The Transport Agent dequeues the message and delivers it to the Trading Partner.

6.1.4 XML Gateway Envelope

In addition to the business document such as a purchase order or invoice in the XML Payload, a set of message attributes are also transmitted. Collectively, these attributes are called the XML Gateway envelope. This subsection discusses some of these attributes.

MESSAGE_TYPE

Payload message format. This defaults to XML. Oracle XML Gateway currently supports only XML.

MESSAGE_STANDARD

Message format standard as displayed in the Define Transactions form and entered in the Define XML Standards form. This defaults to OAG. The message standard entered for an inbound XML document must be the same as the message standard in the trading partner setup.

TRANSACTION_TYPE

External Transaction Type for the business document from the Trading Partner table. The transaction type for an inbound XML document must be the same as the transaction type defined in the Trading Partner form.

TRANSACTION_SUBTYPE

External Transaction Subtype for the business document from the Trading Partner table. The transaction subtype for an inbound XML document must be the same as the transaction subtype defined in the Trading Partner form.

DOCUMENT_NUMBER

The document identifier used to identify the transaction, such as a purchase order or invoice number. This field is not used by the XML Gateway, but it may be passed on inbound messages.

SOURCE_TP_LOCATION_CODE

This is the Source Trading Partner Location Code.

PROTOCOL_TYPE

Transmission Protocol as defined in the Trading Partner table.

PROTOCOL_ADDRESS

Transmission address as defined in the Trading Partner table.

USERNAME

USERNAME as defined in the Trading Partner table.

PASSWORD

The password associated with the USERNAME defined in the Trading Partner table.

PARTY_SITE_ID

The party site identifier for an inbound XML document must be the same as the Source Trading Partner location defined in the Trading Partner form.

ATTRIBUTE3

For outbound messages, this field has the value from the Destination Trading Partner Location Code in the Trading Partner table. For inbound messages, the presence of this value generates another XML message that is sent to the trading partner identified in the Destination Trading Partner Location Code in the Trading Partner table. This value must be recognized by the hub to forward the XML message to the final recipient of the XML Message.

See Also: *Destination Trading Partner Location Code* section on page 199 of the *Oracle XML Gateway User's Guide*. This guide is a part of the Oracle Applications documentation library. Oracle Applications documentation can be accessed from the following link:

<http://www.oracle.com/technology/documentation/applications.html>

PAYLOAD

The XML message.

Parameters defined by the Application

The following parameters may be defined by the base application:

- ATTRIBUTE1
- ATTRIBUTE2
- ATTRIBUTE4
- ATTRIBUTE5

Parameters Not Used

The following parameters are not used:

- PARTYID
- PARTYTYPE

See Also: *Oracle XML Gateway User's Guide* for details on the XML Gateway Execution Engine, Trading Partner validation, and so on. This guide is a part of the Oracle Applications documentation library. Oracle Applications documentation can be accessed from the following link:

<http://www.oracle.com/technology/documentation/applications.html>

6.2 Design-Time Steps for XML Gateway Inbound into Oracle Applications

OracleAS adapter for Oracle Applications is deployed using the BPEL Process Manager (PM) in Oracle JDeveloper. The BPEL PM creates the WSDL interfaces for the XML Gateway message map.

This section describes configuring the OracleAS Adapter for Oracle Applications to use XML Gateway. It includes the following topics:

- [Prerequisites to Configuring OracleAS Adapter for Oracle Applications](#)
- [Configuring OracleAS Adapter for Oracle Applications](#)

6.2.1 Prerequisites to Configuring OracleAS Adapter for Oracle Applications

You need to populate certain variables in the BPEL PM in order to provide context information for Oracle Applications. The `MESSAGE_TYPE`, `MESSAGE_STANDARD`, `TRANSACTION_TYPE`, `TRANSACTION_SUBTYPE`, and `PARTY_SITE_ID` are the mandatory header variables that you need to populate for the XML transaction to complete successfully. Refer to [Section 6.2.2.6, "Configuring the Assign Activity"](#) for more information.

You also need to configure and schedule two listeners on the Oracle Applications side. These are the ECX Inbound Agent Listener and the ECX Transaction Agent Listener. Use the following steps to configure these listeners in Oracle Applications:

1. Log in to Oracle Applications with the responsibility of Workflow Administrator.
2. The Navigator page is displayed. Click the **Workflow Administrator Web Applications** link.
3. Click the **Workflow Manager** link under Oracle Applications Manager.
4. Click the status icon next to **Agent Listeners**.
5. Configure and schedule the **ECX Inbound Agent Listener** and the **ECX Transaction Agent Listener**. Select the listener, and select Start from the **Actions** box. Click **Go**.

6.2.2 Configuring OracleAS Adapter for Oracle Applications

This section describes the tasks required to configure OracleAS Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper. This section covers the following:

- [Creating a New BPEL Project](#)
- [Creating a Partner Link](#)
- [Configuring the Invoke Activity](#)
- [Adding a Partner Link for the File Adapter](#)
- [Configuring the Receive Activity](#)
- [Configuring the Assign Activity](#)

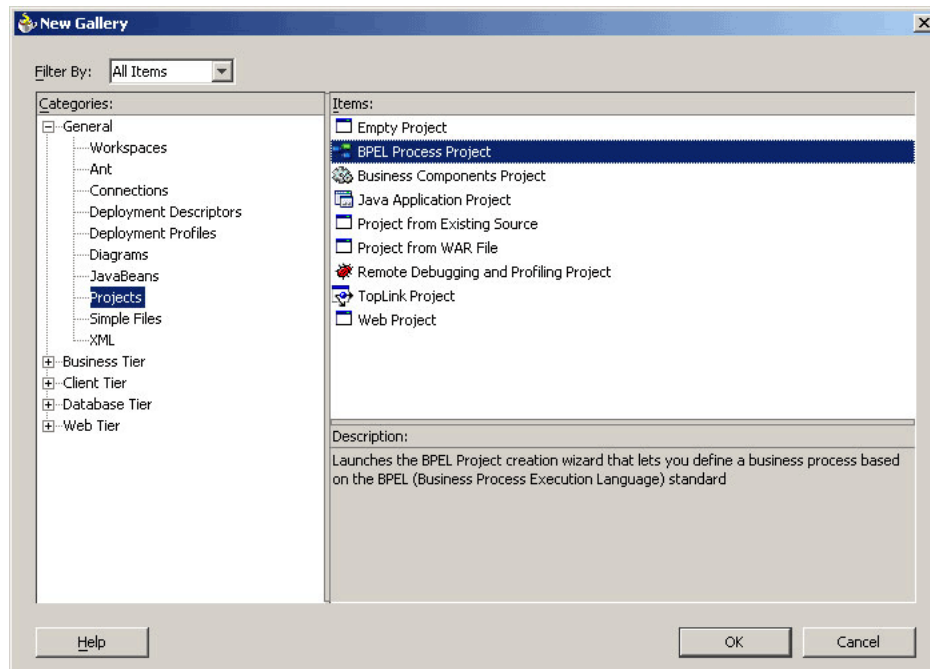
6.2.2.1 Creating a New BPEL Project

Use the following steps to create a new BPEL project:

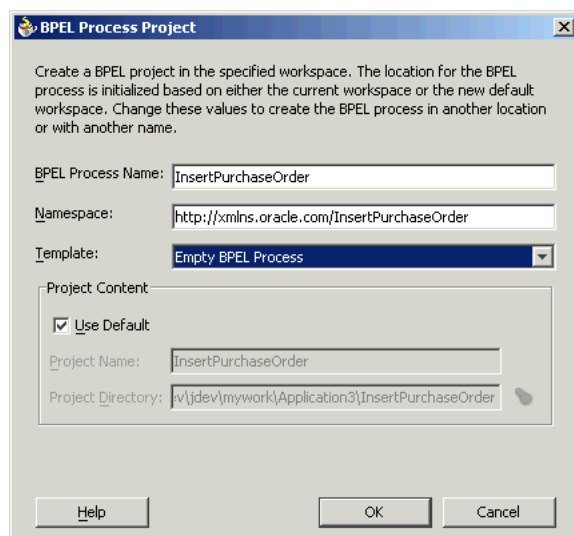
1. Open JDeveloper BPEL Designer.

2. From the **File** menu, select **New**. The New Gallery dialog box is displayed.
3. Select **All Items** from the **Filter By** box. This displays a list of available categories.
4. Expand the **General** node, and then select **Projects**.
5. Select **BPEL Process Project** from the **Items** group, as shown in [Figure 6-2](#).

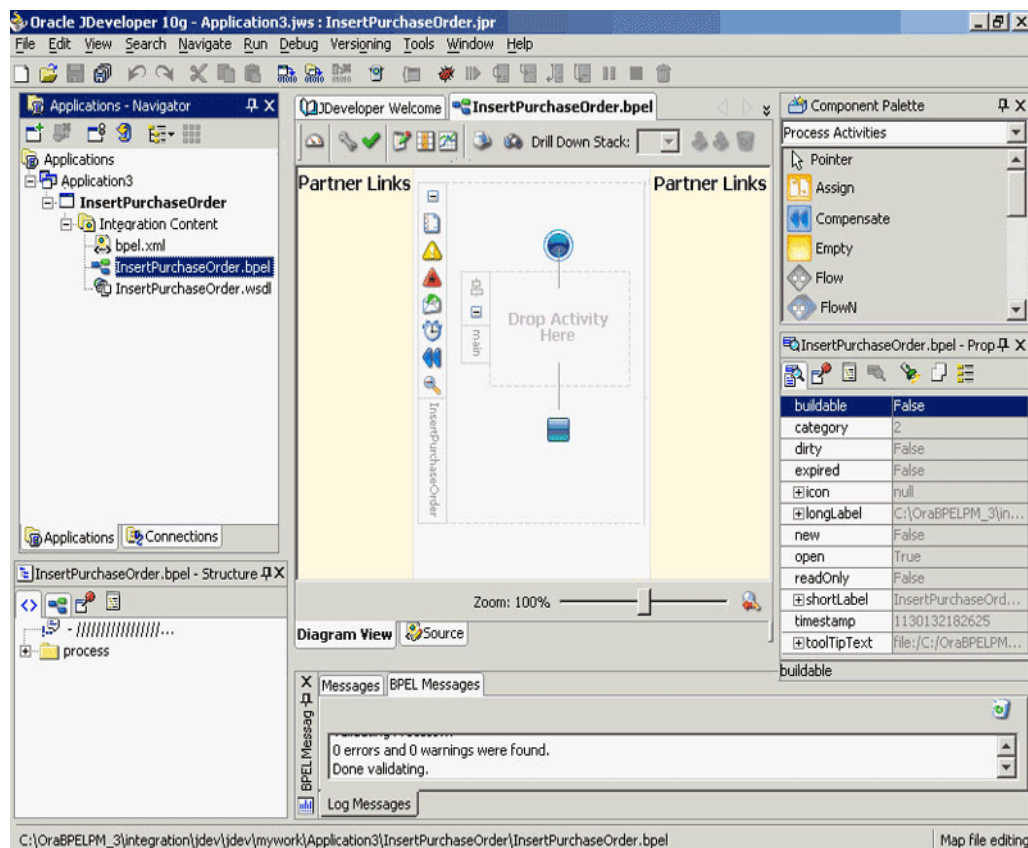
Figure 6-2 Creating a New BPEL Process Project



6. Click **OK**. The BPEL Process Project dialog box is displayed.
7. In the **BPEL Process Name** field, enter a descriptive name. For example, `InsertPurchaseOrder`.
8. From the **Template** box, select **Empty BPEL Process**. Keep the default selection for **Use Default** under Project Content, as shown in [Figure 6-3](#).

