Oracle® Fusion Middleware

Adapter for Oracle Applications User's Guide

11*g* Release 1 (11.1.1.4.0)

Part No. E10537-03

December 2010

ORACLE

Oracle Fusion Middleware Adapter for Oracle Applications User's Guide, 11*g* Release 1 (11.1.1.4.0)

Part No. E10537-03

Copyright © 2005, 2010, Oracle and/or its affiliates. All rights reserved.

Primary Author: Melody Yang

Contributing Author: Sravani Bheemireddy, Neeraj Chauhan, Vimmika Dinesh, Anil Kemisetti, Nadakuditi Ravindra, Veshaal Singh, Sheela Vasudevan

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. 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, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

This software and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third party content, products and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third party content, products or services.

Contents

Send Us Your Comments

Preface

1 Introduction to Adapter for Oracle Applications

Overview of Adapter for Oracle Applications 1-1
Integration with Oracle BPEL Process Manager 1-4
Integration with Oracle WebLogic Server 1-5
New Features in This Release 1-6

2 Adapter for Oracle Applications Features

Overview 2-1
Support for Various Integration Interface Types 2-1
Support for Oracle Integration Repository 2-3
Support for Custom Integration Interfaces in Various Versions of Oracle E-Business Suite 2-4

3 Adapter for Oracle Applications Concepts

Understanding Applications Context 3-1
Supporting for Normalized Message Properties 3-8
Supporting for Multiple Organization Setups 3-12
Supporting for Multiple Languages 3-14
Understanding Adapter for Oracle Applications Security 3-15
Logging 3-20
Secured Connection Between Oracle E-Business Suite and Oracle Fusion Middleware SOA Suite Using J2EE Data Source Implementation 3-34

Understanding the Oracle Applications Module Browser....3-35

4 Using XML Gateway

Overview of XML Gateway....4-2
Design-Time Tasks for XML Gateway Inbound Messaging....4-7
Creating a New BPEL Project....4-9
Creating a Partner Link....4-12
Adding a Partner Link for the File Adapter....4-22
Configuring the Invoke Activities....4-26
Configuring the Assign Activity....4-32
Run-Time Tasks for XML Gateway Inbound Messaging....4-36
Deploying the BPEL Process....4-36
Testing the BPEL Process....4-38
Verifying Records in Oracle Applications....4-40
Design-Time Task for XML Gateway Outbound Messaging....4-44
Creating a New BPEL Project....4-47
Adding a Partner Link....4-51
Adding a Receive Activity....4-56
Adding a Partner Link for File Adapter....4-59
Adding an Invoke Activity....4-63
Adding an Assign Activity....4-65
Run-Time Task for XML Gateway Outbound Messaging....4-68
Deploying the BPEL Process....4-68
Testing the BPEL Process....4-71
Troubleshooting and Debugging....4-76

5 Using Business Events

Overview of Business Events....5-1
Business Events Concepts....5-3
Design-Time Tasks for Outbound Business Events....5-4
Creating a New BPEL Project....5-5
Creating a Partner Link....5-10
Configuring the Receive Activity....5-19
Adding a Partner Link for the File Adapter....5-21
Configuring the Invoke Activity....5-27
Configuring the Assign Activity....5-29
Run-Time Tasks for Outbound Business Events....5-33
Deploying the BPEL Process....5-34
Testing the BPEL Process....5-36
Troubleshooting and Debugging....5-44

6 Using Concurrent Programs

Overview of Concurrent Programs 6-1
Design-Time Tasks for Concurrent Programs 6-2
Creating a New BPEL Project 6-3
Adding Partner Links 6-7
Adding Partner Links for File Adapter 6-30
Configuring the Invoke Activities 6-41
Configuring Assign Activities 6-48
Run-Time Tasks for Concurrent Programs 6-54
Deploying the BPEL Process 6-54
Testing the BPEL Process 6-57
Verifying Records in Oracle Applications 6-60
Troubleshooting and Debugging 6-61

7 Using Interface Tables and Views

Overview of Interface Tables and Views 7-2
Design-Time Tasks for Interface Tables 7-3
Creating a New BPEL Project 7-3
Adding a Partner Link 7-6
Adding a Partner Link for File Adapter 7-23
Configuring the Invoke Activities 7-28
Configuring the Assign Activity 7-31
Run-Time Tasks for Interface Tables 7-35
Deploying the BPEL Process 7-35
Testing the BPEL Process 7-36
Design-Time Tasks for Views 7-39
Creating a New BPEL Project 7-40
Adding a Partner Link 7-44
Adding a Partner Link for File Adapter 7-58
Configuring the Invoke Activities 7-64
Configuring the Assign Activity 7-68
Run-Time Tasks for Views 7-72
Deploying the BPEL Process 7-73
Testing the BPEL Process 7-74

8 Using PL/SQL APIs

Overview of PL/SQL APIs 8-1
Design-Time Tasks for PL/SQL APIs 8-2
Creating a New BPEL Project 8-4

Adding Partner Links ... 8-8
Adding a Partner Link for File Adapter ... 8-22
Defining Wrapper APIs ... 8-27
Declaring Parameters with a DEFAULT Clause ... 8-30
Configuring the Invoke Activities ... 8-33
Configuring the Transform Activity ... 8-38
Configuring an Assign Activity ... 8-40
Run-Time Tasks for PL/SQL APIs ... 8-43
Deploying the BPEL Process ... 8-43
Testing the BPEL Process ... 8-45
Troubleshooting and Debugging ... 8-49

9 Using e-Commerce Gateway

Overview of e-Commerce Gateway Integration ... 9-1
Design-Time Tasks for e-Commerce Gateway ... 9-2
Creating a New BPEL Project ... 9-3
Adding a Partner Link ... 9-8
Adding a Partner Link for File Adapter ... 9-15
Configuring the Invoke Activities ... 9-20
Configuring the Assign Activity ... 9-25
Run-Time Tasks for e-Commerce Gateway ... 9-30
Deploying the BPEL Process ... 9-30
Testing the BPEL Process ... 9-32
Verifying Records in Oracle Applications ... 9-34

A WSDL Definition File and Connection Information Details

WSDL Definition File ... A-1
Configuring Connection Information ... A-2

B Troubleshooting and Workarounds

General Issues and Workarounds ... B-1

Index

Send Us Your Comments

Oracle Fusion Middleware Adapter for Oracle Applications User's Guide, 11*g* Release 1 (11.1.1.4.0)

Part No. E10537-03

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document. Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

Note: Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the new Oracle E-Business Suite Release Online Documentation CD available on My Oracle Support and www.oracle.com. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: appsdoc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at www.oracle.com.

Preface

Intended Audience

Welcome to the *Oracle Fusion Middleware Adapter for Oracle Applications User's Guide*, 11*g* Release 1 (11.1.1.4.0)

This documentation is written for the technical consultants, implementers and system integration consultants who use Oracle Fusion Middleware Adapter for Oracle Applications.

This guide assumes you have a working knowledge of the following:

- The principles and customary practices of your business area.
- Oracle E-Business Suite.
- Oracle BPEL Process Manager.
- Oracle JDeveloper.
- Oracle Database, Oracle Fusion Middleware, Oracle WebLogic Server, and PL/SQL technology.
- Oracle E-Business Suite Integration Interfaces.
- Oracle integration technologies, including Web services, WSDL, XML Gateway, EDI Gateway, and the Business Event System.
- B2B, A2A and BP integrations.

If you have never used these products, Oracle suggests that you attend training classes available through Oracle University.

See Related Information Sources on page xi for more Oracle E-Business Suite product information.

Deaf/Hard of Hearing Access to Oracle Support Services

To reach Oracle Support Services, use a telecommunications relay service (TRS) to call Oracle Support at 1.800.223.1711. An Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process. Information about TRS is available at http://www.fcc.gov/cgb/consumerfacts/trs.html, and a list of phone numbers is available at http://www.fcc.gov/cgb/dro/trsphonebk.html.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. 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.

Structure

1 Introduction to Adapter for Oracle Applications
2 Adapter for Oracle Applications Features
3 Adapter for Oracle Applications Concepts
4 Using XML Gateway
5 Using Business Events
6 Using Concurrent Programs
7 Using Interface Tables and Views
8 Using PL/SQL APIs
9 Using e-Commerce Gateway

A WSDL Definition File and Connection Information Details
B Troubleshooting and Workarounds

Related Information Sources

You can choose from many sources of information, including online documentation, training, and support services, to increase your knowledge and understanding of Oracle Fusion Middleware Adapter for Oracle Applications.

Documentation

You may want to refer to other Oracle Fusion Middleware guides when you set up and use Oracle Fusion Middleware Adapter for Oracle Applications. You can read the guides online by reading from the Oracle Fusion Middleware or Oracle Database Documentation Library CD included in your media pack, or on the Oracle Technology Network (OTN) [http://www.oracle.com/technology/documentation/appserver.html]. If you require printed guides, you can purchase them from the Oracle Store [http://oraclestore.oracle.com].

To download free release notes, installation documentation, white papers, or other collateral, please visit the main OTN page [http://www.oracle.com/technology/]. You must register online before using OTN; registration is free and can be done at the OTN registration page [http://www.oracle.com/technology/membership/].

Training

Oracle offers a complete set of training courses to help you and your staff master Oracle Fusion Middleware 11*g* and reach full productivity quickly. These courses are organized into functional learning paths, so you take only those courses appropriate to your job or area of responsibility.

You have a choice of educational environments. You can attend courses offered by Oracle University at any one of our many Education Centers, you can arrange for our trainers to teach at your facility, or you can use Oracle Learning Network (OLN), Oracle University's online education utility.

> **Tip:** For information about upcoming instructor-led training, please refer to Oracle University's course offerings [http://education.oracle.com/].

In addition, Oracle training professionals can tailor standard courses or develop custom courses to meet your needs. For example, you may want to use your organization's structure, terminology, and data as examples in a customized training session delivered at your own facility.

Support

From on-site support to central support, our team of experienced professionals provides the help and information you need to keep Oracle Fusion Middleware 11*g* working for you. This team includes your Technical Representative, Account Manager, and Oracle's

large staff of consultants and support specialists, with expertise in your business area, managing an Oracle Database, and your hardware and software environment.

Do Not Use Database Tools to Modify Oracle E-Business Suite Data

Oracle STRONGLY RECOMMENDS that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle E-Business Suite data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Oracle E-Business Suite data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle E-Business Suite tables are interrelated, any change you make using an Oracle E-Business Suite form can update many tables at once. But when you modify Oracle E-Business Suite data using anything other than Oracle E-Business Suite, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle E-Business Suite.

When you use Oracle E-Business Suite to modify your data, Oracle E-Business Suite automatically checks that your changes are valid. Oracle E-Business Suite also keeps track of who changes information. If you enter information into database tables using database tools, you may store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

1

Introduction to Adapter for Oracle Applications

This chapter covers the following topics:

- Overview of Adapter for Oracle Applications
- New Features in This Release

Overview of Adapter for 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 value across front-office and back-office functions
- Access to current, accurate, and consistent data

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 non-Oracle applications, you use Oracle Fusion Middleware Adapter for Oracle Applications.

Adapter for Oracle Applications not only provides comprehensive, bidirectional, multimodal, synchronous, and asynchronous connectivity to Oracle Applications, but also supports for all modules of Oracle Applications in Release 12 and Release 11*i* including custom integration interfaces in various versions of Oracle E-Business Suite.

> **Important:** Please note that Adapter for Oracle Applications is also informally known as Oracle E-Business Suite Adapter.

The support for various versions of Oracle E-Business Suite has the following conditions:

- Adapter for Oracle Applications supports only those versions of Oracle E-Business Suite Release 11*i* which work with OWF.G.Rollup 7 applied.
- Adapter for Oracle Applications version 10.1.3.3 onwards supports Oracle E-Business Suite Release 12.0.
- To enable the native Oracle E-Business Suite connectivity using J2EE data sources feature, the minimum requirement for Oracle E-Business Suite Release 11*i* is 11*i*.ATG_PF.H.Delta.6 (RUP6) and for Oracle E-Business Suite Release 12 is Release 12.0.4.

  See "Oracle Fusion Middleware Adapter for Oracle Applications, Release 11*g*", My Oracle Support (formerly Oracle*MetaLink*) Knowledge Document 787637.1 for details.

Major Features

Adapter for Oracle Applications provides the following features:

- It leverages the Integration Repository to provide the information from the source of truth on integration.
- It supports the widest range of integration interface types. They are PL/SQL APIs, Business Events, Open Interface Tables, Concurrent Programs, XML Gateway Interfaces, e-Commerce Gateway Interface, and Interface Views.
- It generates adapter metadata as WSDL files with J2CA extension.

  Note: For more information, see *Oracle Fusion Middleware User's Guide for Technology Adapters*.

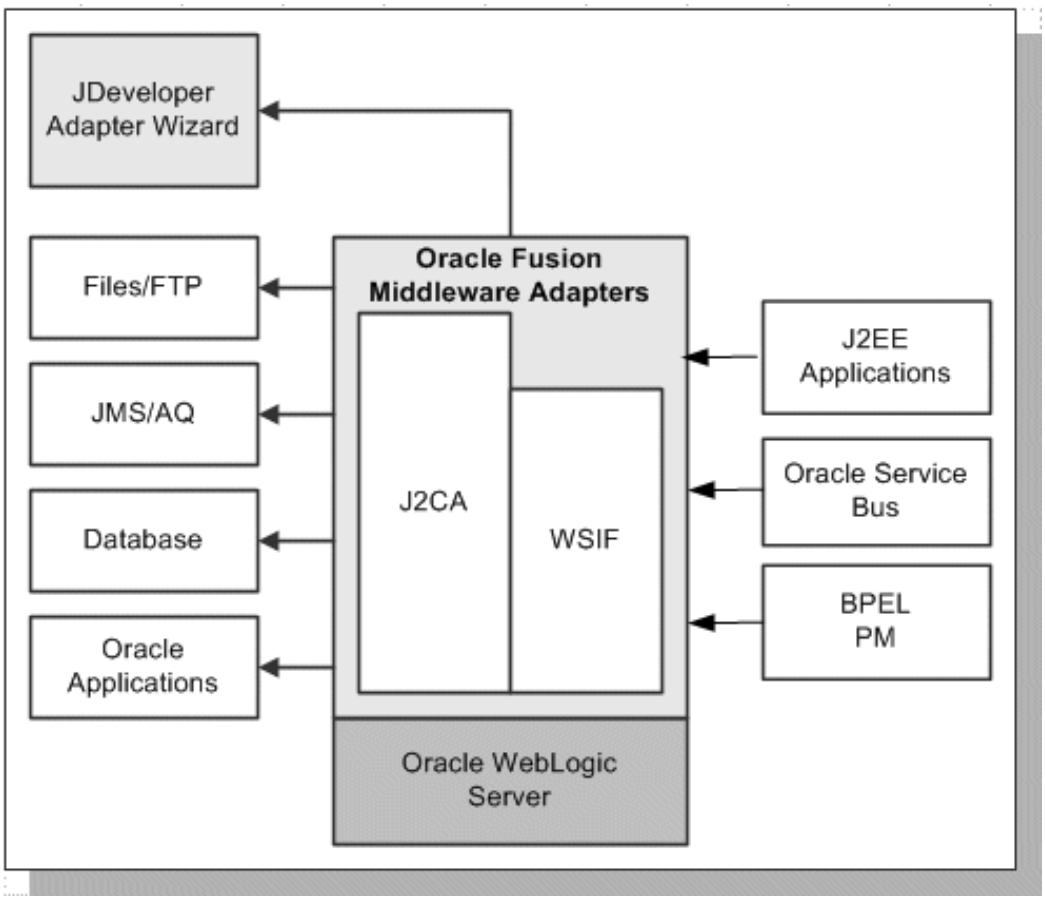
- It works under the securely configured connection between Oracle Applications and Oracle Fusion Middleware using just the FND User name and password for authentication.
- It leverages and supports Oracle User Management function security feature to allow only authorized users to access and execute APIs that they are exposed as Web services to update Oracle Applications.
- It implicitly takes care of application context without bothering about the complexities of invoking the same explicitly.