Figure 6–3 Specifying a Name for the New BPEL Process Project

9. Click **OK**. A new BPEL process, with the required source files including `bpel.xml`, `InsertPurchaseOrder.bpel`, and `InsertPurchaseOrder.wsdl`, is created as shown in [Figure 6–4](#).

Figure 6–4 New BPEL Process Project

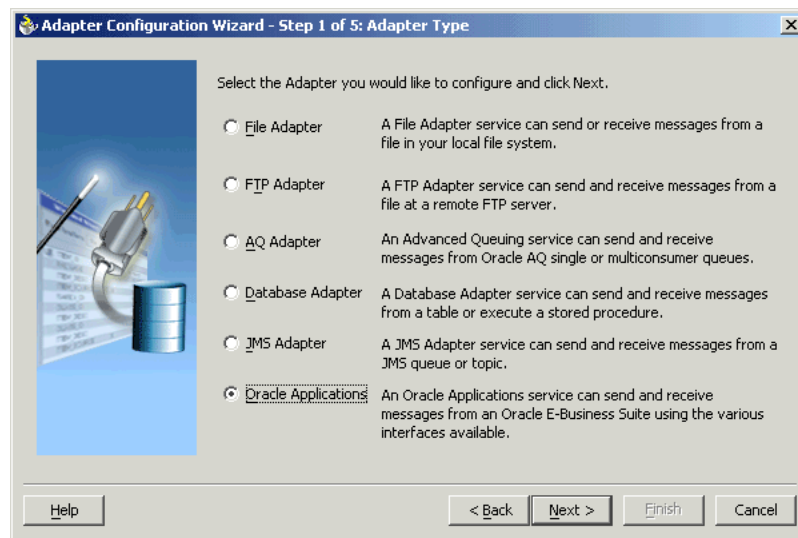
6.2.2.2 Creating a Partner Link

The next task is to add a partner link to the BPEL process. A partner link defines the link name, type, and the role of the BPEL process that interacts with the partner service.

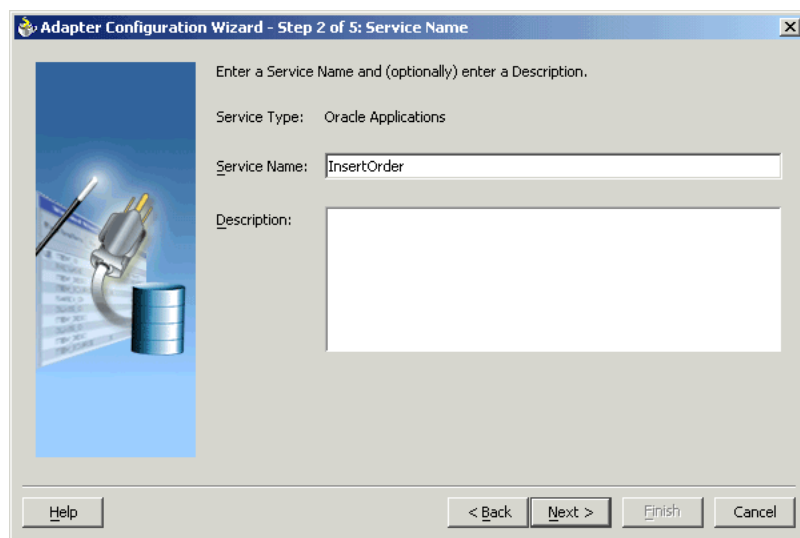
Use the following steps to add a partner link:

1. Drag and drop **PartnerLink**, from the Component Palette, into the border area of the process diagram. The Create Partner Link dialog box is displayed.
2. Click the **Define Adapter Service** icon in WSDL Settings. The Adapter Configuration Wizard is displayed.
3. Click **Next**. The Adapter Type dialog box is displayed.
4. Select **Oracle Applications**, as shown in [Figure 6-5](#). Click **Next**.

Figure 6-5 Selecting OracleAS Adapter for Oracle Applications



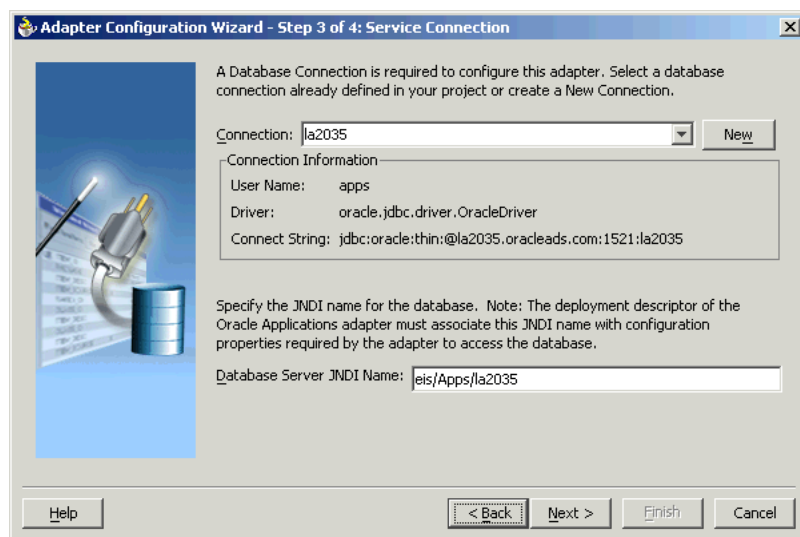
5. The Service Name dialog box is displayed, as shown in [Figure 6-6](#). Enter the following information:
 - a. In the **Service Name** field, enter a service name.
 - b. In the **Description** field, enter a description for the service. This is an optional field. Click **Next**.

Figure 6–6 Specifying the Service Name

6. The Service Connection dialog box is displayed. Enter the Java Naming and Directory Interface (JNDI) name in the **Database Server JNDI Name** field. The JNDI name acts as a placeholder for the connection used when your service is deployed to the BPEL server. This enables you to use different databases for development and later production.

See Also: *Oracle Application Server Adapter Concepts* for understanding JNDI concepts

Figure 6–7 shows how to create a new database connection.

Figure 6–7 Creating a New Database Connection

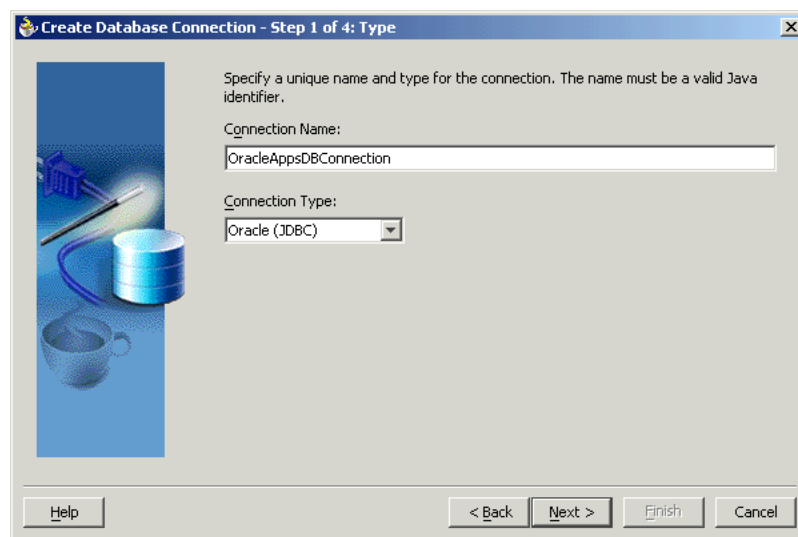
7. Click **New** to define a database connection. The Create Database Connection Wizard is displayed.

Note: You need to connect to the database where Oracle Applications is running.

8. Enter the following information in the Type dialog box:
 - a. In the **Connection Name** field, specify a unique name for the database connection.
 - b. From the **Connection Type** box, select the type of connection for your database connection.

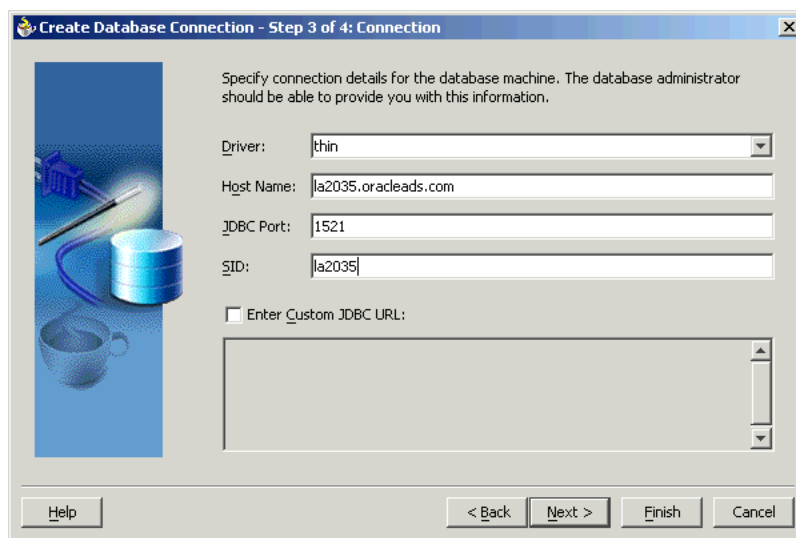
Figure 6–8 shows the Type dialog box.

Figure 6–8 Specifying the Connection Name and Type of Connection



9. Click **Next**. The Authentication dialog box is displayed.
10. Enter information in the following fields:
 - a. In the **UserName** field, specify a unique name for the database connection.
 - b. In the **Password** field, specify a password for the database connection.
11. Click **Next**. The Connection dialog box is displayed.
12. Enter information in the following fields:
 - a. From the **Driver** list, select **Thin**.
 - b. In the **Host Name** field, specify the host name for the database connection.
 - c. In the **JDBC Port** field, specify the port number for the database connection.
 - d. In the **SID** field, specify a unique SID value for the database connection.

Figure 6–9 shows the Connection dialog box.

Figure 6–9 Specifying the New Database Connection Information

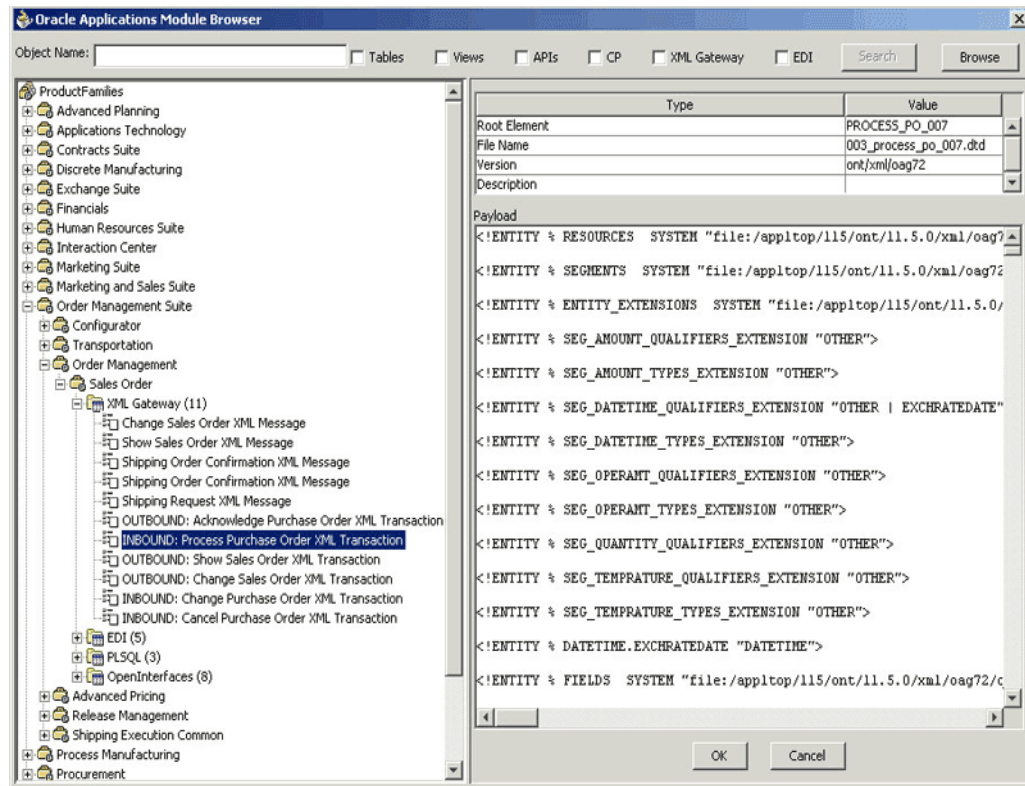
13. Click **Next**. The Test dialog is displayed.
14. Click **Test Connection** to determine whether the specified information establishes a connection with the database.
15. Click **Next**. The Service Connection dialog box is displayed, providing a summary of the database connection.
16. Click **Finish** to complete the process of creating a new database connection.

Once you have completed creating a new connection for the service, you can add an XML Gateway map by browsing through the maps available in Oracle Applications.

1. Click **Next** in the Service Connection dialog box. The Operation dialog box is displayed.

Note: If you are connecting to a pre-11.5.10 Oracle Applications instance, then you would be required to select the interface to Oracle Applications data. Select **XML Gateway** to proceed. Select **Enqueue** or **Dequeue** depending on whether data is inbound into Oracle Applications or outbound from Oracle Applications. Next, choose the DTD file that is converted into an XSD schema file.

2. Click **Get Object** to open the Oracle Applications Module Browser. [Figure 6–10](#) shows the Oracle Applications Module Browser.

Figure 6–10 Specifying the XML Gateway Message Map

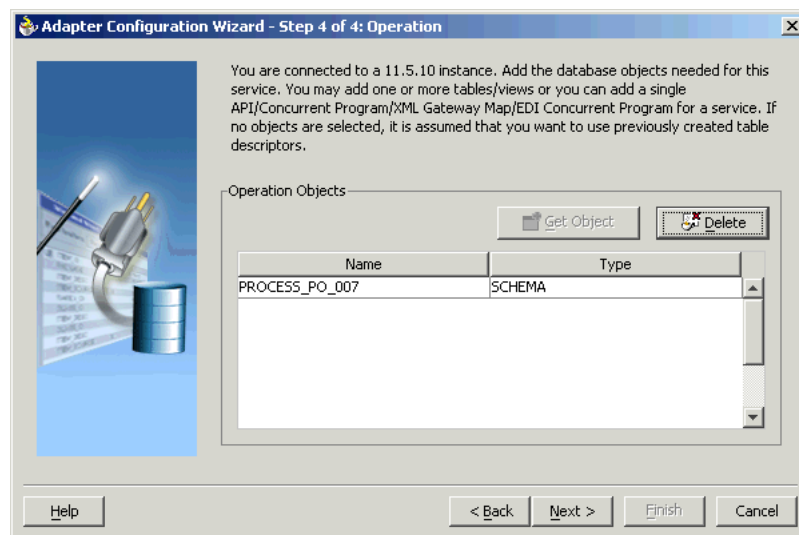
Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. For example, the Marketing Suite or the Order Management Suite are product families in Oracle Applications. The product families contain the individual products. For example, the Order Management Suite contains the Order Management product. The product contains the business entities associated with the product. For example, the Order Management product contains the Sales Order business entity.

Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide. XML Gateway message maps can be found under the XML Gateway category.

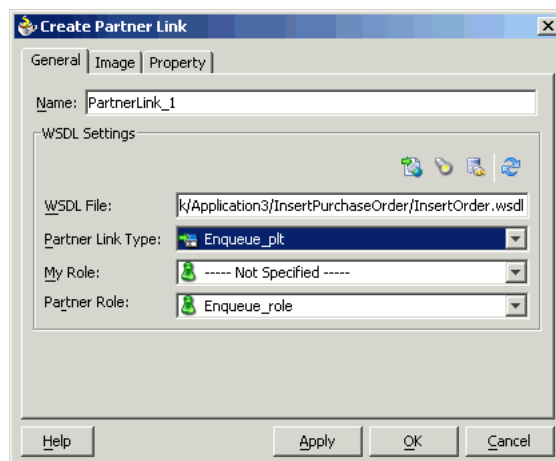
3. Select an inbound or outbound XML Gateway message map. Click **OK**. You can select only one XML Gateway message map for each adapter service. The XML schema is generated. [Figure 6–11](#) shows that the XML Gateway map that has been added.

Note:

- You can also search for an XML Gateway message map by entering the name or part of the name for the message map in the **Object Name** field. Select the **XML Gateway** check box and click **Search**.
 - The custom message maps that you might have saved can be found in the **Others** category.
-

Figure 6–11 Adding the XML Schema

4. Click **Next**.
5. Click **Finish**. When you click **Finish**, the wizard generates the WSDL file corresponding to the XML schema. This WSDL file is now available for the partner link. [Figure 6–12](#) shows the Create partner Link dialog box.

Figure 6–12 Completing the Partner Link Configuration

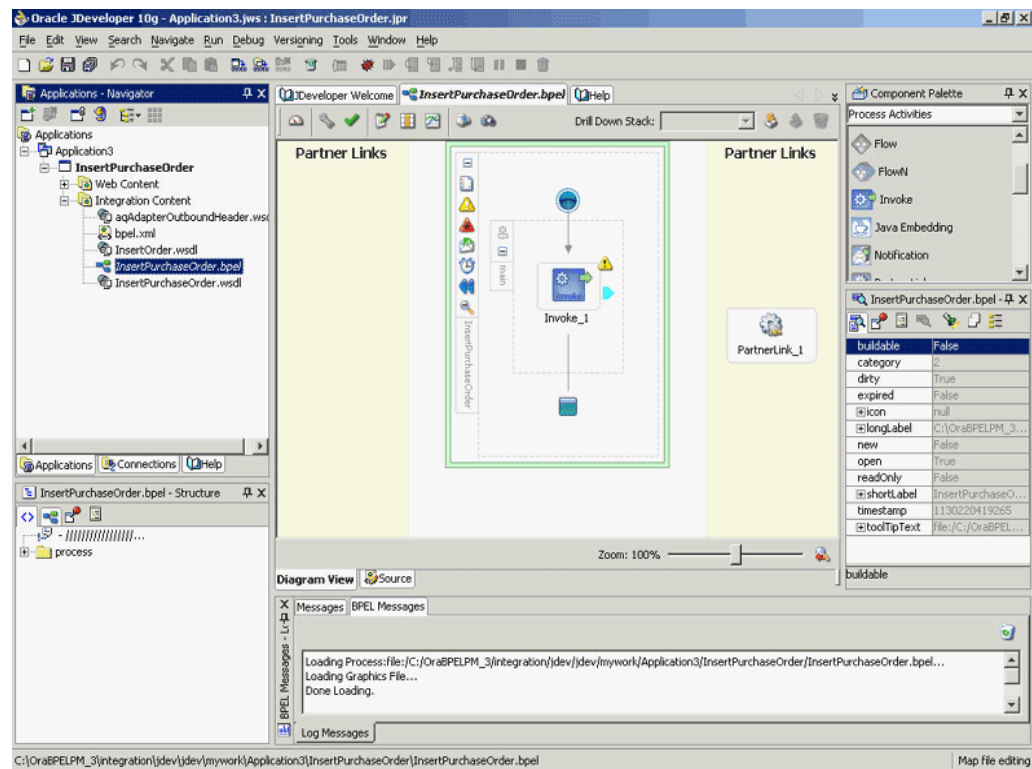
6. Click **OK**. The partner link is created with the required WSDL settings.

6.2.2.3 Configuring the Invoke Activity

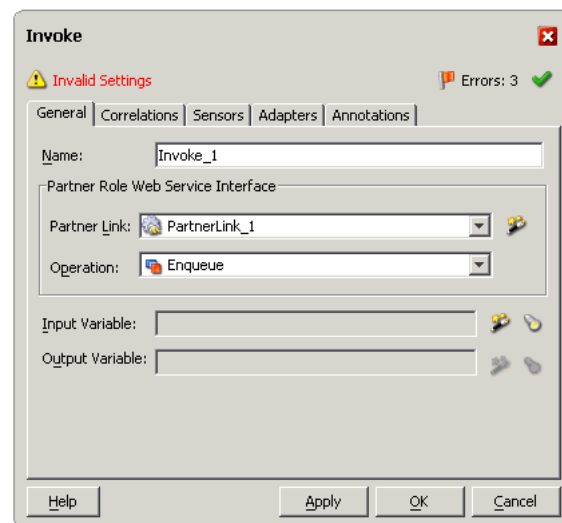
After adding and configuring the partner link, the next task is to configure the BPEL process itself. You can start by configuring the **Invoke** process activity to enqueue the XML Gateway inbound messages.

The following steps discuss configuring the Invoke activity:

1. Drag and drop **Invoke** into the process map window. [Figure 6–13](#) shows the **Invoke** activity after it has been added to the process map.

Figure 6–13 Adding the Invoke Activity

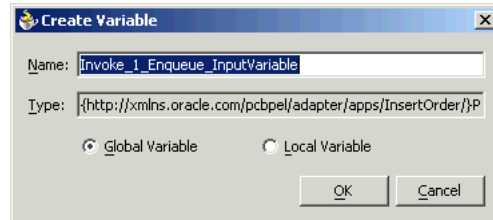
2. Double-click **Invoke** in the process map to open the Invoke dialog box. The **General** tab is selected by default. [Figure 6–14](#) shows the Invoke dialog box.

Figure 6–14 Configuring the Invoke Activity

3. In the **Partner Link** box, select the partner link to invoke. This is the partner link that you configured in the previous section. The **Operation** is automatically selected, depending on the message map that you chose when configuring the partner link. If you selected an inbound message map, then the Enqueue operation is selected.

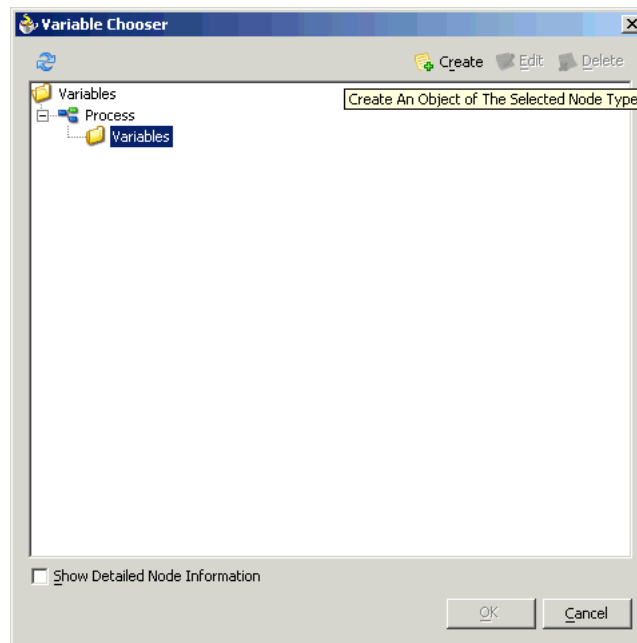
4. Click the **Create** icon next to the **Input Variable** field. Enter a descriptive name for the variable in the Create Variable dialog box that appears. You can also accept the default name. Click **OK**. [Figure 6–15](#) shows the Create Variable dialog box.