- It supports multiple languages and multiple organization access control (MOAC) setups based on the concept of applications context.
- It uses a JDeveloper based design-time tool for dynamically browsing the Oracle Applications interface and configuring the adapter metadata. The design-time tasks are wizard driven, user-friendly, and intuitive providing superior user experiences.
- It provides the global transaction control support implementing two-phase commit by leveraging the underlying JCA standards compliant framework.
- It supports multiple versions of Oracle E-Business Suite from the same instance of Adapter at design time.

Architecture

Adapter for Oracle Applications is based on J2CA 1.0 standards and deployed as a resource adapter within the Oracle WebLogic Server container. The architecture of Adapter for Oracle Applications is similar to the architecture of technology adapters.

Adapter for Oracle Applications Architecture



For more information on technology adapters, see *Oracle Fusion Middleware User's Guide for Technology Adapters*.

Installing Adapter for Oracle Applications

Adapter for Oracle Applications and Oracle JCA Adapters are available as part of the Oracle Fusion Middleware install. In addition, these adapters support both Oracle WebLogic Server and middle tier deployments.

For more information, see *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite*.

Integration with Oracle BPEL Process Manager

Based on the service-oriented architecture (SOA), Oracle BPEL Process Manager (BPEL PM) provides a comprehensive solution for creating, deploying, and managing Oracle BPEL Process Manager business processes.

Adapter for Oracle Applications can easily expose public integration interface within

Oracle E-Business Suite as standard Web services. These services can be created and configured in the Oracle JDeveloper at design time using BPEL Designer. At run time, Oracle E-Business Suite integration flows are deployed in the Oracle WebLogic Server for execution of the services to complete the integration.

Design Time

The Oracle JDeveloper BPEL Designer, a graphical drag and drop environment, is used to design BPEL-based process flows and Web services orchestration.

When you create a partner link in JDeveloper BPEL Designer, the Adapter Configuration Wizard starts which enables you to select and configure the Adapters for Oracle Applications or other adapters. With proper database and service connection setups, you can select an interface in or out from Oracle E-Business Suite and add the XML schema. When configuration is complete, the wizard generates a WSDL file corresponding to the XML schema for the partner link.

Additional process activities are added to the BPEL process if necessary to assign parameters and invoke the service.

Run Time

Oracle Adapter for Oracle Applications is based on the J2CA 1.5 specification, and a BPEL process is deployed on the 11g run time on the Oracle WebLogic Server. A JCA Binding Component acts as the glue layer that integrates the standard J2CA 1.5 resource adapter with the Oracle BPEL Process Manager during run time. The JCA Binding Component acts as a pseudo J2CA 1.5 container.

> **Note:** Only the JCA 1.5 integration allows the BPEL PM to receive inbound events (from EIS to J2EE/BPEL PM). The Oracle BPEL Process Manager acts as a pseudo JCA 1.5 container and implements the JCA 1.5-specific System Contracts. The JCA 1.5 resource adapter and the BPEL PM instance must be deployed in the same Oracle WebLogic Server container.

The Web service invocation launched by the BPEL Invoke activity is converted to a J2CA CCI (Common Client Interface) outbound interaction, and the J2CA response is converted back to a Web service response. This end-to-end invocation is synchronous.

Testing the BPEL Process at Run Time

After deploying the BPEL process, you should validate the design by testing the deployed BPEL process to test the interface integration.

For detailed design-time and run-time tasks for each integration interface, see the individual interface chapter explained later in this book.

Integration with Oracle WebLogic Server

Oracle WebLogic Server is a scalable, enterprise-ready Java Platform, Enterprise Edition (Java EE) application server. Its infrastructure enables enterprises to deploy

mission-critical applications in a robust, secure, highly available, and scalable environment and is an ideal foundation for building applications based on service-oriented architectures (SOA). SOA is a design methodology aimed at maximizing the reuse of application services.

In addition, Oracle WebLogic Server consists of a J2CA container for hosting J2CA resource adapters. J2CA defines standard Java interfaces for simplifying the integration of a J2EE server with various back-end applications. All client applications run within the Oracle WebLogic Server environment.

Design Time

Oracle JDeveloper is used to create Web services represented in WSDL files and XML Schema Definition (XSD) files for the adapter request-response service.

The Oracle WebLogic Server clients use these XSD files during run time for calling the J2CA outbound interaction.

Run Time

Oracle Adapter for Oracle Applications is based on the J2CA 1.5 specification, but is deployed as the J2CA 1.5 resource adapter within the Oracle WebLogic Server container. The J2CA 1.5 specification addresses the life-cycle management, message-inflow (for Adapter Event publish), and work management contracts.

For more information about using Oracle WebLogic Server with Oracle JDeveloper, see the Using WebLogic Server with Oracle JDeveloper section, *Oracle Fusion Middleware Installation Guide for Oracle JDeveloper*.

New Features in This Release

This section describes the new features that have been added in Oracle Fusion Middleware Adapter for Oracle Applications 11*g* Release 1 (11.1.1.4.0).

Logging for Oracle Fusion Middleware Adapter for Oracle Applications

Oracle Fusion Middleware Adapter for Oracle Applications implements the Oracle SOA Suite's logging framework to write the diagnostic log files in text format. Therefore, whenever the Oracle E-Business Suite services are invoked using Oracle Adapter for Oracle Applications, the log messages are recorded which can be accessed by system administrator. This enriches the problem identification mechanism to track any issues during service invocations at run time for Oracle Adapter for Oracle Applications.

With proper log-level configuration in Oracle Enterprise Manager Fusion Middleware Control Console, you can view Oracle Diagnostic Logging (ODL) level log files written to a single file at run time for Oracle Adapter for Oracle Applications.

For more information on viewing adapter logs, see Logging, page 3-20.

2

Adapter for Oracle Applications Features

Overview

Adapter for Oracle Applications enables you to orchestrate discrete data into a meaningful business process and creates Web services for various interface types within Oracle E-Business Suite. It plays the role of service provider for Oracle E-Business Suite to allow seamless integration between business partners, processes, applications, and end users in heterogeneous environment.

Adapter for Oracle Applications provides the following features which are further discussed in this chapter:

- Support for Various Integration Interface Types, page 2-1.
- Support for Oracle Integration Repository, page 2-3
- Support for Custom Integration Interfaces in Various Versions of Oracle E-Business Suite, page 2-4

Support for Various Integration Interface Types

Adapter for Oracle Applications acts as a highly flexible integration interface for Oracle Applications. It supports the following interface types for integrating with Oracle Applications:

- **PL/SQL APIs**

  These APIs enable you to insert and update data in Oracle Applications using PL/SQL.

- **Business Events**

  A business event is an occurrence in an internet application that might be significant to other objects in a system or to external agents. An example of a

business event can be the creation of a new sales order or changes to an existing order.

Oracle Workflow uses the Business Event System that leverages the Oracle Advanced Queuing (AQ) infrastructure to communicate and manage business events between systems. The Business Event System consists of an Event Manager and workflow process event activities. The Event Manager lets you register subscriptions to significant events; event activities representing business events within workflow processes let you model complex business flows or logics within workflow processes.

When a local event occurs, the subscribing code is executed in the same transaction as the code that raised the event. Subscription processing can include executing custom code on the event information, sending event information to a workflow process, and sending event information to other queues or systems.

- **Open Interface Tables**

  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.

- **Concurrent Programs**

  Concurrent programs enable you to move data from interface tables to base tables or execute any application logic.

- **Oracle XML Gateway**

  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 interface that exposes OAGIS-formatted XML documents for commonly used Oracle Application business objects and business interfaces. XML Gateway integrates with interface tables, Oracle Workflow Business Event System (BES), and interface views to insert and retrieve data from Oracle Applications. It maps the underlying table data to XML and back.

- **Oracle e-Commerce (EDI) Gateway**

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

- **Interface Views**

  Interface views help you to retrieve data from Oracle Applications using the application tables.

Please note that Adapter for Oracle Applications also supports the following custom integration interface types that are exposed by the Oracle Applications Module Browser, not by Oracle Integration Repository:

- Customized PL/SQL APIs
- Customized Business Events

  Note: Business events integration interface type is also exposed by Oracle Applications Module Browser, not by Oracle Integration Repository.

- Customized XML Gateway Maps

Support for Oracle Integration Repository

Oracle Integration Repository, an integral part of Oracle E-Business Suite, is a prebuilt catalog of information about the numerous public integration interfaces delivered with Oracle Applications. It provides a comprehensive view of the integration interfaces with details for Oracle E-Business Suite.

Oracle Integration Repository can only provide information about an integration interface that has been specifically annotated by the developer to make it public. Adapter for Oracle Applications takes advantage of the annotations that have already been created to make the following integration interface types visible in the Oracle Applications Module Browser:

- XML Gateway message maps
- PL/SQL APIs
- Concurrent programs
- Open Interface tables
- Interface views
- e-Commerce Gateway EDI messages

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

For more information about Oracle Integration Repository, see the Navigating Through Oracle Integration Repository section, *Oracle E-Business Suite Integrated SOA Gateway User's Guide*. This guide is part of the Oracle Applications documentation library. Oracle Applications documentation can be accessed with the following link:

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

Note: Oracle Integration Repository is integral part of Oracle

E-Business Suite Release 12.0 onwards; however, it is available as hosted environment for the Release 11.5.10 version at `http://irep.oracle.com`.

Support for Custom Integration Interfaces in Various Versions of Oracle E-Business Suite

Oracle Adapter for Oracle Applications leverages Integration Repository for Oracle E-Business Suite Release 11.5.10 and Release 12 as the source of truth for the integration content. However, the implementation is based on the version of Oracle E-Business Suite. For pre-Release 11.5.10 instances, Oracle Adapter for Oracle Applications connects directly to the application database for information on integration interfaces. Adapter for Oracle Applications also supports selecting custom integration interfaces and the design-time navigation steps to reach to these custom interfaces depending on the following versions of Oracle E-Business Suite:

- Release 12, page 2-5
- Release 11.5.10, page 2-5
- Pre-Release 11.5.10, page 2-6

Important: Please note that the support for various versions of Oracle E-Business Suite has the following conditions:

- Adapter for Oracle Applications supports only those versions of Oracle E-Business Suite Release 11*i* which work with OWF.G.Rollup 7 applied.
- Adapter for Oracle Applications version 10.1.3.3 onwards supports Oracle E-Business Suite Release 12.0.
- To enable the native Oracle E-Business Suite connectivity using J2EE data sources feature, the minimum requirement for Oracle E-Business Suite Release 11*i* is 11*i*.ATG_PF.H.Delta.6 (RUP6) and for Oracle E-Business Suite Release 12 is Release 12.0.4.

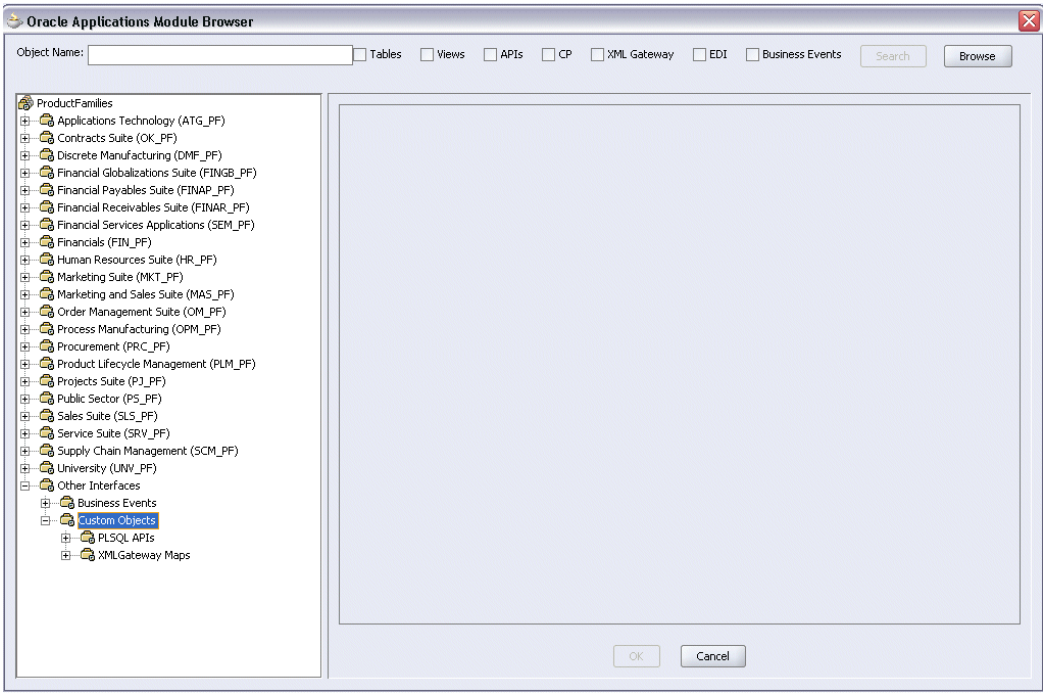
  See "Oracle Fusion Middleware Adapter for Oracle Applications, Release 11*g*", My Oracle Support Knowledge Document 787637.1 for details.

From the business service creation and run-time perspectives, Adapter for Oracle Applications supports customized PL/SQL APIs as far as the packages are available in the APPS schema. The Oracle Applications Module Browser can expose these customized PL/SQL APIs for integration purposes during the design time.

Support for Oracle E-Business Suite Release 12

From Release 12, Oracle Integration Repository is shipped as part of the Oracle E-Business Suite which enables Adapter for Oracle Applications to directly connect to the live database of Oracle Integration Repository querying for the public interfaces and then displaying the list of customized PL/SQL APIs under the `Other Interfaces` node in the Oracle Applications Module Browser.

Supporting Custom Integration Interfaces in Release 12



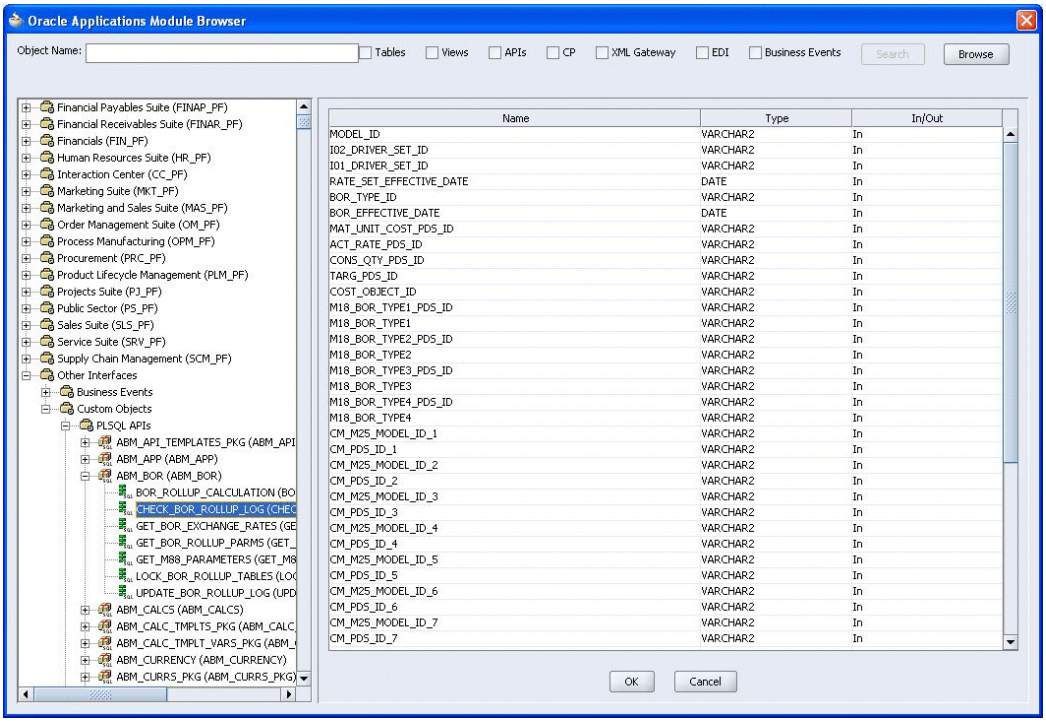
Please note that Adapter for Oracle Applications allows you to extract the Integration Repository data file from the live database you connect to Oracle Applications and create a local copy of the Integration Repository data file in your workplace. Next time when you look for public interfaces, the system can retrieve data from the cache in your workplace.

For detailed information about connecting to Oracle E-Business Suite Release 12, please refer to the Creating a Partner Link or Adding a Partner Link design-time task for each integration interface.

Support for Oracle E-Business Suite Release 11.5.10

To support the Release 11.5.10 version of Oracle E-Business Suite, Adapter for Oracle Applications provides the Integration Repository data file bundled as part of the product in xml format. At the design time, Adapter for Oracle Applications queries public interfaces from the native XML data file of the Integration Repository located in the Adapter and displays the list of custom integration interfaces under the `Other Interfaces` node in the Oracle Applications Module Browser.

Supporting Custom Integration Interfaces in Release 11.5.10



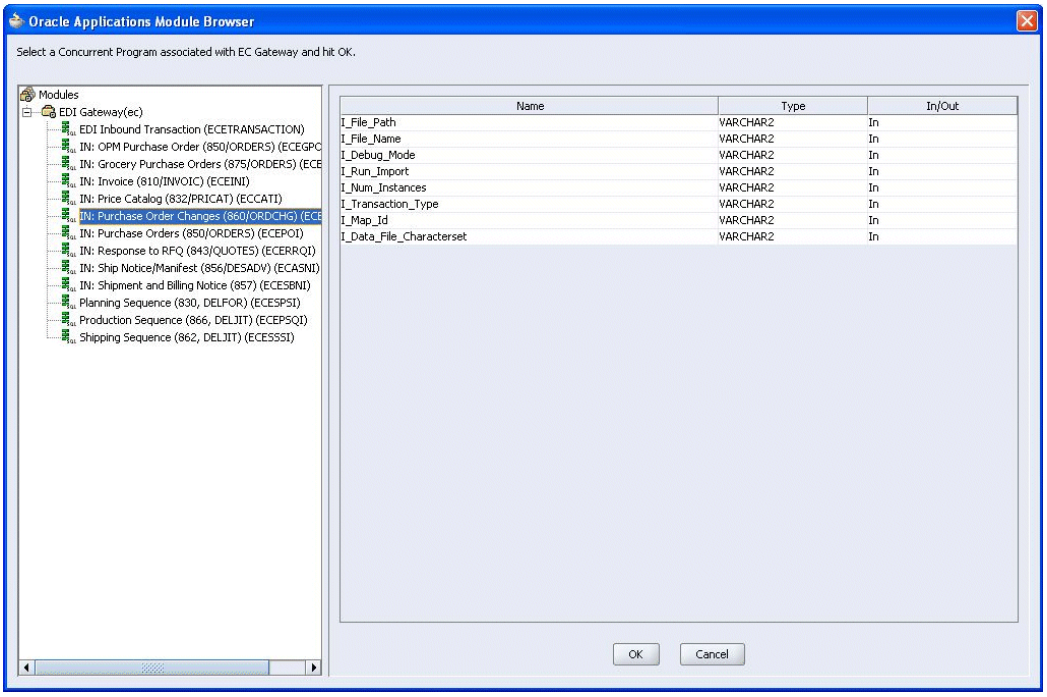
Support for Oracle E-Business Suite Pre-Release 11.5.10

To support the pre-Release 11.5.10 versions of Oracle E-Business Suite, Oracle Adapter for Oracle Applications connects to the live application database for the integration information on all interface types. Since there is no differentiation between public, private, and customized PL/SQL APIs in the pre-Release 11.5.10 versions of Oracle E-Business Suite, Adapter for Oracle Applications displays them all under the node of each module through Oracle Applications Module Browser.

Before making a selection from the browser for the pre-Release 11.5.10, you must select an interface type you want to use in the Adapter Configuration Wizard. All interfaces of your selected type will be displayed in the browser.

For example, you will find a list of concurrent programs associated with e-Commerce (EDI) Gateway displayed in the Oracle Application Module Browser as follows if the EDI Gateway interface type is selected.

Supporting Custom Integration Interfaces in pre-Release 11.5.10



When you make a selection through the module browser at design time, Adapter for Oracle Applications validates your selected API against the database. If it exists in the database for a particular version of your instance, then the associated WSDL file will be generated successfully.

3

Adapter for Oracle Applications Concepts

This chapter covers the following topics:

- Understanding Applications Context
- Understanding Adapter for Oracle Applications Security
- Logging
- Secured Connection Between Oracle E-Business Suite and Oracle Fusion Middleware SOA Suite Using J2EE Data Source Implementation
- Understanding the Oracle Applications Module Browser

Understanding Applications Context

Applications context is required for secured transaction processing into and out of Oracle Applications.

Applications context is a combination of **Username**, **Responsibility**, **Responsibility Application**, **NLS Language**, **Security Group**, and **Organization ID**. These elements are used in passing values that may be required in a business activity or to complete a BPEL process.

To understand applications context, you need to first understand how Organization ID and multiple organizations are related.

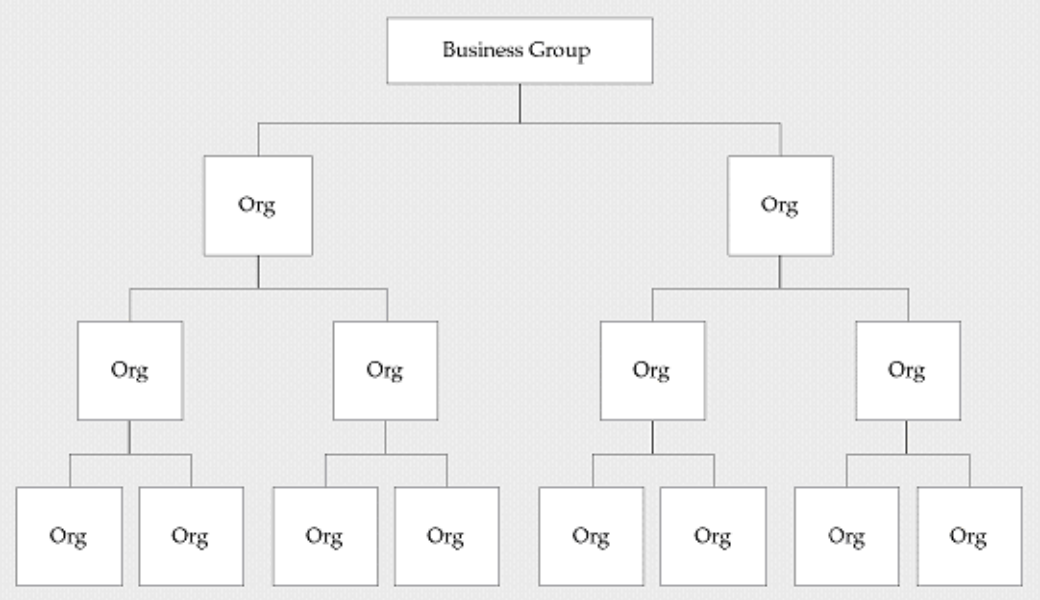
Applications Context in Multiple Organizations

You can define multiple organizations and the relationships between them in a single installation of Oracle Applications. These organizations can be sets of books, business groups, legal entities, operating units, or inventory organizations.

Multilevel organization hierarchies can be defined with a business group at the top of each hierarchy. When you define new organizations, they are automatically assigned to the business group associated with your current session. Each organization is part of a business group. The business group is usually the top box on an enterprise organization

chart.

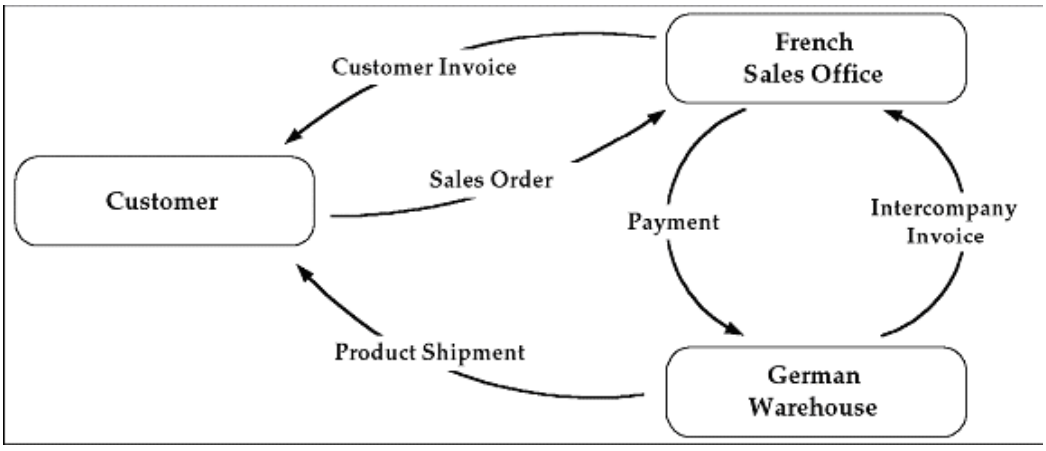
Business Group Hierarchy



Example of a Multiple-Organization Setup

Using the accounting, distribution, and materials management functions in Oracle Applications, you define the relationships among inventory organizations, operating units, legal entities, and sets of books to create a multilevel company structure, as shown in the following diagram.

A Multiple-Organization Transaction

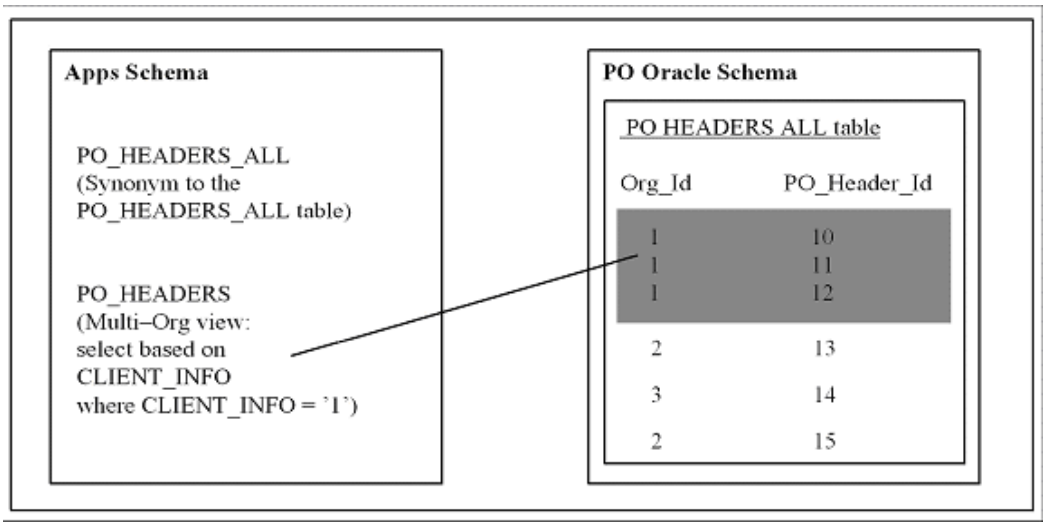


Consider two different organizations in your company: One is a French sales office and the other is a German warehouse. There is a sales order transaction with the customer, and this illustrates how the entire Order-to-Deliver process would work:

1. The customer places a sales order with the French sales office.
2. The German warehouse delivers the product shipment to the customer.
3. The German warehouse issues an inter-company invoice to the French sales office.
4. The French sales office makes the inter-company payment to the German warehouse.
5. The French sales office sends the customer invoice to the customer.
6. The customer makes payment to the French sales office.

The database architecture is the same for a multiple-organization and non-multiple-organization installation, and uses the standard install tools feature that automatically creates synonyms in the APPS schema for each base product table and defines these synonyms with the same name as the base product tables. For example, the PO Oracle schema has a table named PO_HEADERS_ALL and the APPS schema has a corresponding synonym of the same name, PO_HEADERS_ALL. The PO_HEADERS_ALL synonym can be used to access unpartitioned data.

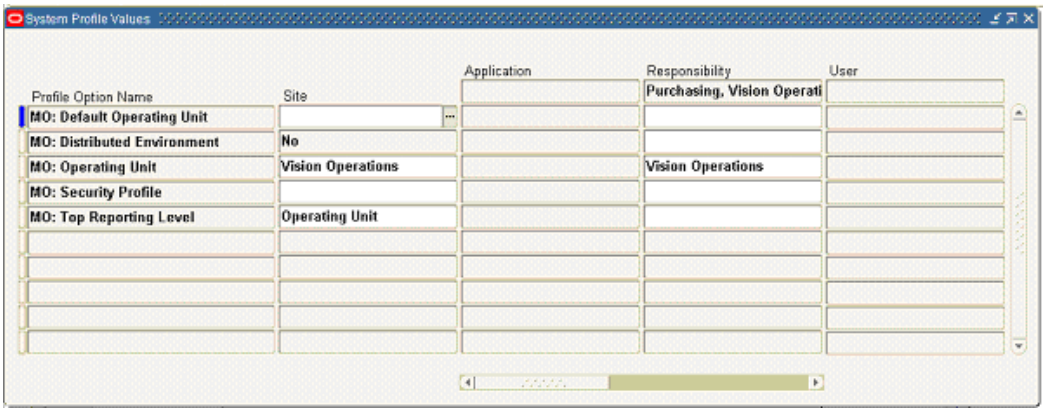
Schema Synonyms



Multi-Organization Access Control (MOAC) Security by Operating Units

While setting up the system profile values, the username and responsibility are tied up with the organization or operating units.