Figure 6–15 Creating the Input Variable

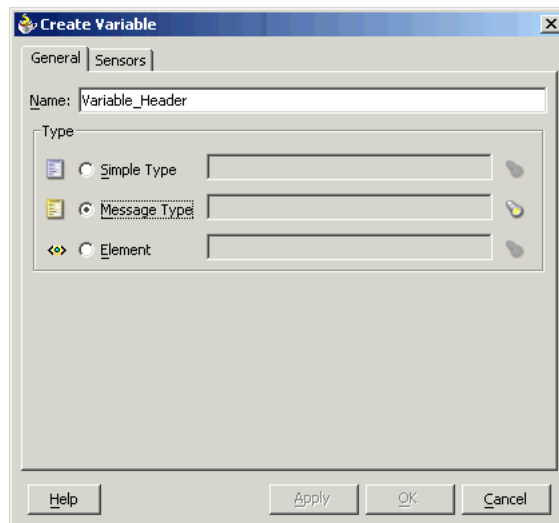


5. Click the **Adapters** tab in the Invoke dialog box. You need to specify the **Input Header Variable** here. Click the **Browse** icon next to the **Input Header Variable** field.
6. In the Variable Chooser dialog box that appears, select the Variables node and click **Create**. [Figure 6–16](#) shows the Variable Chooser dialog box.

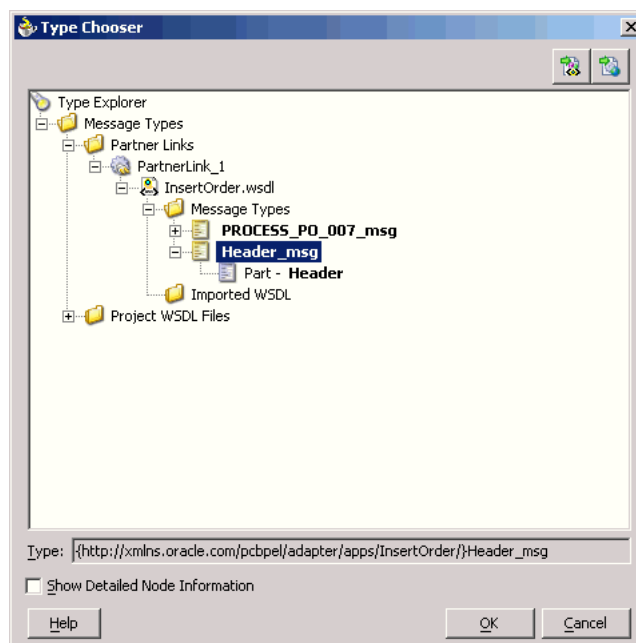
Figure 6–16 Specifying the Input Header Variable



7. The **Create Variable** dialog box is displayed. [Figure 6–17](#) shows the Create Variable dialog box. Type a descriptive name for the variable in the **Name** field. Select the **Message Type** option and click the **Browse** icon next to it.

Figure 6–17 Creating the Input Header Variable

8. In the Type Chooser dialog box that appears, select the appropriate message type and click OK. Figure 6–18 shows the Type Chooser dialog box.

Figure 6–18 Specifying the Message Type

9. In the Create Variable dialog box, click OK. In the Variable Chooser dialog box, click OK.
10. In the Invoke dialog box, click **Apply**. Click **OK**.

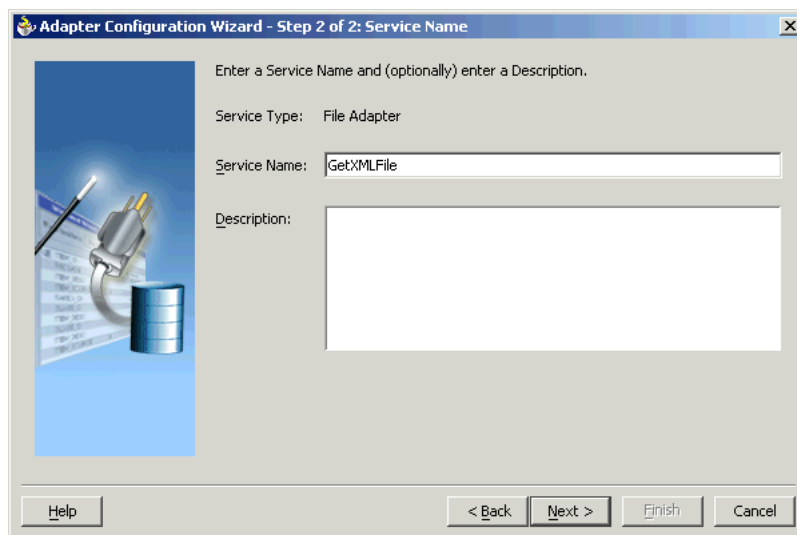
Note: If you have configured a partner link for outbound messages from Oracle Applications, then you need to configure a **Receive** activity in place of the **Invoke** activity. The **Receive** activity is used to dequeue the XML Gateway outbound messages.

6.2.2.4 Adding a Partner Link for the File Adapter

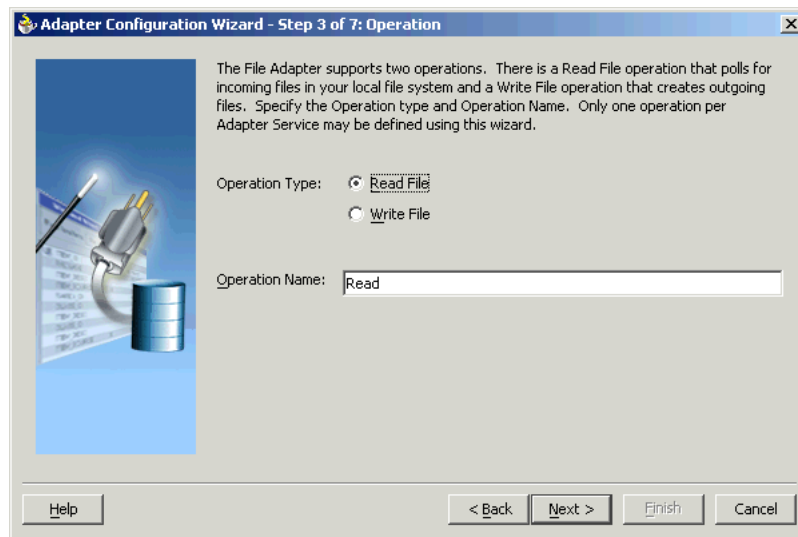
If you are configuring an inbound message, then you would need to add another partner link for the file adapter. This allows the inbound message to pick up an XML file received from the third party application. The data is inserted into Oracle Applications through the partner link that was configured earlier. Use the following steps to add a partner link for the file adapter:

1. Drag and drop **PartnerLink** into the border area of the process diagram. The Create Partner Link dialog box is displayed.
2. Click the **Define Adapter Service** icon in the WSDL Settings section. The Adapter Configuration Wizard is displayed.
3. Click **Next**. The Adapter Type dialog box is displayed.
4. Select **File Adapter**. Click **Next**.
5. Enter a name for the file adapter service in the **Service Name** field. You can also enter an optional description. [Figure 6–19](#) shows the Service Name dialog box. Click **Next**.

Figure 6–19 Specifying the Service Name

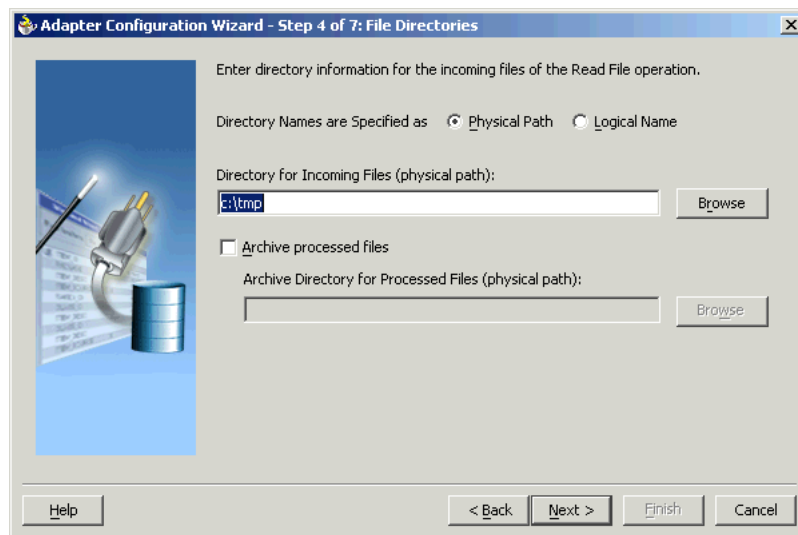


6. The Operation dialog box is displayed. Select the **Operation Type**. The **Operation Type** specifies whether you are polling for files or creating outgoing files. Select **Read File** if you need to read files for inbound operations. Enter a name for the Operation in the **Operation Name** field. You can also use the default name. [Figure 6–20](#) shows the Operation dialog box. Click **Next**.

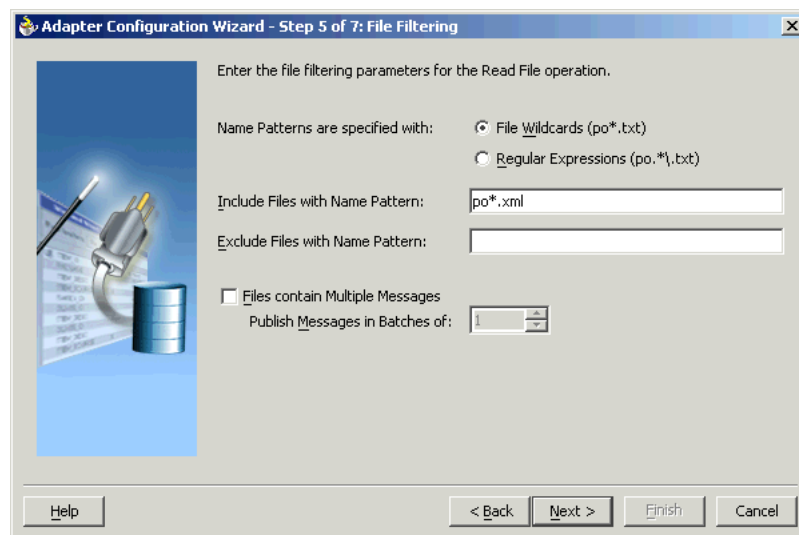
Figure 6–20 Specifying the Operation Name and Type

Note: If you are configuring the BPEL Process for outbound XML messages from Oracle Applications, then you need to choose **Write File** in place of **Read File**.

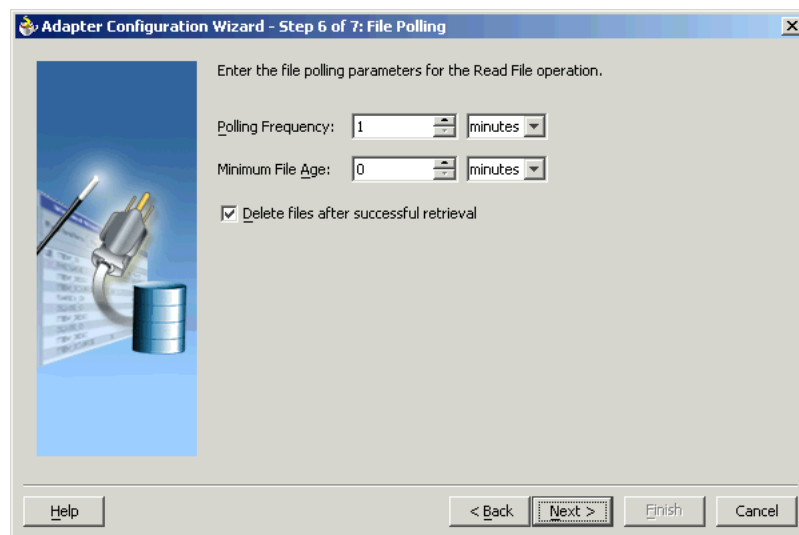
7. The File Directories dialog box is displayed. Enter the physical path for the XML files. You can also choose the location by using **Browse**. [Figure 6–21](#) shows the File Directories dialog box. Click **Next**.