Multiple-Organization System Profiles

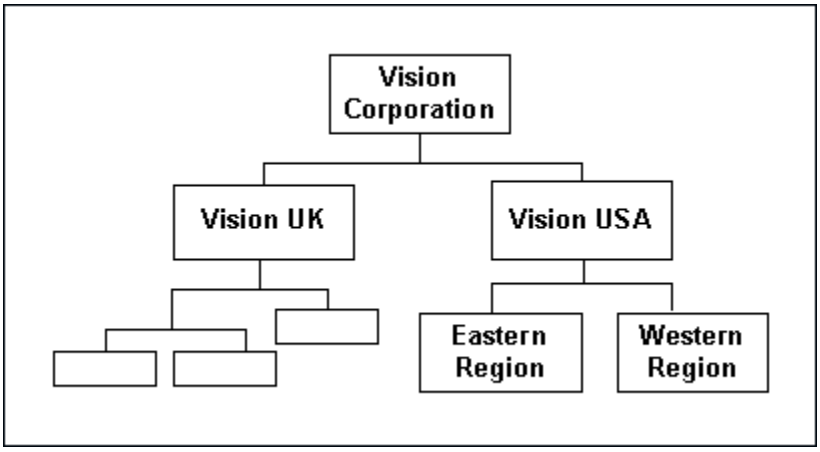


To have a secured way for users to only access or report on data for the operating units they have access to, Adapter for Oracle Applications uses the MOAC security feature to determine the operating unit access privileges and derive the Organization ID based on relevant profile values.

With MOAC, a system administrator can predefine the scope of access privileges as a security profile, and then use the profile option *MO: Security Profile* to associate the security profile with a responsibility. By using this approach, multiple operating units are associated with a security profile and that security profile is then assigned to a responsibility. Therefore, through the access control of security profiles, users can access data in multiple operating units without changing responsibility.

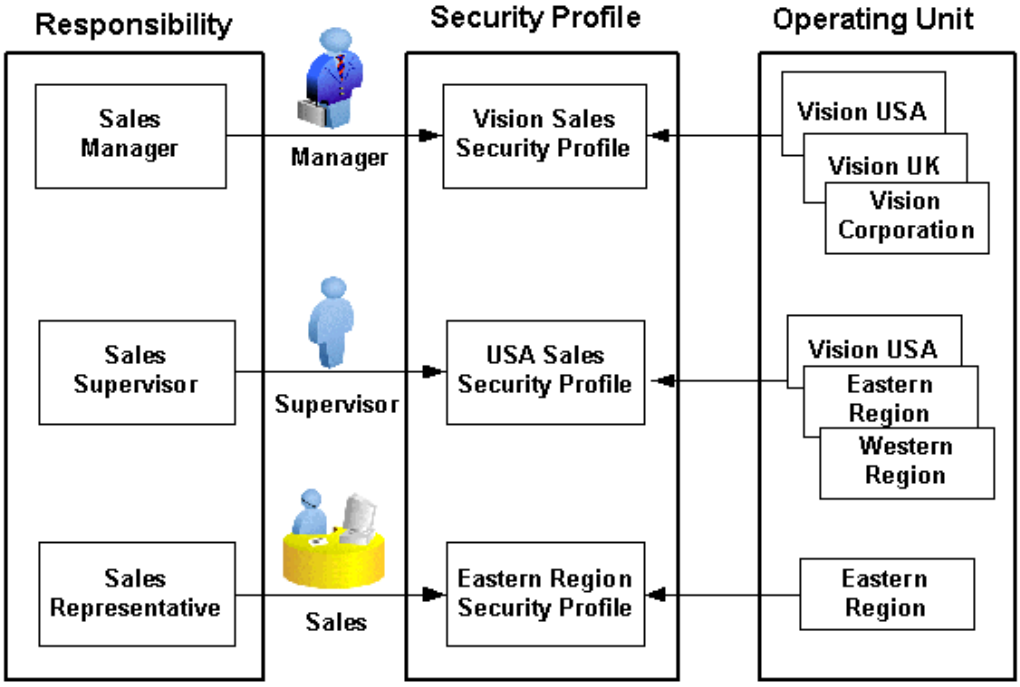
Security profiles are defined based on organization hierarchies. For example, a sales company consists of USA and UK operating units; the USA operating unit has Western Region Sales and East Region Sales. Sales managers are responsible for both USA and UK sales, supervisors are responsible for either USA or UK, and sales representatives are only responsible for their designated sales regions. The Sales organization hierarchy can be illustrated as follows:

Sales Organization Hierarchy



To secure sales data within the company, relevant operating units can be associated with predefined security profiles. For example, all sales data access privileges are grouped into the Vision Sales security profile. A USA Sales security profile is created for USA related data, and a regional security profile is created for designated regional data. The system administrator can associate these security profiles containing multiple operating units with users through appropriate *responsibilities*. Therefore, sales supervisors can easily access sales data in the Eastern or Western region without changing their responsibilities. The following diagram illustrates the relationship between security profiles, responsibilities, and operating units for this sales company:

Relationship Diagram Between Security Profiles, Responsibilities, and Operating Units

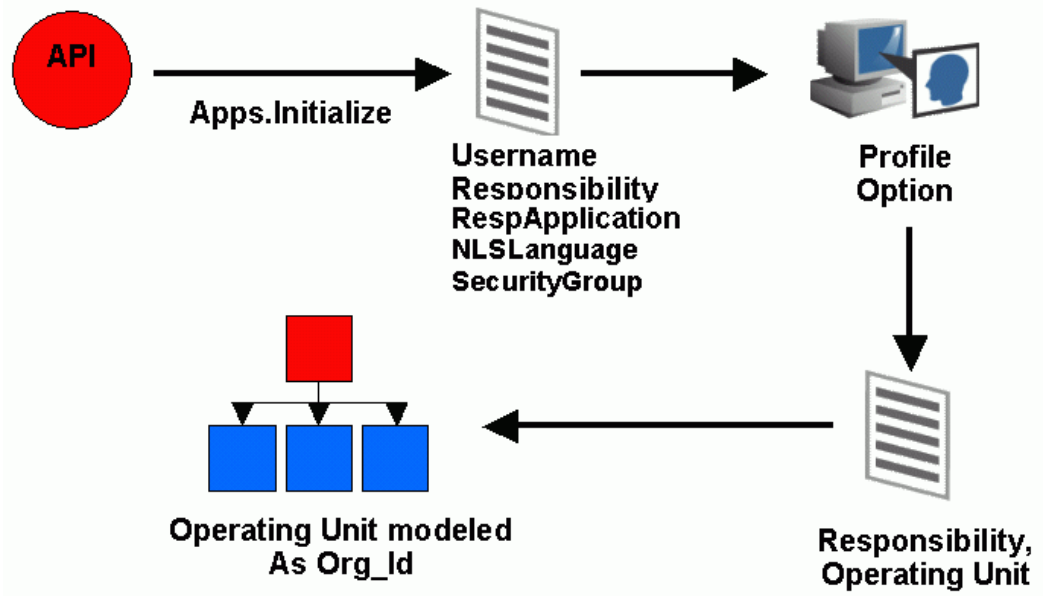


Responsibility Determines Operating Units

Because responsibilities are associated with security profiles that linked to operating units, your responsibility is the key in determining which operating units you will have the access privileges.

The following diagram illustrates how Oracle Applications use the profile options in a multi-organization environment:

Building Applications Context for Multiple Organizations



1. When the system integrator runs, the process achieves the integration with Oracle Applications using PL/SQL APIs.

2. Applications context is set, taking into account the values passed for the header properties Username, Responsibility, Responsibility Application, Security Group, and NLS Language. If the values for the new header properties Responsibility Application, Security Group, and NLS Language are not passed, context information will be determined based on Username and Responsibility.

3. With these parameters, a lookup on all System Profile Values assigned to that responsibility is done to determine the Operating Unit within a multi-organization environment.

4. The Operating Unit is modeled as Organization ID derived from the security profile value.

5. The data is read and written into the Oracle Applications with the parameters of Username, Responsibility, Responsibility Application, Security Group, NLS Language, and Organization ID.

To integrate business processes or invoke BPEL processes, it is essential to propagate these applications context values or messages through a flexible mechanism that allows you to effectively set each individual context value if needed and pass them to complete a BPEL process and meet integration needs.

The following topics are discussed in this section:

- Supporting for Normalized Message Properties, page 3-8
- Supporting for Multiple Organization Setups, page 3-12
- Supporting for Multiple Languages, page 3-14

Supporting for Normalized Message Properties

To effectively set applications context values required in a BPEL process or to populate mandatory header variables for XML Gateway inbound transactions to be completed successfully, Adapter for Oracle Applications provides a flexible mechanism that allows each context value and header variable to be set and passed in the adapter user interface directly through the Invoke activity. This message normalization feature not only provides a flexible solution on header support, but also simplifies the design-time tasks without using an Assign activity to pass header values.

Setting Message Properties for Applications Context

The following header message properties are used in setting applications context required in a business activity or to complete a BPEL process:

- `jca.apps.Username`
- `jca.apps.Responsibility`
- `jca.apps.ORG_ID`
- `jca.apps.RespApplication`
- `jca.apps.SecurityGroup`
- `jca.apps.NLSLanguage`

> **Note:** Existing header property `jca.apps.Responsibility` used in the earlier releases can now take Responsibility Key as well as Responsibility Name as input. If the header property `jca.apps.NLSLanguage` is set, and Responsibility Name is passed, the value passed for `jca.apps.Responsibility` is expected to be in the same language. However, Responsibility Key as well as all other header properties are language independent.
>
> All these header properties would be used together to set the application context. Alternatively, passing just the Username and Responsibility would work as it did in the earlier releases.

In the case of a null or empty value, the default Username is `SYSADMIN`, the default Responsibility is `System Administrator`, the default Security Group Key is `Standard`, and the default NLS Language is `US`.

Since the *NLS Language* and *Organization ID* property values are included in message normalization for supporting applications context, Adapter for Oracle Applications also supports the following features based on the concept of application context:

- Supporting for Multiple Organization Setups, page 3-12
- Supporting for Multiple Languages, page 3-14

Setting Message Properties for XML Gateway Inbound Transactions

The following header message properties are used in setting XML Gateway information required for XML Gateway inbound/enqueue transactions:

- `jca.apps.ecx.TransactionType`
- `jca.apps.ecx.TransactionSubtype`
- `jca.apps.ecx.PartySiteId`
- `jca.apps.ecx.MessageType`
- `jca.apps.ecx.MessageStandard`
- `jca.apps.ecx.DocumentNumber`
- `jca.apps.ecx.ProtocolType`
- `jca.apps.ecx.ProtocolAddress`
- `jca.apps.ecx.Username`
- `jca.apps.ecx.Password`
- `jca.apps.ecx.Attribute1`
- `jca.apps.ecx.Attribute2`
- `jca.apps.ecx.Attribute3`
- `jca.apps.ecx.Attribute4`
- `jca.apps.ecx.Attribute5`
- `jca.apps.ecx.Payload`

Design-Time Tasks for Message Properties Support

Adapter for Oracle Applications uses the following procedures to complete the design-time tasks to support message normalization:

Note: At run time, the XML Gateway message properties should be

processed before invoking AQ Adapter.

1. Create a new BPEL project, page 4-9
2. Create a partner link, page 4-12
3. Configure an Invoke Activity, page 4-63

   This activity involves the following tasks:

   - *Configure basic information in the General tab:*

     Configure an **Invoke** activity by linking the activity to the partner link you just created. This opens the General tab in the Edit Invoke dialog box with Partner Link and Operation information populated.

     You can create an Input Variable and a Output Variable for the Invoke activity.

     For detailed instructions, see Configure basic information in the General tab, page 4-27

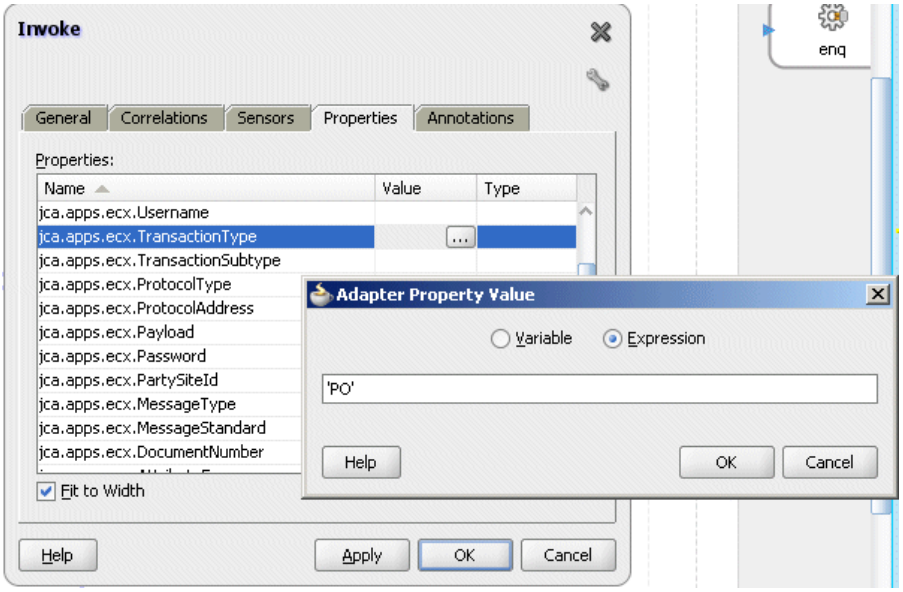
   - *Set the Header Message Properties in the Properties tab:*

     This is to set the necessary message properties for the following purposes:

     - To set XML Gateway header variables for an inbound transaction.

       For example, locate XML Gateway header property `jca.apps.ecx.TransactionType` first and then enter a value (such as `'PO'`) for the selected property.

Entering a Selected Property Value

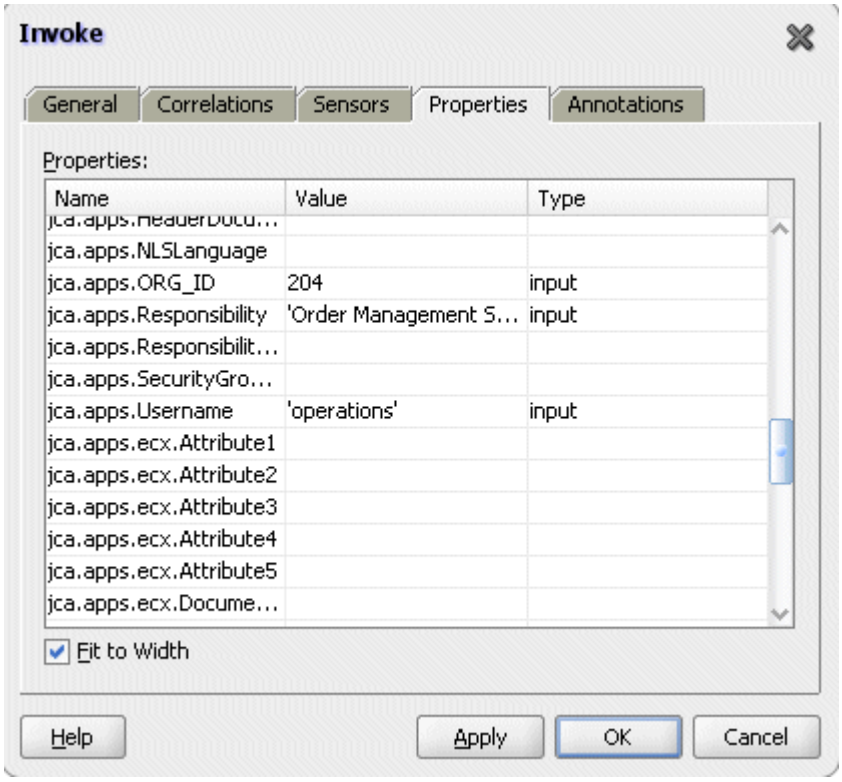


For detailed instructions, see Setting ECX Header Message Properties, page 4-30.