Figure 6–21 Specifying the Physical Path for the XML Files

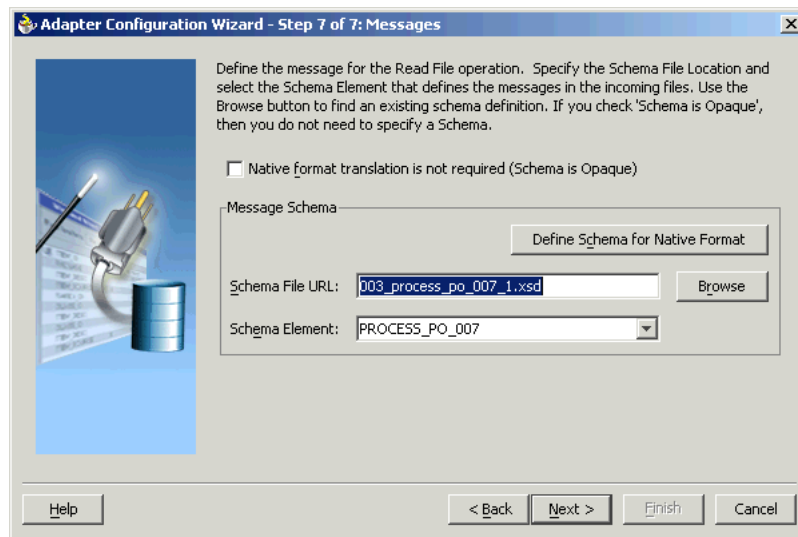
8. The File Filtering dialog box is displayed. This enables you to specify the pattern using which you want to match the XML files that need to be picked. [Figure 6–22](#) shows the File Filtering dialog box. Specify a pattern and click **Next**.

Figure 6–22 Specifying File Filtering Parameters for the Read Operation

9. The File Polling dialog box is displayed. You need to specify the frequency at which the directory needs to be polled for the XML files. Specify a **Polling Frequency**. Click **Next**. Figure 6–23 shows the File Polling dialog box.

Figure 6–23 Specifying the Polling Frequency

10. The Messages dialog box is displayed, as shown in Figure 6–24. You need to specify the schema file location and select the schema element that defines the messages in the incoming files. You can use **Browse** to select the schema file location. Enter the required information, and then lick **Next**.

Figure 6–24 Specifying the Schema File Location and Schema Element

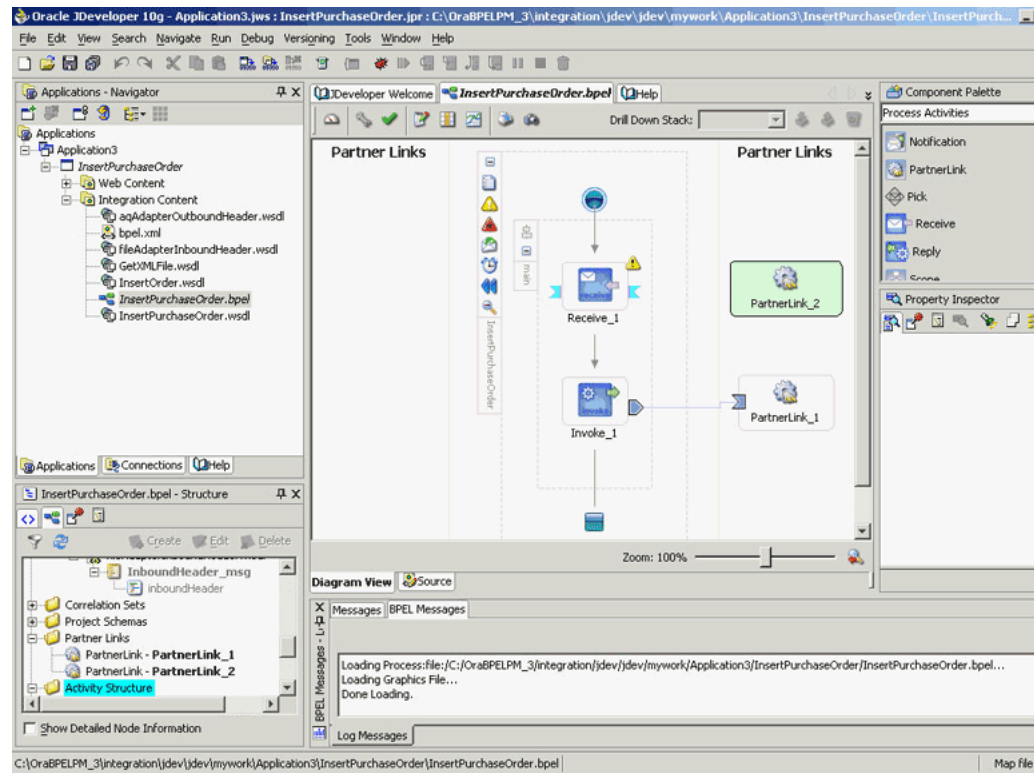
11. Click **Finish** to finish creating the File Adapter service. Click **OK** in the Create Partner Link dialog box to create the partner link for the File Adapter service.

Note: The File Adapter is just one of the ways to read or write XML messages. The XML messages can also be exchanged using an FTP Adapter, SMTP Adapter, or through another BPEL process.

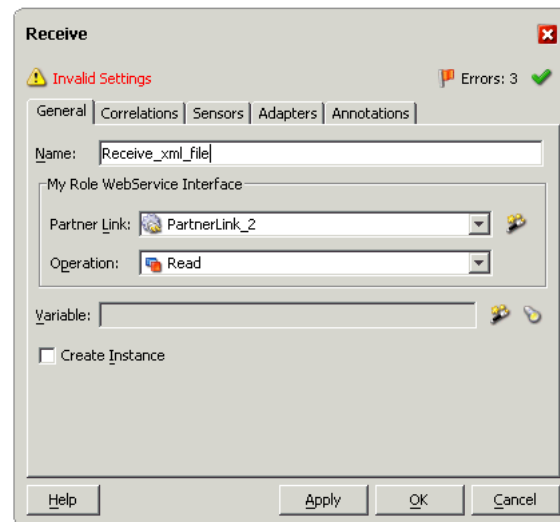
6.2.2.5 Configuring the Receive Activity

The next task is to configure a Receive activity to receive XML data from the partner link that you configured for the file adapter service. Use the following steps to configure the Receive activity:

1. Drag and drop **Receive** into the process map window. The **Receive** activity should be placed in between **Start** and **Invoke**. [Figure 6–25](#) shows the process map window after the **Receive** activity has been added.

Figure 6–25 Adding the Receive Activity

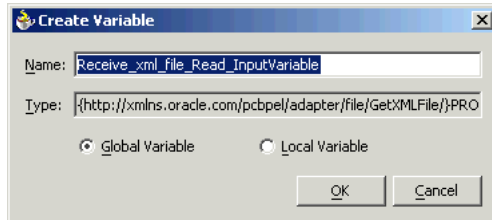
2. Double-click **Receive** in the process map to open the **Receive** dialog box. The **General** tab is selected by default. [Figure 6–26](#) shows the Receive dialog box.

Figure 6–26 Configuring the Receive Activity

3. In the **Partner Link** box, choose the partner link corresponding to the file adapter service. The **Operation** is selected by default depending on the **Partner Link** that you select.
4. Click the **Create** icon next to the **Variable** field.

5. The **Create Variable** dialog box appears. Enter a **Name** for the variable. You can also accept the default name. Click **OK**. [Figure 6–27](#) shows the Create Variable dialog box.

Figure 6–27 *Creating the Receive Variable*



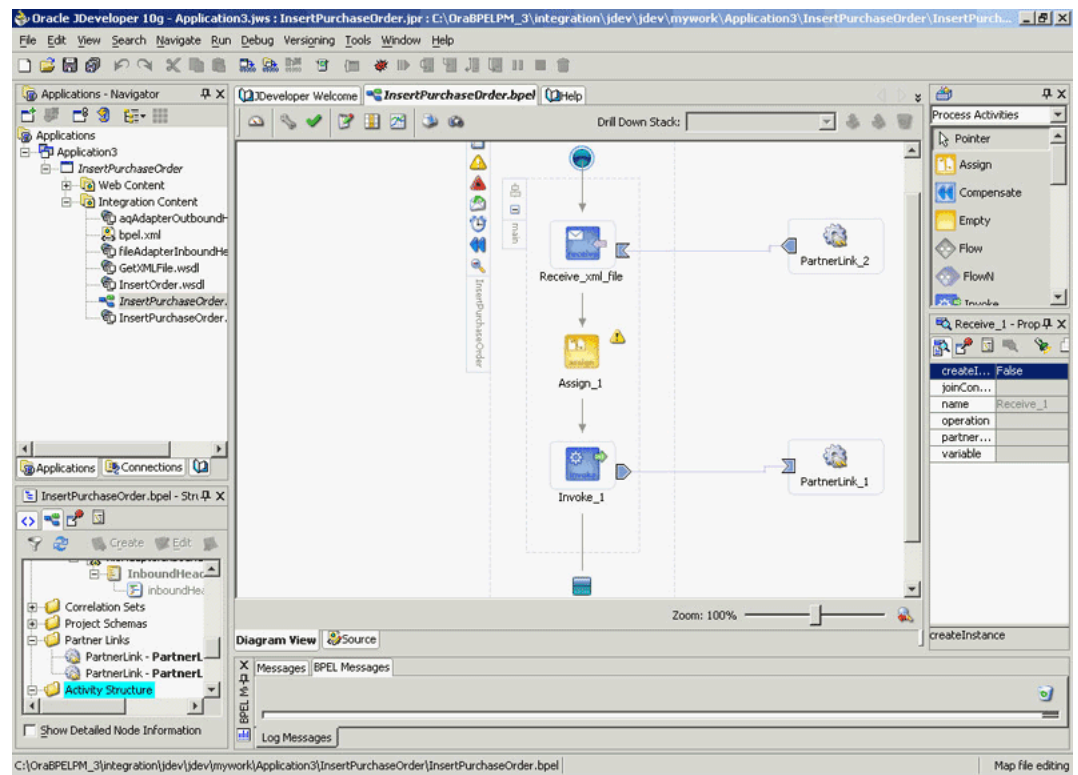
6. Click **Apply** in the Receive dialog box, and then click **OK**. This associates the **Receive** activity with the partner link configured for the file adapter service.

Note: If you are configuring the process for outbound XML messages from Oracle Applications, then you need to link the file adapter partner link to the **Invoke** activity. The Invoke process activity invokes the file adapter interface to write the XML message to an XML file.

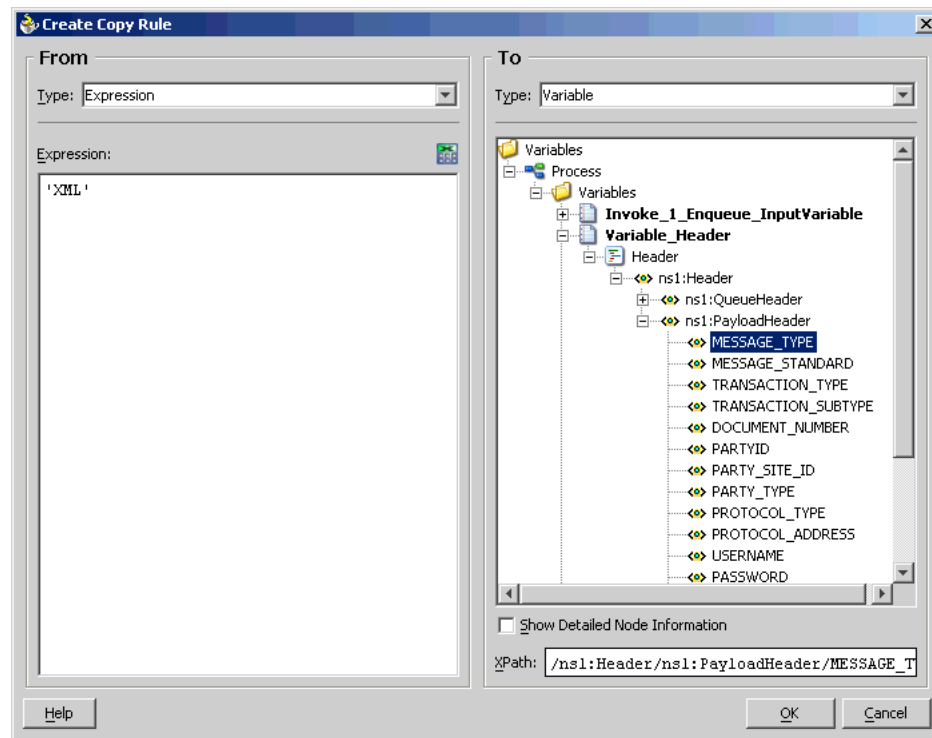
6.2.2.6 Configuring the Assign Activity

The next task is to add an Assign activity to the process map. This is used to populate parameters that provide context information to Oracle Applications. The following steps discuss this task:

1. Drag and drop the **Assign** activity to the process map. The Assign activity needs to be dropped in between the **Receive** and **Invoke** activities. [Figure 6–28](#) shows the process map as it appears after adding the **Assign** activity.

Figure 6–28 Adding the Assign Activity

2. You now need to configure the **Assign** activity. Double-click the **Assign** activity in the process map.
3. The Assign dialog box is displayed. The **Copy Rules** tab is displayed by default. Click **Create**.
4. The Create Copy Rule dialog box is displayed. In the To group, expand the Variables node by clicking the plus sign next to it. Expand the Variable_Header, Header, ns1:Header, and the ns1:PayloadHeader child nodes. [Figure 6–29](#) shows the Create Copy Rule dialog box.

Figure 6–29 Specifying Context Information for Oracle Applications

5. Select the MESSAGE_TYPE variable. In the From group, select Expression as the **Type**. Enter a MESSAGE_TYPE in the **Expression** field. The MESSAGE_TYPE would be 'XML' for XML Gateway. Click **OK**.
6. Repeat steps 3 and 4. Select the MESSAGE_STANDARD variable. MESSAGE_STANDARD is the XML message standard that you are using. Enter an **Expression** for the MESSAGE_STANDARD, say 'OAG'. Click **OK**.
7. Repeat steps 3 and 4. Select the TRANSACTION_TYPE variable. This variable defines the type of transaction that you are performing, say a purchase order operation. Enter an **Expression** for the TRANSACTION_TYPE, say 'PO'. Click **OK**.
8. Repeat steps 3 and 4. Select the TRANSACTION_SUBTYPE variable. This variable defines the subtype for the transaction that you are performing. Enter an **Expression** for the TRANSACTION_SUBTYPE, say 'Process'. This means that you are processing a Purchase Order TRANSACTION_TYPE. Click **OK**.
9. Repeat steps 3 and 4. Select the PARTY_SITE_ID variable. The PARTY_SITE_ID identifies the trading partner. Enter an **Expression** for the PARTY_SITE_ID. Click **OK**.

Note: The MESSAGE_TYPE, MESSAGE_STANDARD, TRANSACTION_TYPE, TRANSACTION_SUBTYPE, and PARTY_SITE_ID are the mandatory header variables that you need to populate for the XML transaction to complete successfully.

6.3 Run-Time Steps for XML Gateway Inbound into Oracle Applications

After designing the BPEL process, the next step is to deploy, run and monitor it. This section discusses the following:

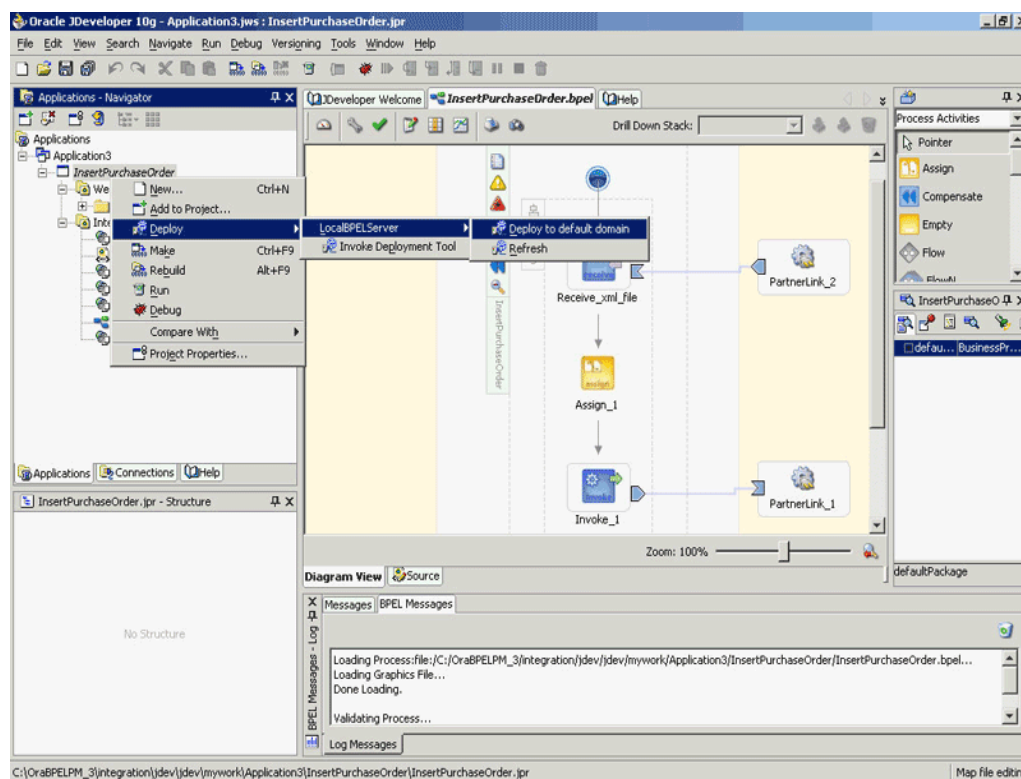
- [Deploying the BPEL Process](#)
- [Testing the BPEL Process](#)
- [Verifying Records in Oracle Applications](#)

6.3.1 Deploying the BPEL Process

You need to deploy the BPEL process before you can run it. The BPEL process is first compiled and then deployed to the BPEL server. The following steps discuss deploying the BPEL process to a BPEL server:

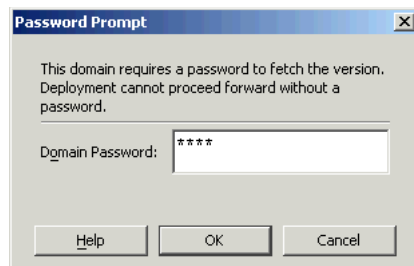
1. Select the BPEL project in the Applications window.
2. Right-click the project name. Select **Deploy** from the menu that appears.
3. Select **Local BPEL Server** followed by **Deploy to Default Domain**, if you are deploying the process on the local BPEL server. [Figure 6–30](#) illustrates deploying a BPEL process to a local BPEL server.

Figure 6–30 *Deploying the BPEL Process*

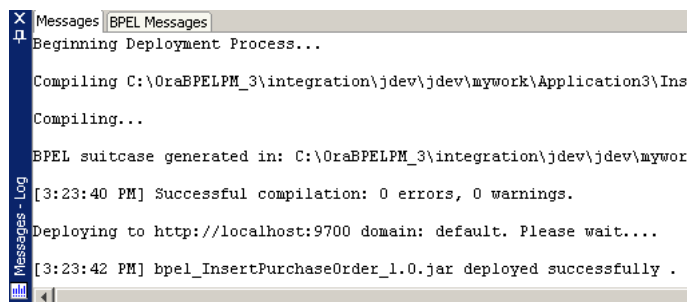


Note: You can select **Invoke Deployment Tool** if you want to deploy to a different BPEL server.

4. The Password Prompt dialog box appears. Enter the password for the default domain in the **Domain Password** field. Click **OK**. [Figure 6–31](#) shows the Password Prompt dialog box.

Figure 6–31 Specifying the Domain Password

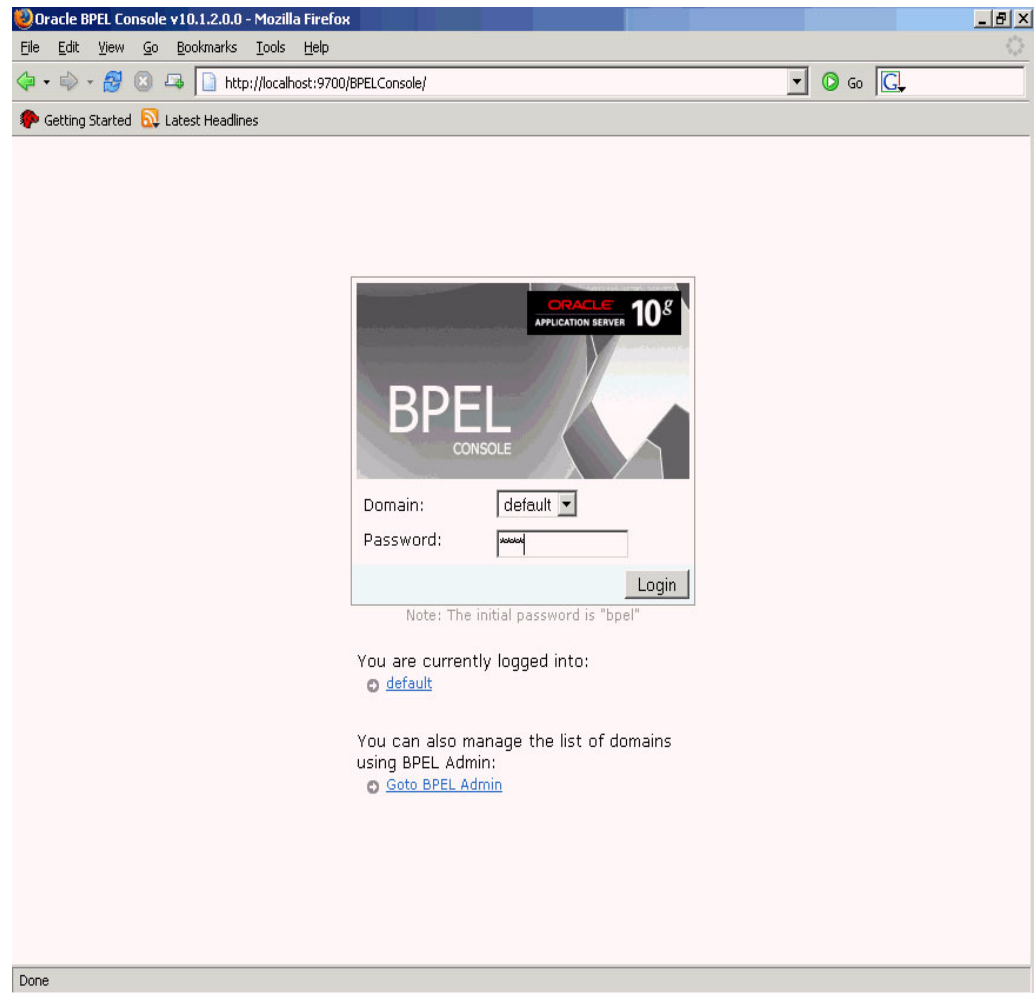
5. The BPEL process is compiled and deployed. You can check the progress in the Messages window. [Figure 6–32](#) shows the Messages window.