- To set applications context for Oracle Applications to identify the application user, responsibility, and the user's associated organization information.

  For example, locate the `jca.apps.Username` property from the property list and then enter `'OPERATIONS'` as the property value.

Header Property Values



For detailed instructions, see Setting the Header Properties for Application Context, page 6-46.

Supporting for Multiple Organization Setups

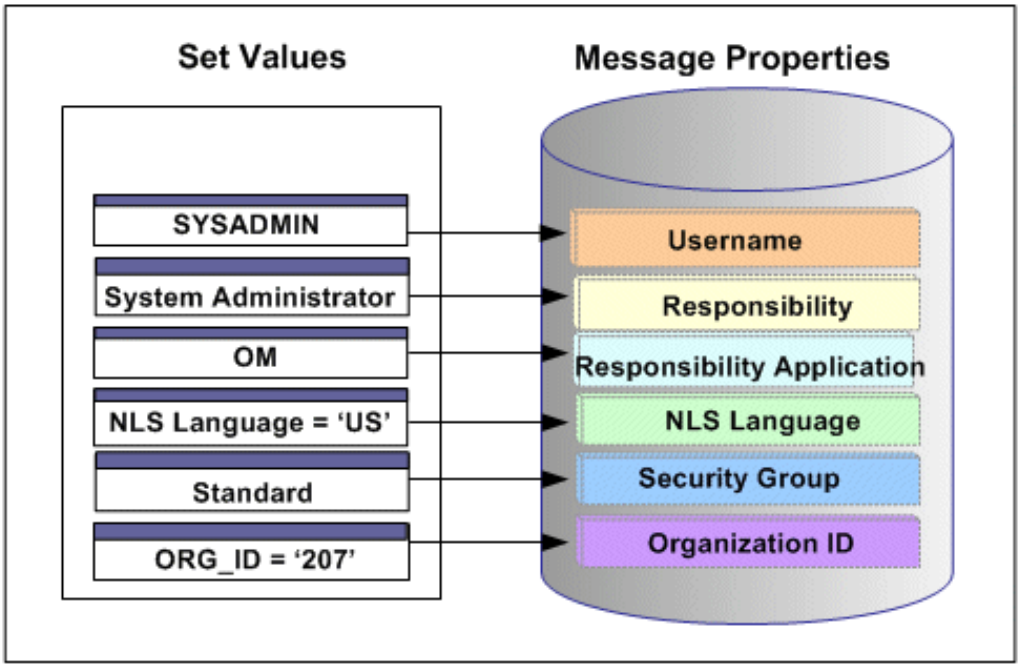
Instead of implicitly deriving organization information from a profile value during the Oracle Applications setups, based on message normalization feature, Adapter for Oracle Applications allows Organization ID information to be directly entered through the Properties tab of an Invoke activity during the design time to support multiple organization setups.

Adapter for Oracle Applications uses the header properties to include *Username, Responsibility, Responsibility Application, NLS Language, Security Group*, and *Organization ID*, the essential elements for applications context. Once you set the message properties contained in the header for a business activity through a PL/SQL API or an interface which requires the applications context to be set, these values in the header will be passed and used as an input to the rest of activities in the BPEL process.

Note: Integration interface types that require applications context to be

set are PL/SQL APIs, concurrent programs, and EDI programs.

Setting Header Message Properties



The advantage of having this header message properties mechanism in supporting multiple organization setups is that with only one single BPEL process, organization information can be easily placed into multiple organizations within the Oracle E-Business Suite if the Organization ID value has been specified in the header. While in the past, since Organization ID is implicitly derived from the profile value based on an application logon user's username and responsibility; therefore, only that associated organization for the invocation of the deployed BPEL process can be inserted.

With the example described earlier in the Multiple Organization Setup section, when a change order is placed within the French sales office, a sales manager from the French office logs on to the system to update the order which invokes a PL/SQL API for that change. If the Organization ID contained in the header has been assigned with a property value, such as `207` for the French sales office, the Organization ID associated with the sales manager will be set to French sales office for the invocation of the API.

> **Note:** For Oracle E-Business Suite Release 12, *Organization ID* is automatically included in the header along with other applications context elements. For Release 11.5.10, you must apply 11i.ATG_PF.H.delta.5 (RUP5) (patch 5473858) to Oracle Applications instance in order to have *Organization ID* displayed in the header.

Supporting for Multiple Languages

By leveraging the message normalization feature addressed earlier and the Multiple Language Support (MLS) feature from Oracle E-Business Suite, Adapter for Oracle Applications provides a comprehensive language support mechanism that allows an appropriate language to be dynamically set at run time.

NLS Language Property Value Takes the Priority

When an application user retrieves data from the system for a transaction or receives error messages while executing APIs, the session language of data or error messages can be determined based on the following conditions:

1. The NLS Language value can be set by using the header property `jca.apps.NLSLanguage`. If a valid language value is passed (i.e. language is enabled in Oracle E-Business Suite instance), the session language is set to the corresponding language.

2. In case the NLS Language header property is not passed, Adapter for Oracle Applications will use the default language of the user, based on the user preferences, to set the session language.

3. If the user's default language is not found or set, NLS Language (`jca.apps.NLSLanguage`) property value would be set to `'US'` (American).

This mechanism allows an appropriate session language to be dynamically set at run time first based on the passed NLS Language property value, then the user's default language, and last determined by the default NLS Language property value 'US'.

For more information about how to set NLS Language property value by using `jca.apps.NLSLanguage`, see Supporting for Normalized Message Properties, page 3-8.

Determining a User's Default Language

To identify the default language used in the database session for data query and retrieval, Adapter for Oracle Applications will first examine the *ICX: Language* profile value at all levels including user, responsibility, application, and site. If it is not set at any of those levels, Adapter for Oracle Applications then takes `NLS_LANGUAGE` parameter from the database instance National Language Support (NLS) parameters. The NLS parameters initialized in the session are:

- `NLS_LANGUAGE`
- `NLS_SORT`
- `NLS_DATE_FORMAT`
- `NLS_DATE_LANGUAGE`
- `NLS_NUMERIC_CHARACTERS`

- NLS_TERRITORY

For example, when a user with a default language Japanese logs into the system and performs a transaction through the execution of an API that the user defined in the Partner link of the BPEL process, Adapter for Oracle Applications will first examine if the NLS Language property value is passed. If it is passed with a valid language, then the session language will be set based on the passed value regardless of the default language. If the NLS Language property is not passed, Adapter for Oracle Applications will set the session language to the preferred / default language of the user. In case that cannot be found, the language would be set to `'US'` (American). The default language is set in the General Preferences page of Oracle Applications.

> **Note:** The default language set in the General Preference page updates the *ICX: Language* profile option.

When the applications context is set, user preferences like date formats, time zone information, etc. would be taken care automatically.

Please refer to the Set Preferences section, Getting Started with Oracle Applications chapter, *Oracle Applications User's Guide* for the information on how to set the user preferences.

Understanding Adapter for Oracle Applications Security

Security is the most critical feature that is designed to guard application content from unauthorized access. By leveraging Oracle User Management function security, Adapter for Oracle Applications provides a security feature which only allows users with authorized privileges to execute APIs that they are exposed through the BPEL process to update Oracle Applications. This protects application programming interfaces (APIs) from unauthorized access or execution without security checks.

Please note that Adapter for Oracle Applications provides this security support as an optional feature. If you want all the login users to access and execute APIs without security checks, you can turn the security feature off using the "EBS Adapter for BPEL, Function Security Enabled" (EBS_ADAPTER_FUNCTION_SEC_ENABLED) profile option.

- If it is set to 'Y', then the function security feature is enabled and all API calls for PL/SQL APIs, Oracle e-Commerce Gateway, and concurrent programs will be checked for user security before they are invoked.

- If it is set to 'N' (default value), then the function security feature is disabled. No security check is implemented during the invocation of all API calls.

> **Note:** For more information about this feature, see "Oracle Fusion Middleware Adapter for Oracle Applications, Release 11*g*", My Oracle

Support Knowledge Document 787637.1 for details.

This section includes the following topics:

- Function Security for Adapter for Oracle Applications, page 3-16
- Creating Security Grants, page 3-17

Function Security for Adapter for Oracle Applications

Function security is the basic access control in Oracle Applications. It restricts user access to individual menus and menu options within the system regardless of which application data in the row. Since APIs are stored procedures that enable you to insert and update data in Oracle Applications, when having the function security layer enforced on the access to an API, it actually implicitly restricts the data access to the application.

To allow appropriate users with right privileges to execute APIs, Adapter for Oracle Applications leverages Oracle User Management Role-Based Access Control security (RBAC) to reinforce the function security through user roles and whether a user can access an API is determined by the roles granted to the user. A role can be configured to consolidate the responsibilities, permissions, permission sets, and function security policies that users require to perform a specific function. This simplifies mass updates of user permissions because changes can be done through roles which will inherit the new sets of permissions automatically. Based on the job functions, each role can be assigned a specific permission or permission set if needed. For example, a procurement organization may include 'Buyer', 'Purchasing Manager', and 'Purchasing Support' roles. The 'Purchasing Manager' role would include a permission set that contains all Purchase Order (PO) Creation, PO Change, and Contract PO related APIs allowing the manager role to perform a job function while the Buyer or Support role may not have the access privileges.

In Adapter for Oracle Applications, all annotated APIs that reside in Oracle Integration Repository are registered on the FND_FORM_FUNCTIONS table so that the function security (FND_FORM_FUNCTIONS) can be applied. This allows the creation of a secured function for each API.

By leveraging the concept of permission sets, Adapter for Oracle Applications allows related APIs to be grouped and sequenced under one permission set; each permission set can be associated with a function role and then assigned to users through security grants. When a user logs in to Oracle E-Business Suite and tries to access an API exposed through the BPEL process, if the security feature is enabled, the function security API will be invoked to validate whether the user is authorized to have the execution privileges on the API.

For example, if a user does not have the access privileges for a PL/SQL API exposed through a BPEL process, the execution of that BPEL process will fail while trying to invoke the PL/SQL API.

Without the authorized privileges, the Function Security Validation Exception message will be raised indicating that the user does not have the privilege for a specific PL/SQL API.

For more information on Function Security and RBAC security models, see *Oracle Applications System Administrator's Guide - Security* for details.

Creating Security Grants

To secure the API invocation only to a user with appropriate execution privileges, Adapter for Oracle Applications uses the following steps to create security grants to users through user roles:

1. Creating a Permission Set, page 3-17
2. Creating a User Role, page 3-19
3. Granting a Permission Set to a User Through a Role, page 3-20

Creating a Permission Set

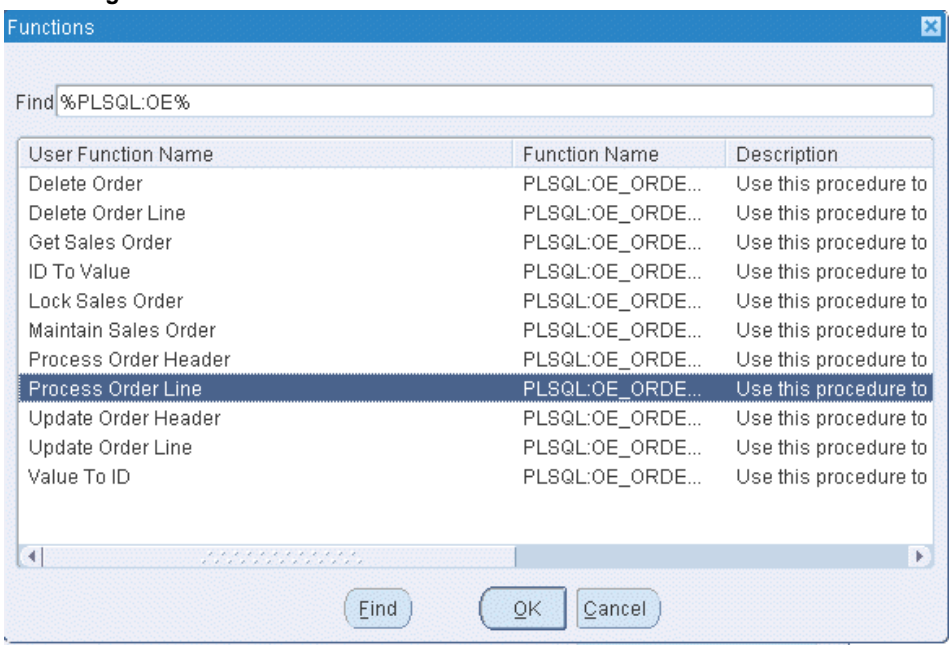
Use the following steps to create a permission set:

1. Log in to Oracle E-Business Suite using the System Administrator responsibility.
2. Select Application: Menu from the Navigator to access the Menus window.
3. Enter the following menu information:
   - Menu: Enter an appropriate menu name (such as 'OE_PROCESS_LINE_PS').
   - User Menu Name: Enter an appropriate user menu name (such as 'Order Manager Process Line Permission Set').
   - Menu Type: Permission Set
   - Description: Enter description information for this menu.
4. Add all the functions that you want to group on this Permission Set by entering values for Seq and Function.
   1. Enter the Seq field.
   2. In the Function column, search for the functions you want to assign to this permission set.

      Select an appropriate function name by performing a search in the Functions window. For example the syntax for searching public PL/SQL APIs is: `PLSQL:<package name>:<procedure name>`. You can enter `%PLSQL:OE%`

in the Find field and click **Find** to execute the search.

Searching for Functions

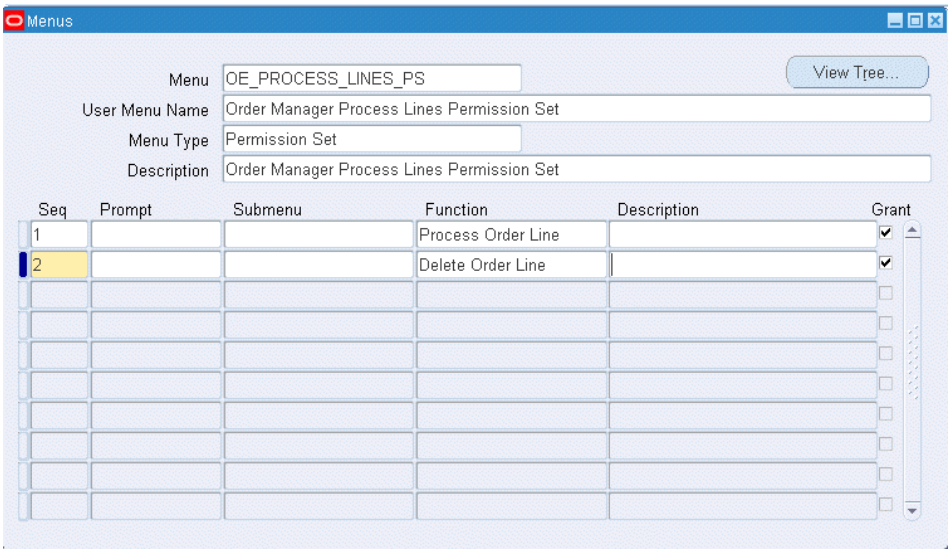


Based on your BPEL process, select appropriate functions and then grant the permissions to the APIs that you will be invoking from the BPEL process.