Figure 6–32 Messages Window

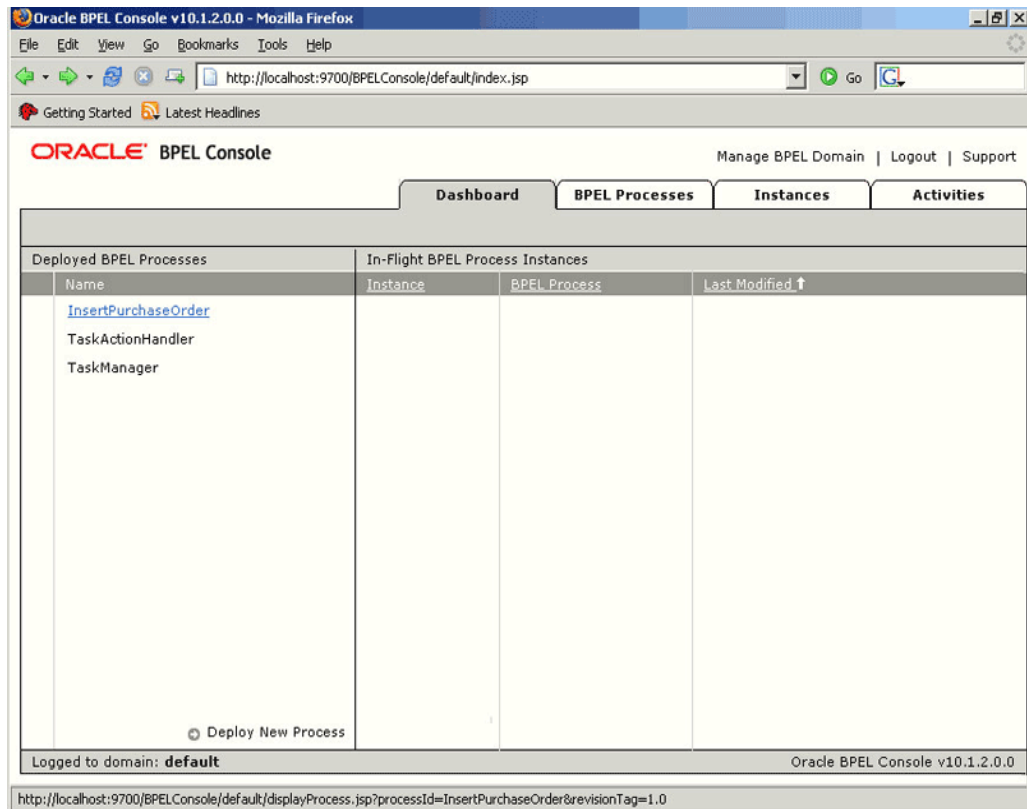
6.3.2 Testing the BPEL Process

Once the BPEL process is deployed, it can be seen in the BPEL console. You can manage and monitor the process from the BPEL console. You can also test the process and the integration interface by manually initiating the process. The following steps discuss manually initiating and monitoring the BPEL process:

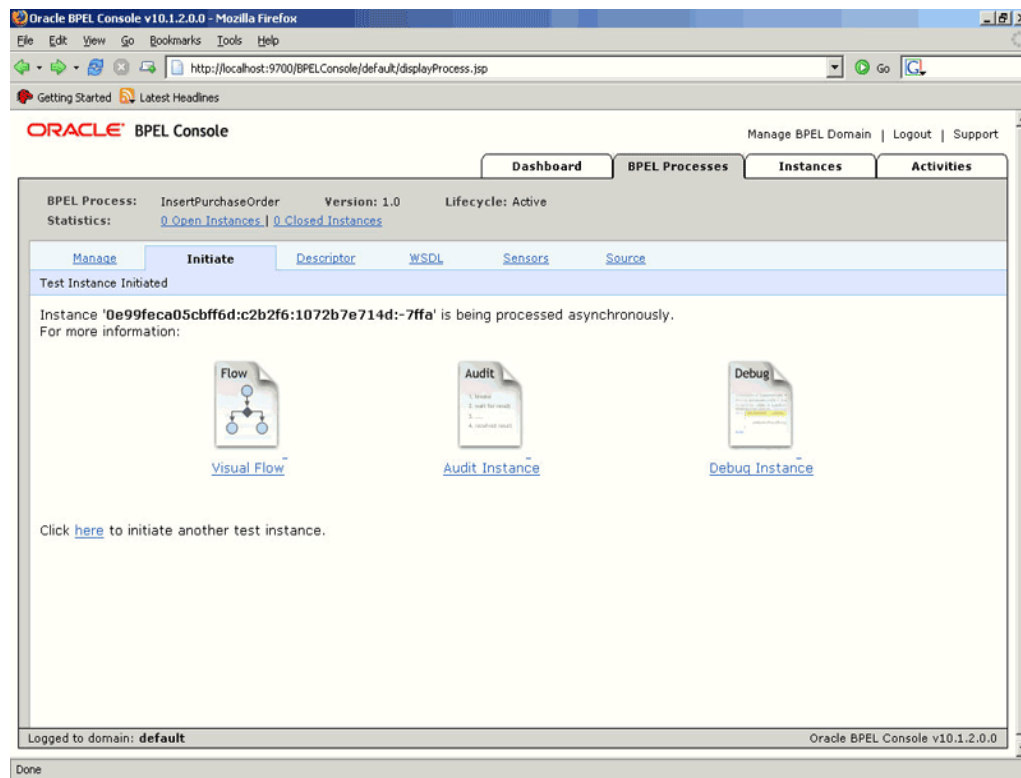
1. To open the BPEL console, click **Start**, and then choose **Programs**. In the Programs menu, select **Oracle - ORACLE_HOME, Oracle BPEL Process Manager 10.1.2**, and then select **BPEL Console**.
2. The BPEL console login screen is displayed. Select **Default** in the **Domain** box. Enter the password for the default domain in the **Password** field. Click **Login**. [Figure 6–33](#) shows the BPEL console login screen.

Figure 6–33 BPEL Console Login Screen

3. Oracle BPEL console is displayed. The list of deployed processes is shown under Deployed BPEL Processes. [Figure 6–34](#) shows the BPEL console screen.

Figure 6–34 Deployed BPEL Processes

4. Click the BPEL process that you want to initiate. The Initiate page is displayed. Enter the input values required by the process. You can also specify an XML file for the file adapter to pick.
5. Click **Post XML Message** to initiate the process.
6. The BPEL process is now initiated. You can check the process flow by clicking the **Visual Flow** icon. [Figure 6–35](#) shows the BPEL Console Initiate page.

Figure 6–35 BPEL Console Initiate Page

7. The audit trail provides information on the steps that have been executed. The audit trail also records the Reference ID that is returned for the transaction. You can check the audit trail by clicking the **Audit Instance** icon.

If the BPEL process runs into an error, then a corresponding error code is returned. Table 6–1 lists the common error codes, their descriptions, and the fix that you can use for them.

Table 6–1 Error Codes for the BPEL Process

Code No.	Code Name	Description	Fix
12400	APPS_MESSAGE_STANDARD_NOT_FOUND	Message Standard not set in the header	Set the Message Standard value in the header
12401	APPS_TRANSACTION_TYPE_NOT_FOUND	Transaction Type not set in the header	Set the Transaction Type value in the header
12402	APPS_TRANSACTION_SUBTYPE_NOT_FOUND	Transaction Subtype not set in the header	Set the Transaction Subtype value in the header
12403	APPS_PARTY_SITE_ID_NOT_FOUND	Party Site Id not set in the header	Set the Party Site Id value in the header

Table 6–1 (Cont.) Error Codes for the BPEL Process

Code No.	Code Name	Description	Fix
12404	APPS_MESSAGE_TYPE_NOT_FOUND	Message Type not set in the header	Set the Message Type value in the header
12406	APPS_CONTEXT_ERROR	Error in setting Apps Context	Check the username and responsibility values
12407	APPS_AUTHENTICATION_ERROR	Invalid FND username/password	Check the username and password values
12408	APPS_UNKNOWN_EX	Unknown error in Apps Interaction	Check if all the header values are valid
12409	APPS_XMLG_HEADER_NULL	XML Gateway header is null	Pass the required parameters in header

6.3.3 Verifying Records in Oracle Applications

You can verify all successful entries in Oracle Applications. You need to look at the relevant module in Oracle Applications to check for the new records. For example, you can look for a Sales Order entry in the Order Management module.

You can check for unsuccessful entries in corrections. You can use the Reference ID of a transaction to look for a particular record. This is the same Reference ID that you noted in the BPEL audit trail. You can reprocess a record after making the necessary configuration changes. However, you cannot alter the data in the record.

6.3.3.1 Using Transaction Monitor

The Transaction Monitor is a tool for monitoring the status of inbound and outbound transactions originating from and going into Oracle Applications that have been processed by the XML Gateway and delivered or received by the Oracle Transport Agent. The Transaction Monitor shows a complete history and audit trail of these documents.

You can navigate to the Transaction Monitor page using the Workflow Administrator Web responsibility. The Transaction Monitor provides the following:

- Flexible search criteria to support access to a specific document or group of documents
- Search results at the document header level with drill down by document ID
- Resend capability for outbound messages
- Viewing capability of the XML message content

See Also: *Oracle XML Gateway User's Guide* for details on using the Transaction Monitor. This guide is a part of the Oracle Applications documentation library. Oracle Applications documentation can be accessed from the following link:

<http://www.oracle.com/technology/documentation/applications.html>

6.4 Design-Time Steps for XML Gateway Outbound from Oracle Applications

This section discusses the design-time steps, for an XML Gateway outbound message, that are different from the design-time steps for an inbound message. This section includes the following topics:

- [Prerequisites to Configuring OracleAS Adapter for Oracle Applications](#)
- [Configuring OracleAS Adapter for Oracle Applications](#)

6.4.1 Prerequisites to Configuring OracleAS Adapter for Oracle Applications

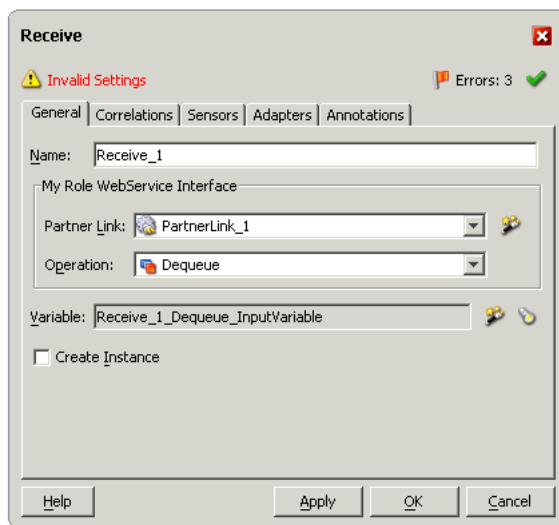
For invoking an outbound partner link, you need to set up the correlation identifier in Oracle Applications. The correlation identifier enables you to label messages meant for a specific agent, in case there are multiple agents listening on the outbound queue. The agent listening for a particular correlation picks up the messages that match the correlation identifier for the agent. You can use the following steps to set up the correlation identifier:

1. Log in to Oracle Applications with the XML Gateway responsibility. The Navigator page is displayed.
2. Click the **XML Gateway** link.
3. Click the **Define Lookup Values** link under XML Gateway.
4. Enter `COMM_METHOD` for the **Type** field.
5. Enter `BPEL` for the **Code** field. Fill in the other mandatory fields. Oracle XML Gateway puts the correlation of BPEL when enqueueing the message on the `ECX_OUTBOUND` queue.