For example, for a sales order line change BPEL process, you select sales order line change related functions contained in the order change PL/SQL API and group them as a permission set, and then grant the permission set to an appropriate user through a role.

See: Creating a User Role, page 3-19.

Permission Set Menu



5. Save the Permission Set.

Creating a User Role

Permission sets are granted through user roles. Therefore, you must first create a role and then assign the role to a user.

Use the following steps to create a user role:

1. Log in to Oracle E-Business Suite using the User Management responsibility.
2. Select Roles & Role Inheritance from the Navigator to access the Roles & Role Inheritance page.
3. Click **Create Role** to access the Create Role page.
4. Enter the following information to create a role:
   - Category: Select Miscellaneous from the drop-down list.
   - Role Code: Enter an appropriate role code (such as 'EBS_ADAPTER_ROLE').
   - Display Name: Enter appropriate information for the display name (such as 'EBS Adapter Role').
   - Description: Enter appropriate information for the description (such as 'EBS Adapter Role').

- Application: Select an appropriate application (such as 'Application Object Library').
- Active Date: Enter an appropriate date which is earlier than or equal to today's date so that the role can become valid right away.

5. Save the information and click **Create Grant**.
6. Enter the following information in the Create Grant: Define Grant page:
   - Name: Enter an appropriate name (such as 'EBS_ADAPTER_GRANT').
   - Description: Enter description information for this grant.
7. Click **Next**.
8. In the Create Grant: Define Object Parameters and Select Set page, select the Permission Set you created earlier in the Creating a Permission Set section, page 3-17 and click **Next**.
9. Click **Finish**.

Granting a Permission Set to a User Through a Role

Use the following steps to grant a permission set to a user through a role:

1. Log in to Oracle E-Business Suite using the User Management responsibility.
2. Select Users from the Navigator to access the User Maintenance page.
3. Search for the user you want to assign the role and click **Go**.
4. Select the **Update** icon next to the user name that you want to assign the role.
5. In the Update User page, click **Assign Roles** to have the Search window populated which allows you to search for the role that you created earlier.
6. Select the role (such as 'EBS_ADAPTER_ROLE') and save your update.

Logging

Oracle Fusion Middleware Adapter for Oracle Applications implements the Oracle SOA Suite's logging framework to write the diagnostic log files in text format. Therefore, whenever the Oracle E-Business Suite services are invoked using Oracle Adapter for Oracle Applications, the log messages are recorded which can be accessed by system administrator. This enriches the problem identification mechanism to track any issues during service invocations at run time for Oracle Adapter for Oracle Applications.

Oracle Adapter for Oracle Applications and technology adapters implement the `LogManager` interface of JCA Binding Component, which redirects the log files for both inbound and outbound interactions to the `soa-diagnostic.log` file in the Oracle Diagnostic Logging (ODL) format. These log files record all types of events including startup and shutdown information, errors and warning messages, access information on HTTP requests, and additional information. It provides a quick, inside-out view about the invocation process which greatly helps administrators identify and resolve potential issues efficiently. With proper log-level configuration in Oracle Enterprise Manager Fusion Middleware Control Console, you can view ODL level log files written to a single file at run time for Oracle Adapter for Oracle Applications as well as technology adapters.

To better understand how the logging mechanism work, the following topics are discussed in this chapter:

- Enabling Logging for Adapters, page 3-21
- Searching and Viewing Adapter Logs, page 3-26

For more information about SOA Suite, see *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite.*

Enabling Logging for Adapters

All logs of Oracle Adapter for Oracle Applications and technology adapters are redirected to the `soa-diagnostic.log` file in the Oracle Diagnostic Logging (ODL) format. To be able to view logs for Oracle Adapter for Oracle Applications, you need to set or enable an appropriate message type and its associated log level for the logger `oracle.soa.adapter` using the Oracle Enterprise Manager Fusion Middleware Control Console. This enables the log settings at the Oracle SOA Suite level.

> **Note:** The logger configuration is under `$FMWHOME/user_projects/domains/soainfra/config/fmwconfig/servers/soa_server1/logging.xml` and the corresponding logger name for all Oracle JCA Adapters is called `oracle.soa.adapter`.

The following table lists the diagnostic message types and log levels:

> **Note:** For each message type, possible values for message level are from 1 (highest severity) through 32 (lowest severity). Some components support only some of the levels for each message type. Generally, you need to specify only the type; you do not need to specify the level.
>
> The default message type for each logger is set to NOTIFICATION, level 1.

Message Type	Level	Description
INCIDENT_ERROR	1	A serious problem that may be caused by a bug in the product and that should be reported to Oracle Support. Examples are errors from which you cannot recover or serious problems.
ERROR	1	A serious problem that requires immediate attention from the administrator and is not caused by a bug in the product. An example is if Oracle Fusion Middleware cannot process a log file, but you can correct the problem by fixing the permissions on the document.
WARNING	1	A potential problem that should be reviewed by the administrator.
NOTIFICATION	1	A major lifecycle event such as the activation or deactivation of a primary subcomponent or feature.
NOTIFICATION	16	A finer level of granularity for reporting normal events.
TRACE	1	Trace or debug information for events that are meaningful to administrators, such as public API entry or exit points.
TRACE	16	Detailed trace or debug information that can help Oracle Support diagnose problems with a particular subsystem.

Message Type	Level	Description
TRACE	32	Very detailed trace or debug information that can help Oracle Support diagnose problems with a particular subsystem.

To set the diagnostic message type and log level for Oracle Adapter for Oracle Applications in SOA Suite:

Use the following procedures to set the message type and its associated log level:

1. Navigate to `http://<servername>:<portnumber>/em`.

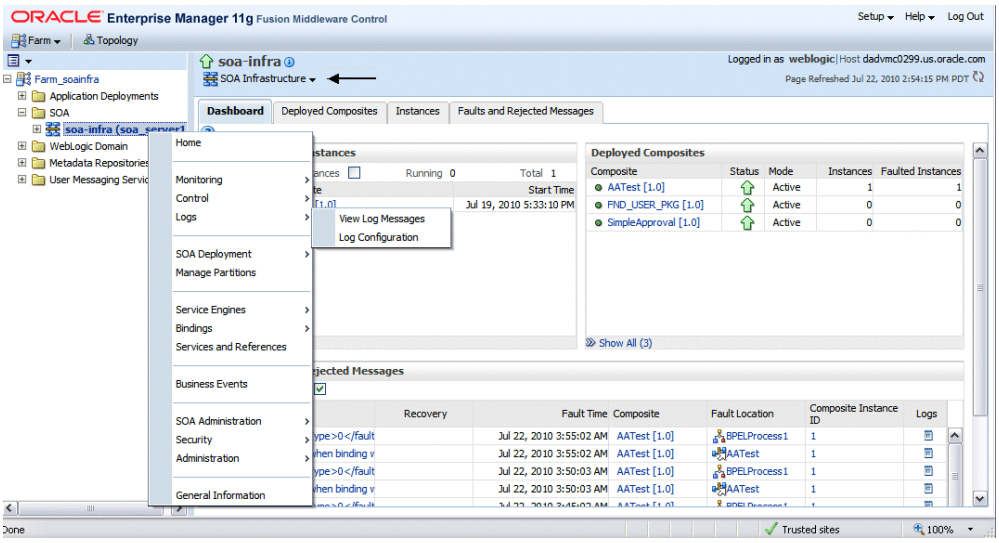
   The Oracle Enterprise Manager Fusion Middleware Control Console home page is displayed.

2. Enter username and password to log on to the console.

3. Right-click `soa-infra` from the SOA Folder in the Navigator tree.

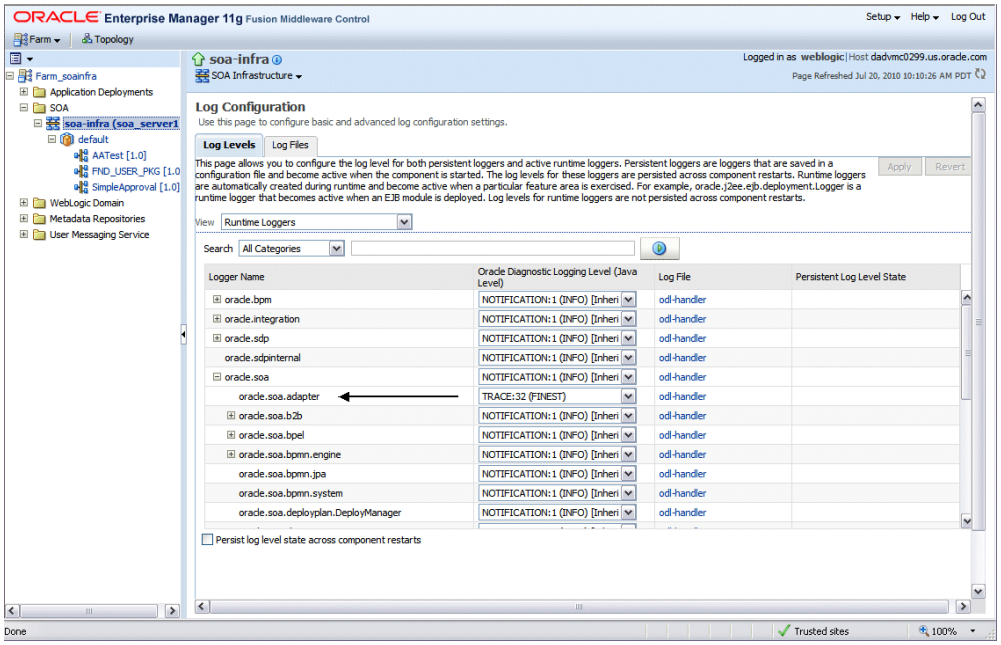
4. Select **Logs > Log Configuration** from the pop-up menu.

   Note: You can also access the Log Configuration page by clicking the SOA Infrastructure Menu and selecting **Logs > Log Configuration** from the drop-down menu.



This opens the Log Configuration page where you can view a list of loggers (persistent or active run time) and configure the Oracle Diagnostic Logging (ODL) level for setting the amount and type of information to write to a log file and the log level state.

Configuring Log Level for Adapters



5. Select the **Log Levels** tab.

6. Select one of the following values from the View drop-down list:

   - **Runtime Loggers (default):** Runtime loggers are automatically created during run time and become active when services are getting executed. For example, `oracle.soa.b2b` or `oracle.soa.bpel` are runtime loggers. Log levels for runtime loggers are not persistent across component restarts.

   - **Loggers with Persistent Log Level State:** Persistent loggers are loggers that are saved in a configuration file and become active when the component is started. The log levels for these loggers are persistent across component restarts.

     Note: By default, the log level is set for Runtime Loggers. Runtime loggers do not persistent across when a component restarts. To ensure that log levels persist across component when it restarts, select **Loggers With Persistent Log Level State** from the View list.

7. Expand the `oracle.soa` node to locate `oracle.soa.adapter` logger in the Logger Name list. Select the logging level from the Oracle Diagnostic Log Level drop-down list. For example, select 'TRACE:32 (FINEST)'.

8. Click **Apply**.

Creating and Editing Log File in the Log Files Tab

You can edit a specific log file by clicking the log handler link displayed in the Log File column. This opens the Log Files tab where you can configure the basic and advanced log configuration settings. These settings include handler's name, the log file in which the log messages are logged, the format of the log messages, the rotation policies used, and other parameters based on the log file configuration class.

For example, select the log handler from the table and click **Edit Configuration**. The Edit Log File dialog box is displayed.

- To change log file location, enter a new path in the Log Path field.
- To configure message levels, select the logging level from the Log Level drop-down list. For example, select 'TRACE:32 (FINEST)'.
- To configure log file rotation, in the Rotation Policy section, select either Size Based or Time Based log file with appropriate information.

Fore more information on how to configure log file, see the Managing Log Files and Diagnostic Data Chapter, *Oracle Fusion Middleware Administrator's Guide*.

For more information about SOA Suite, see *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite.*

Searching and Viewing Adapter Logs

Oracle Adapter for Oracle Applications and technology adapters implement the `LogManager` interface of JCA Binding Component, which redirects the log files written to a single file at run time in the Oracle Diagnostic Logging (ODL) format.

For both outbound and inbound interactions, the log files are redirected to the single file `soa-diagnostic.log`.

The log files for Oracle SOA Suite that is deployed to the `server-soa` managed server are located in `MW_HOME/user_projects/domains/<domain_name>/servers/server-soa/logs/soa-diagnostic.log`.

To search and view Oracle Adapter for Oracle Applications log files, you can use Oracle Enterprise Manager Fusion Middleware Control Console, the WLST `displayLogs` command-line tool, or you can download log files to your local client and view them using another tool (for example a text editor, or another file viewing utility). For information on how to use WLST command-line tool to search and view log files, see the Managing Log Files and Diagnostic Data Chapter, *Oracle Fusion Middleware Administrator's Guide*.

Steps to Search and View Adapter Logs Using Oracle Enterprise Manager Fusion Middleware Control Console

Use the following steps to search Adapter Logs through the Oracle Enterprise Manager

Fusion Middleware Control Console:

1. Navigate to `http://<servername>:<portnumber>/em`.

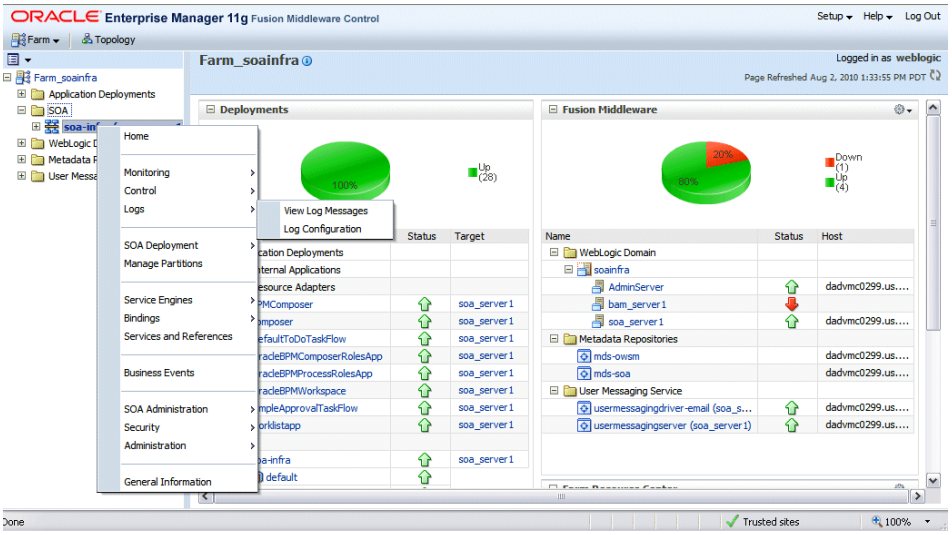
   The Oracle Enterprise Manager Fusion Middleware Control Console home page is displayed.

2. Enter username and password to log on to the console.

3. There are two ways to access the Log Messages page from the Navigator tree in the left pane:

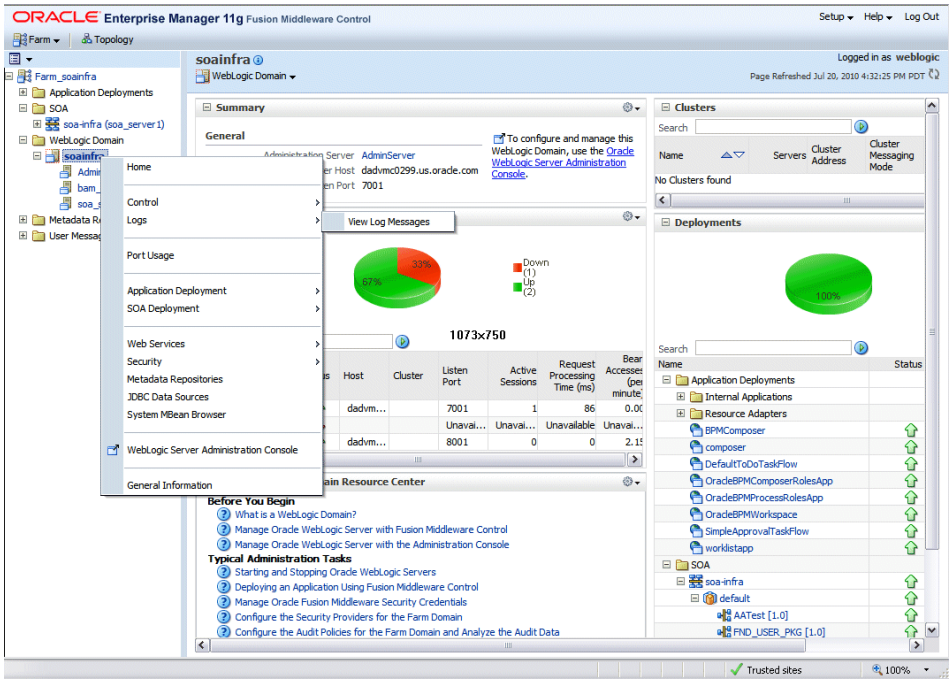
   - From the SOA folder, right-click `soa-infra`.

Navigating from SOA Folder



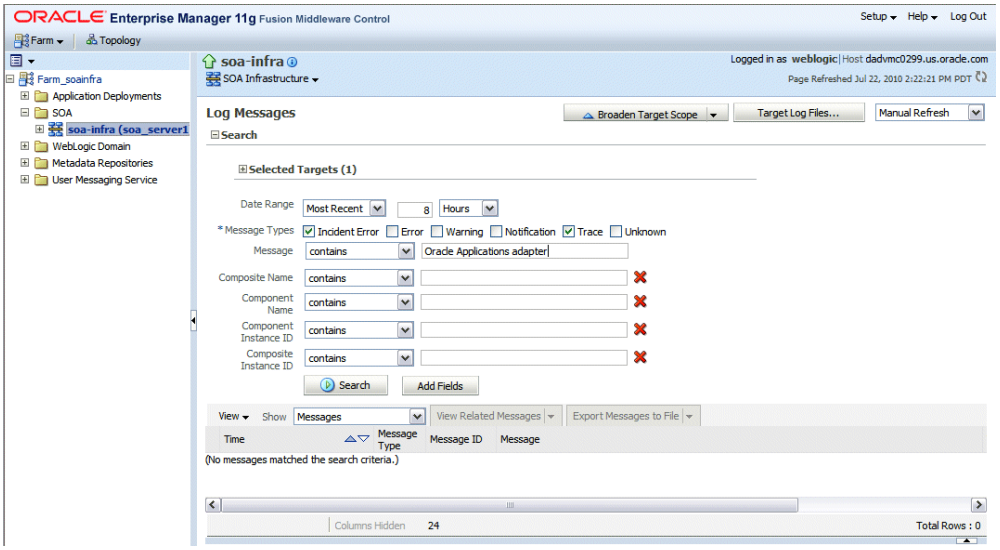
   - From the WebLogic Domain folder, right-click `soainfra`.

Navigating from WebLogic Domain Folder



Select **Logs > View Log Messages** from the pop-up menu. The Log Messages page is displayed with the Search section and a table that shows a summary of the messages with default search criteria.

Log Messages Page: Search



4. Enter the following search criteria for searching the log messages for Oracle Adapter for Oracle Applications:

   - **Date Range:** Select your value from the drop-down list and enter an appropriate number. For example, 'Most Recent' 6 hours.

     If 'Time Interval' is selected from the drop-down list, select the calendar icon for Start Date and then select the date and time. Similarly, select the calendar icon for End Date and then select the date and time.

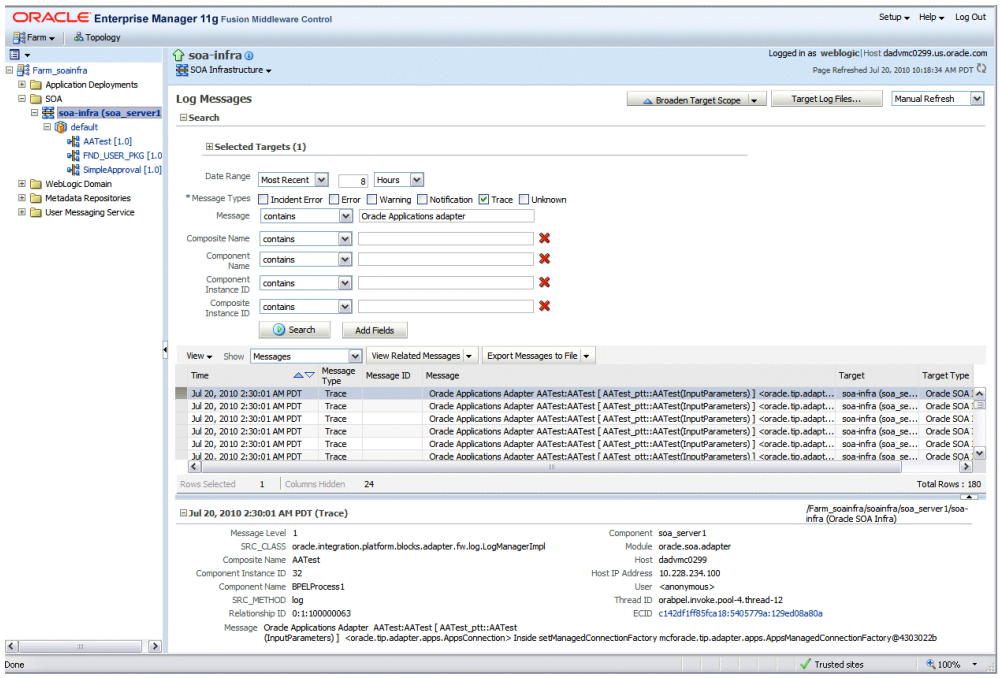
   - **Message Types:** Select one or more of the message types. For example, select the **Trace** check box if it is the message type configured earlier for Oracle Adapter for Oracle Applications.

   - **Message:** Select 'Contains' from the list of values and then enter 'Oracle Applications Adapter' in the text box.

   - Specify more search criteria if needed in the Search section.

     You can optionally add more search criteria by clicking **Add Fields**. This action allows you to add more criteria, such as Host, which lets you narrow the search to particular hosts. Then click **Add**.

5. Execute the search by clicking **Search**. All messages that match your search criteria will be retrieved and displayed in the table. These messages can be displayed as messages, or can be grouped by message type or message ID depending on the selected value in the Show field.



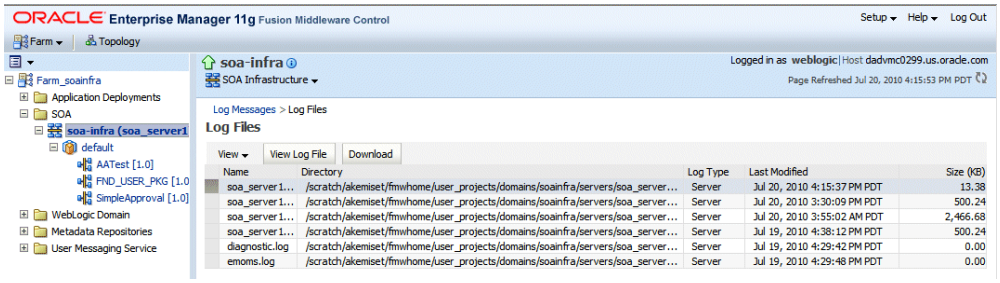
6. Click one of the log messages from the table. The selected message details, such as message level, component, ECID (Execution Context ID), Relationship ID, actual message, are displayed below the table of messages.

   Clicking the ECID link retrieves related messages with the same ECID in the Related Messages by ECID page. For more information on related messages, see Correlating Messages Across Log Files and Components, page 3-32.

7. Select an appropriate output option from the **Export Messages to File** drop-down list if you want to export log messages as a Oracle Diagnostic Log Text file (.txt) XML file (.xml), or Comma-Separated List (.csv) file.

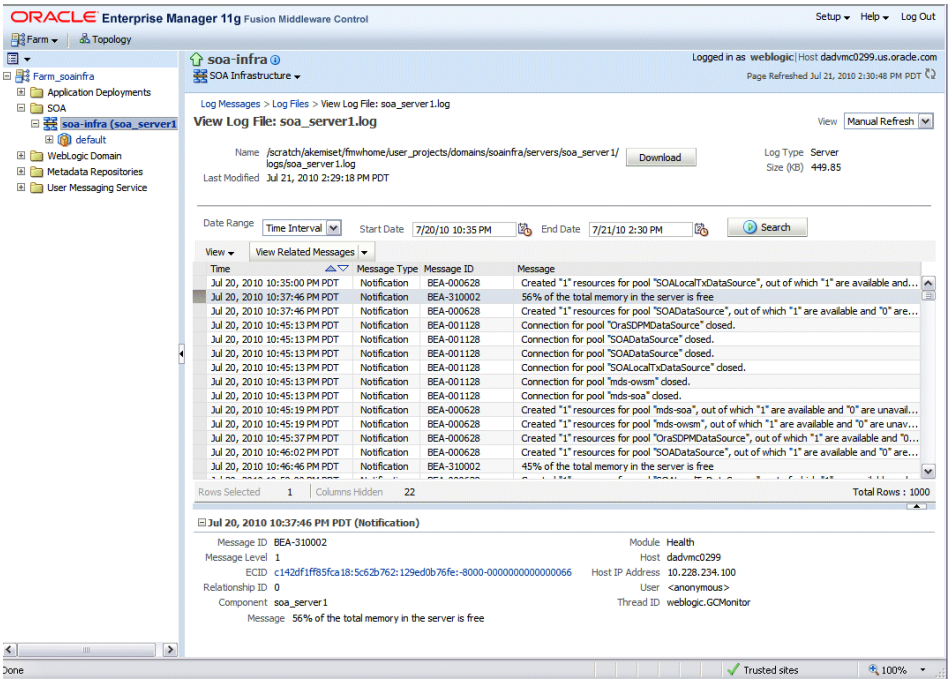
8. Click **Target Log Files** to open the Log Files page where you can view a list of log files related to the managed server (`server-soa`).

Log Files Page



1. Select a file and click **View Log File**. The View Log File page is displayed for the selected log file where you can view a list of messages contained in this log.

View Log File Page



2. To view the details of a message, select the message. The message details, such as message level, component, ECID, Relationship ID, actual message, are displayed below the table of messages.

3. To view messages that are related by time or ECID, click **View Related Messages** and select 'by Time' or by 'ECID (Execution Context ID)'.

   Alternatively, clicking the ECID link directly from the message details also retrieves related messages with the same ECID in the Related Messages by ECID page.

Correlating Messages Across Log Files and Components

Oracle Fusion Middleware components provide **message correlation** information for diagnostic messages. Message correlation information helps those viewing diagnostic messages to determine relationships between messages across components. Each diagnostic message contains an Execution Context ID (ECID) and a Relationship ID.

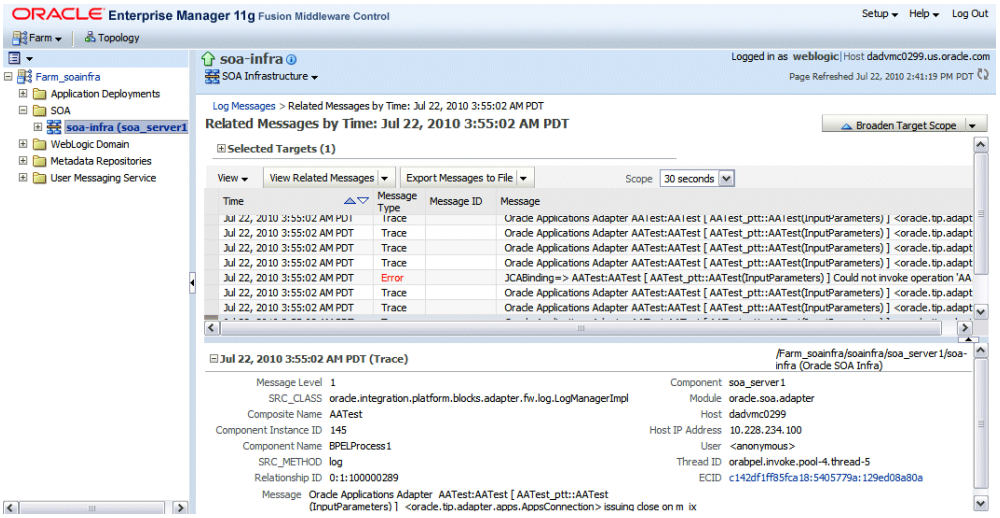
An ECID is a globally unique identifier associated with the execution of a particular request. An ECID is generated when the request is first processed. A Relationship ID distinguishes the work done in one thread on one process from the work done by any other threads on this and other processes on behalf of the same request.

While viewing log messages in the Oracle Enterprise Manager Fusion Middleware Control Console, you can view correlated messages by selecting a log message first, and then selecting one of the following values from the **View Related Messages** drop-down list:

> **Note:** The **View Related Messages** selection is available only when 'Messages' is chosen in the Show field when displaying all matching messages based on search criteria. If 'Group by Message Type' or 'Group by Message ID' is selected in the Show field, then all matching messages are displayed by groups based on message type or message ID. In this situation, the **View Related Messages** field is not available.