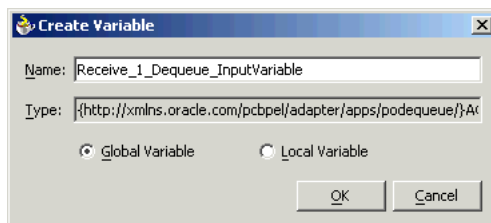
6.4.2 Configuring OracleAS Adapter for Oracle Applications

When configuring the OracleAS Adapter for Oracle Applications to use an outbound XML Gateway map, you need to configure the Receive activity for the associated partner link. The Receive activity dequeues the outbound XML messages. The following steps are used to configure the Receive activity:

1. Drag and drop **Receive** into the process map window.
2. Double-click the **Receive** activity to configure the **Receive** properties. The Receive dialog box is displayed. [Figure 6-36](#) shows the Receive dialog box.

Figure 6–36 Configuring the Receive Activity

3. Select the **Partner Link** corresponding to the partner link configured with the outbound XML Gateway map.
4. The **Dequeue Operation** is automatically selected if the partner link has been configured with an outbound XML Gateway map.
5. The next task is to specify a **Variable** to receive the message data from the partner link. Click the **Create** icon to the right of the Variable field. The **Create Variable** dialog box is displayed. [Figure 6–37](#) shows the Create Variable dialog box.

Figure 6–37 Creating a Variable to Receive Message Data

6. Type a **Name** for the variable. You can also accept the default name. Click **OK**.
7. Click **Apply** in the Receive dialog box. Click **OK**.

Note: You can define a **Header Variable** under the **Adapters** tab of the Receive dialog box. This header variable is populated with context information from the outbound XML message. Values for fields like MESSAGE_TYPE, MESSAGE_STANDARD and trading party information like PARTY_SITE_ID are returned through this variable.

6.5 Run-Time Steps for XML Gateway Outbound from Oracle Applications

After deploying the BPEL process, you can compile, deploy and test it. In Oracle Applications, you can check for outbound transactions that have been processed by the XML Gateway and delivered to the Transaction Agent, by using the Transaction Monitor. You can also use the Transaction Monitor to resend an outbound document, if

necessary. [Section 6.3, "Run-Time Steps for XML Gateway Inbound into Oracle Applications"](#) discusses the Transaction Monitor.

See Also: *Oracle XML Gateway User's Guide* for details on using the Transaction Monitor. This guide is a part of the Oracle Applications documentation library. Oracle Applications documentation can be accessed from the following link:

<http://www.oracle.com/technology/documentation/applications.html>

If you have used a file adapter to write outbound messages from Oracle Applications to files, then you can check the output directory location for the presence of these files after the BPEL process has run.

Sample WSDL and oc4j-ra.xml File

This appendix contains the following sections:

- [Sample WSDL File](#)
- [Sample oc4j-ra.xml File](#)

A.1 Sample WSDL File

A Web Service Definition Language (WSDL) is generated by the JDeveloper BPEL Designer during design time. The WSDL file generated by the Adapter Wizard is the adapter service definition. In addition, it specifies various operations exposed by the service. The operations are based on user input to the Adapter Wizard. An operation either retrieves or inserts data to Oracle Applications and is represented by a JCA activation or interaction spec.

Figure A–1 shows a sample WSDL file.

Figure A–1 Sample WSDL File

```
<binding name="Ora_binding" type="tns:Ora_ptt">
<jca:binding />
  <operation name="Ora">
    <jca:operation
      SchemaName="APPS"
      PackageName="XXBPPEL_CUSTOMER"
      ProcedureName="CREATE_PERSON_PRIMITIVE"
      InteractionSpec="oracle.tip.adapter.apps.AppsStoredProcedureInteractionSpec" />
    </jca:operation>
    <input />
  </operation>
</binding>
<service name="Ora">
  <port name="Ora_ptt" binding="tns:Ora_binding">
    <!-- Your runtime connection is declared in
    J2EE_HOME/application-deployments/default/DbAdapter/oc4j-ra.xml
    These mct properties here are from your design time connection and
    save you from having to edit that file and restart the application server
    if eis/Apps/OracleApps is missing.
    These mct properties are safe to remove -->
    <jca:address location="eis/Apps/OracleApps" UIConnectionName="OracleApps" UIOracleAppType="
    DBOBJECT"
    ManagedConnectionFactory="oracle.tip.adapter.apps.AppsManagedConnectionFactory"
    mct.ConnectionString="jdbc:oracle:thin:@la2037.oracleads.com:1521:la2037"
    mct.UserName="apps"
    mct.Password="53CB0F044A0D3DD2C063679F18F89870" />
  </port>
</service>
<plt:partnerLinkType name="Ora_ptt" >
<plt:role name="Ora_role" >
  <plt:portType name="tns:Ora_ptt" />
</plt:role>
```

A.2 Sample oc4j-ra.xml File

OracleAS Adapter for Oracle Applications is deployed as J2CA 1.5 resource adapters within the same OC4J container as BPEL Process Manager during installation. Although the OracleAS Adapter for Oracle Applications is physically deployed as J2CA 1.5 resource adapters, the logical deployment of the adapter involves creating the connection entries for the J2CA 1.0 resource adapter by editing the `oc4j-ra.xml` file and using JDeveloper during design time. For the logical deployment changes to take effect, the OC4J container process must be restarted.

You can modify the connection and login properties in the `oc4j-ra.xml` file. A sample `oc4j-ra.xml` is shown as follows:

```
<connector-factory location="eis/Apps/apps1" connector-name="Oracle Applications
Adapter">
    <config-property name="connectionString"
value="jdbc:oracle:thin:@localhost:1215:orcl"/>
    <config-property name="userName" value="oraapps"/>
    <config-property name="password" value="oraapps"/>
    <config-property name="usesExternalConnectionPooling" value="false"/>
    <config-property name="dataSourceName" value=""/>
    <config-property name="usesExternalTransactionController" value="false"/>
</connector-factory>
```

For a Middle Tier installation, refer to the `oc4j-ra.xml` file at the following location:

```
$ORACLE_HOME\j2ee\OC4J_
BPEL\application-deployments\default\AppsAdapter
```

For a BPEL Process Manager with standalone OC4J installation, refer to the `oc4j-ra.xml` file at the following location:

```
$ORACLE_
HOME\integration\orabpel\system\appserver\oc4j\j2ee\home\applica
tion-deployments\default\AppsAdapter
```

Note: The run-time errors that might for OracleAS Adapter for Oracle Applications are similar to OracleAS Adapter for Databases. Refer to the Troubleshooting the OracleAS Adapter for Databases section in *Oracle BPEL Process Manager Developer's Guide* for information about handling error messages.

Index

A

APIs

- adding a new partner link, 3-4
- configuring OracleAS Adapter for Oracle Applications, 3-2
- configuring the Invoke activity, 3-15
- configuring the Transform activity, 3-16
- creating a new BPEL project, 3-2
- deploying the BPEL process, 3-18
- design-time steps, 3-1
- overview, 3-1
- prerequisites to configure, 3-1
- run-time steps, 3-18
- testing the BPEL process, 3-20

B

B2B, 6-2

base tables, 2-1

C

concurrent programs

- adding a partner link, 4-4
- configuring OracleAS Adapter for Oracle Applications, 4-2
- configuring the Invoke activity, 4-9
- configuring the Transform activity, 4-11
- creating a new BPEL project, 4-2
- deploying the BPEL process, 4-14
- design-time steps, 4-1
- overview, 4-1
- prerequisites to configure, 4-1
- run-time steps, 4-14
- testing the BPEL process, 4-16
- verifying records, 4-18

E

EDI

- adding a partner link, 5-4
- configuring OracleAS Adapter for Oracle Applications, 5-2
- configuring the Invoke activity, 5-10
- configuring the Transform activity, 5-12
- creating a new BPEL project, 5-2

deploying the BPEL process, 5-15

design-time steps, 5-2

overview, 5-1

prerequisites to configuring OracleAS Adapter for Oracle Applications, 5-2

run-time steps, 5-15

testing the BPEL process, 5-17

verifying records in Oracle Applications, 5-20

I

integration architecture, 6-1

interface tables, 2-1

adding a partner link, 2-4

configuring OracleAS Adapter for Oracle Applications, 2-2

configuring the Assign activity, 2-13

configuring the Invoke activity, 2-11

creating a new BPEL project, 2-2

deploying the BPEL process, 2-14

design-time steps, 2-2

overview, 2-1

prerequisites to configure, 2-2

run-time steps, 2-14

testing the BPEL process, 2-16

M

message queues, 6-2

inbound queues, 6-2

outbound queues, 6-3

N

new features, 1-3

O

Oracle Applications

overview, 1-1

OracleAS Adapter for Oracle Applications

architecture, 1-2

features, 1-2

installing e-Business Suite Adapter, 1-4

interfaces, 1-3

issues and workarounds, 1-4

- new features, 1-4
- overview, 1-1

S

- sample oc4j-ra.xml file, A-2
- sample WSDL file, A-1
- standards-based messaging, 6-1

V

- views, 2-18
 - adding a partner link, 2-21
 - configuring Oracles Adapter for Oracle Applications, 2-19
 - configuring the Assign activity, 2-31
 - configuring the Invoke activity, 2-29
 - creating a new BPEL project, 2-19
 - deploying the BPEL process, 2-32
 - design-time steps, 2-19
 - overview, 2-18
 - prerequisites to configure, 2-19
 - run-time steps, 2-32
 - testing the BPEL process, 2-34

W

- wrapper APIs, 3-10
 - describing DEFAULT clause handling in wrapper procedures, 3-14
 - describing parameters with DEFAULT clause, 3-13
- WSDL, A-1

X

- xml
 - overview, 6-1
- XML gateway envelope, 6-3
 - ATTRIBUTE3, 6-4
 - DOCUMENT_NUMBER, 6-3
 - MESSAGE_STANDARD, 6-3
 - MESSAGE_TYPE, 6-3
 - parameters defined by the application, 6-4
 - parameters not used, 6-4
 - PARTY_SITE_ID, 6-4
 - PASSWORD, 6-4
 - PAYLOAD, 6-4
 - PROTOCOL_ADDRESS, 6-3
 - PROTOCOL_TYPE, 6-3
 - SOURCE_TP_LOCATION_CODE, 6-3
 - TRANSACTION_SUBTYPE, 6-3
 - TRANSACTION_TYPE, 6-3
 - USERNAME, 6-4
- XML gateway inbound
 - adding a partner link for the file adapter, 6-17
 - configuring OracleAS Adapter for Oracle Applications, 6-5
 - configuring the Assign activity, 6-22
 - configuring the Invoke activity, 6-13
 - configuring the Receive activity, 6-20

- creating a new BPEL project, 6-5
- creating a partner link, 6-8
- deploying the BPEL process, 6-25
- design-time steps, 6-5
- prerequisites to configure, 6-5
- run-time steps, 6-24
- testing the BPEL process, 6-26
- using transaction monitor, 6-30
- verifying records, 6-30
- XML gateway outbound
 - configuring OracleAS Adapter for Oracle Applications, 6-31
 - design-time steps, 6-31
 - prerequisites to configure, 6-31
 - run-time steps, 6-32