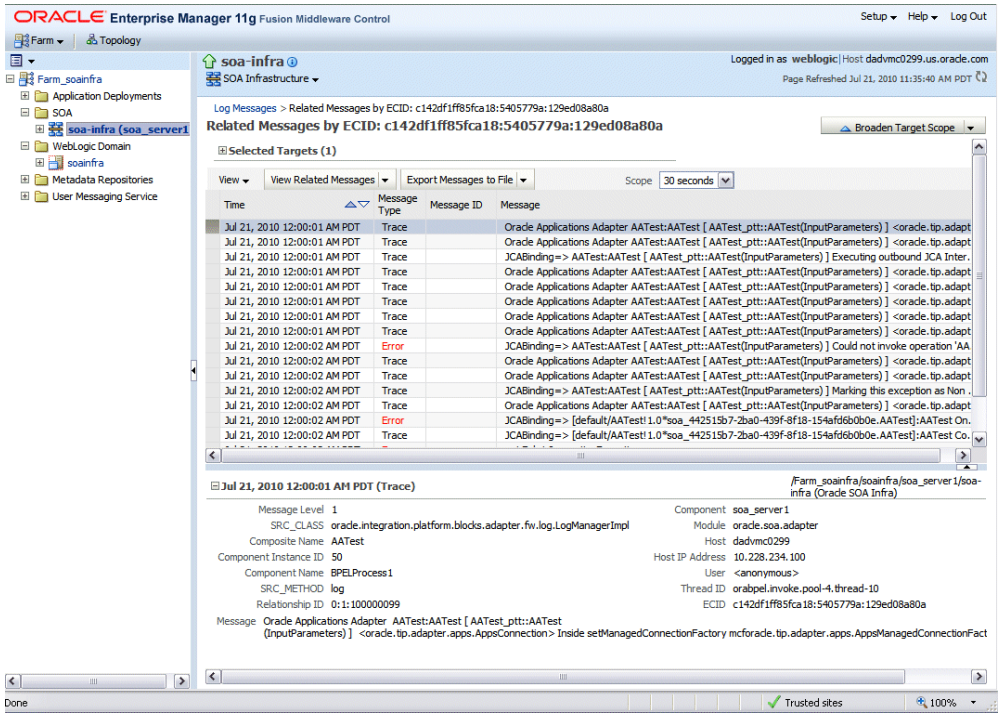
- **by Time:** This displays the Related Messages by Time page where all messages with the same timestamp as the selected message are displayed in this page.

Related Messages by Time Page



- **by ECID (Execution Context ID):** This displays the Related Messages by ECID page where all messages with the same ECID as the selected message are displayed in this page.

Related Messages by ECID Page



By searching for related messages using the message correlation information, multiple messages can be examined and the component that first generates a problem can be identified. Message correlation data can help establish a clear path for a diagnostic message across components, within which errors and related behavior can be understood.

Secured Connection Between Oracle E-Business Suite and Oracle Fusion Middleware SOA Suite Using J2EE Data Source Implementation

By implementing the J2EE Data Source for secured connection between Oracle E-Business Suite and Oracle SOA Suite, two distinct advantages can be leveraged. Firstly, to get the secured connection to the Oracle E-Business Suite's application database you do not require the database administrator's username and password, just FND username and password (concept of Oracle Applications username and password) is sufficient. Secondly, since the password is not stored in the middleware, not only this eliminates the security risk, but also does away the need to keep the password in-sync between Oracle E-Business Suite and SOA Suite.

Oracle Adapter for Oracle Applications uses a new mechanism to authenticate users at run time and get the connection to Oracle E-Business Suite databases through the use of J2EE data sources. This approach is native to Oracle E-Business Suite in defining the connection pool to access the application database.

With this new mechanism, account details information including application login user name and password that was required as part of the configuration for database connection is now added together with the dbc file location as input parameters during the J2EE data source creation.

To accomplish this process, the following steps are used to define J2EE data source connection to the Oracle E-Business Suite database:

1. Register your Service-Oriented Architecture (SOA) suite middle tier node on the Oracle E-Business Suite environment and generate the dbc file used by the data source implementation to instantiate the connections.

2. Copy the dbc file to the middle tier server where your SOA suite server runs, and place it on a location in the file system to which the SOA suite owner has access.

3. Create a connection pool where you need to enter the application login user name, password, and dbc file location as the connection factory properties.

4. Create an application data source. This is the step that you associate the application data source with the Java Naming and Directory Interface (JNDI) name for the application database connection for Oracle Adapter for Oracle Applications.

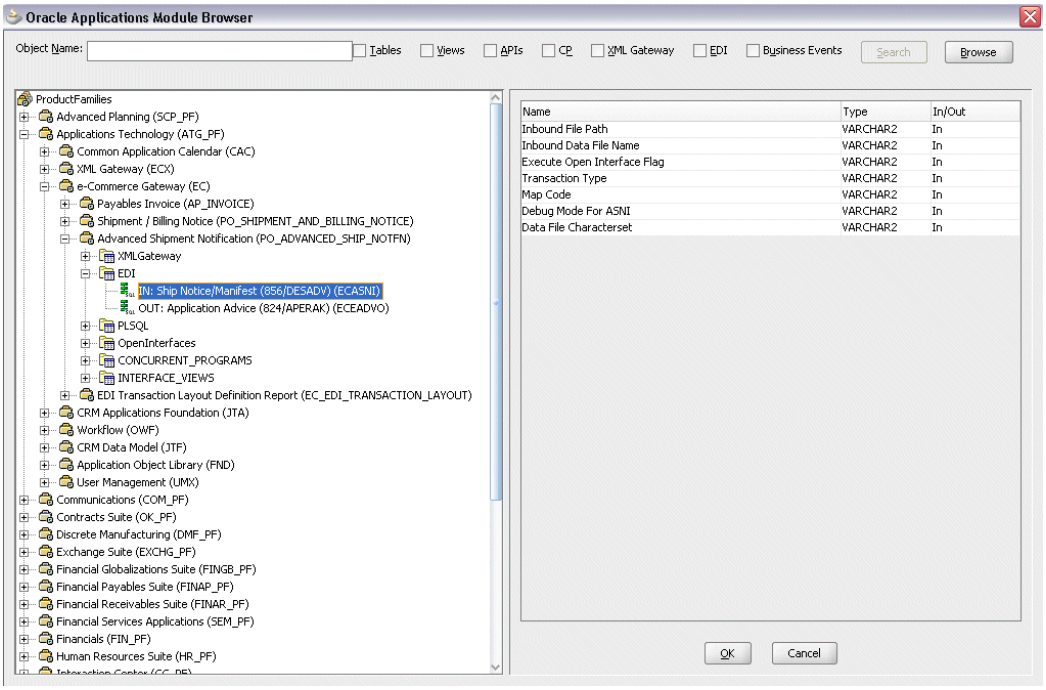
To enable the native Oracle E-Business Suite connectivity using J2EE data sources feature,

> **Note:** To have this feature available, the minimum requirement for Oracle E-Business Suite Release 11*i* is 11*i*.ATG_PF.H.Delta.6 (RUP6) and for Oracle E-Business Suite Release 12 is 12.0.4 release.
>
> Additionally, you must apply necessary patches to enable the connectivity between Oracle E-Business Suite and an external application server. See "Oracle Fusion Middleware Adapter for Oracle Applications, Release 11*g*" My Oracle Support Knowledge Document 787637.1 for details.

Understanding the Oracle Applications Module Browser

In addition to the interfaces that are made available through Oracle Integration Repository, Adapter for Oracle Applications enables you to use business events, customized PL/SQL APIs, customized XML Gateway maps, and selected concurrent programs, all of which you can explore using the Oracle Applications Module Browser.

Oracle Applications Module Browser



The Oracle Applications Module Browser is a key component of Adapter for Oracle Applications. You use the Module Browser to select the interface needed to define a partner link. The Module Browser combines interface data from Oracle Integration Repository with information about the additional interfaces supported by Adapter for Oracle Applications, organized in a tree hierarchy as follows:

```
ProductFamilies
 |-[product_family]
 |   |-[product]
 |      |-[business_entity]
 |         |-XML Gateway ([n])
 |         |-EDI ([n])
 |         |-PLSQL ([n])
 |         |   |-[package_name]
 |         |-OpenInterfaces ([n])
 |            |-[OpenInterface_name]
 |               |-Tables ([n])
 |               |-Views ([n])
 |               |-ConcurrentPrograms ([n])
 |-Other Interfaces
    |-Business Events
    |-Custom Objects
       |-PLSQL APIs
       |   |-[package_name]
       |-XMLGateway Maps
          |-Inbound
          |-Outbound
```

- The items under Other Interfaces, as well as certain PL/SQL APIs and concurrent programs under the *[product family]* hierarchy, are available through Adapter for Oracle Applications, but not through Oracle Integration Repository.

- The number of interfaces indicated by `[n]` only appears in the case of an Oracle E-Business Suite 11.5.10 instance is used. It will not be displayed if you are connecting to an Oracle E-Business Suite pre-11.5.10 or Release 12 instance.

The Oracle Integration Repository interface data populates the ***[product_family]*** sections, grouped according to the products and business entities to which they belong. Each interface type heading is followed by a number ***[n]*** indicating how many of that type are listed in that section.

Business events appear under Other Interfaces. Customized XML Gateway maps appear under **Other Interfaces > Custom Objects,** categorized as either inbound or outbound.

Customized PL/SQL APIs appear in two places:

- Procedures within a package that's already exposed via Oracle Integration Repository appear under the package name within a product family hierarchy.

- Procedures within a completely new package appear under the package name, under **Other Interfaces > Custom Objects**.

4

Using XML Gateway

This chapter covers the following topics:

- Overview of XML Gateway
- Design-Time Tasks for XML Gateway Inbound Messaging
- Creating a New BPEL Project
- Creating a Partner Link
- Adding a Partner Link for the File Adapter
- Configuring the Invoke Activities
- Configuring the Assign Activity
- Run-Time Tasks for XML Gateway Inbound Messaging
- Deploying the BPEL Process
- Testing the BPEL Process
- Verifying Records in Oracle Applications
- Design-Time Task for XML Gateway Outbound Messaging
- Creating a New BPEL Project
- Adding a Partner Link
- Adding a Receive Activity
- Adding a Partner Link for File Adapter
- Adding an Invoke Activity
- Adding an Assign Activity
- Run-Time Task for XML Gateway Outbound Messaging
- Deploying the BPEL Process
- Testing the BPEL Process
- Troubleshooting and Debugging

Overview of XML Gateway

Oracle Adapter for Oracle Applications provides a bridge between Oracle Applications and third party applications. Inbound and outbound XML data is exchanged between Oracle Applications and third party applications through the 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.

Oracle XML Gateway is an XML messaging based integration infrastructure essentially for business partner integration which includes a set of services that allows easy integration between Oracle Applications and third party applications. Oracle Applications utilize the Oracle Workflow Business Event System to support event-based XML message creation and consumption.

Oracle Adapter for Oracle 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.

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.

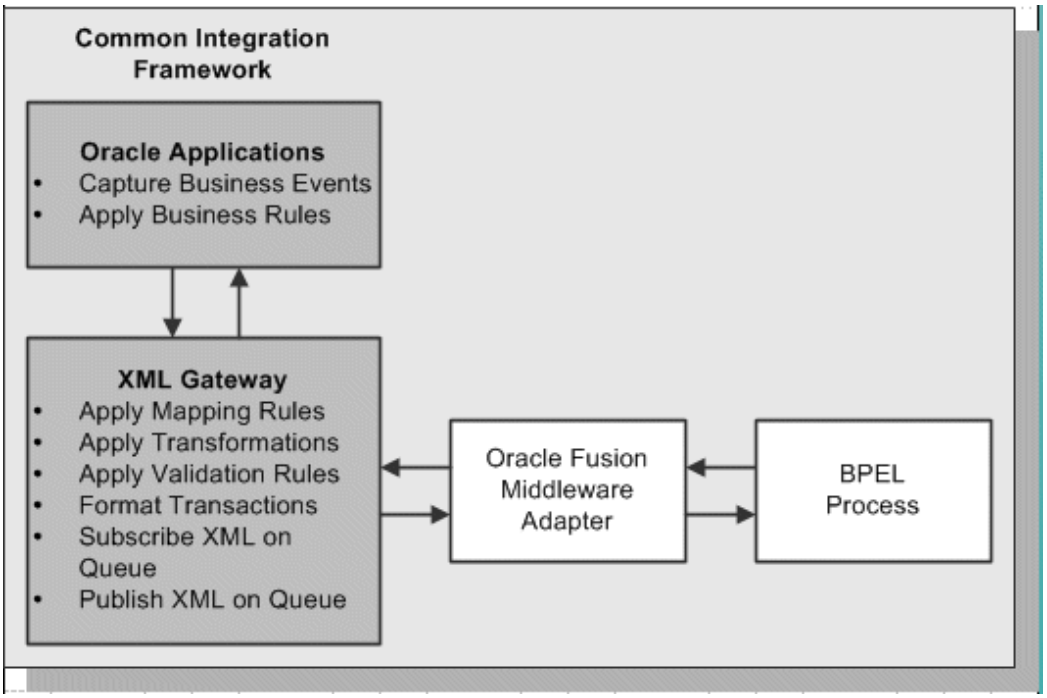
Integration Architecture

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 inbound and outbound XML Gateway message processing. For XML Gateway inbound message processing, the inbound message will be placed in the ECX_INBOUND queue. Agent Listeners running on ECX_INBOUND would enable further processing by the Execution Engine. Oracle XML Gateway picks this XML message, does trading partner validation, and inserts data into Oracle Applications. For XML Gateway outbound message processing, the outbound message will be first enqueued to the ECX_OUTBOUND queue. Oracle BPEL PM listens to ECX_OUTBOUND queue for the message with the same correlation Id BPEL. The message will then be dequeued to retrieve outbound data and then the outbound map will be invoked to update Oracle Applications.

XML Gateway Integration Architecture



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.

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, page 4-4. Oracle Workflow Business Event System copies the inbound messages to the proper inbound Transaction Queue.

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.

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. The following table describes some of these attributes.

Envelope Attributes

Attribute	Description
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.

Attribute	Description
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.
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.

Attribute	Description
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. **Note:** For more information, see *Destination Trading Partner Location Code* in 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`

Note: See *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

Design-Time Tasks for XML Gateway Inbound Messaging

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 Adapter for Oracle Applications to use XML Gateway. It describes the tasks required to configure Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper.

BPEL Process Scenario

Take the XML Gateway Inbound Process PO XML Transaction as an example to explain the BPEL process creation. In this example, the XML Gateway inbound message map is exposed as a Web service through `PROCESS_PO_007` inbound map. It allows sales order data including header and line items to be inserted into Order Management system while an associated purchase order is created.

When a purchase order is sent by a trading partner, the purchase order data is used as input to the BPEL process along with ECX Header properties such as `jca.apps.ecx.TransactionType`. The BPEL process then pushes this purchase order in ECX_INBOUND queue. Agent Listeners running on ECX_INBOUND would enable further processing by the Execution Engine. Oracle XML Gateway picks this XML message, does trading partner validation, and inserts order data to Order Management Application.

If the BPEL process is successfully executed after deployment, you should get the same order information inserted into the Order Management table once a purchase order is created.

Prerequisites to Configure XML Gateway Inbound

Setting XML Gateway Header Properties

You need to populate certain variables in the BPEL process in order to provide context information for Oracle Applications. This is accomplished by setting individual fields as header properties from the `SYSTEM.ECXMSG` object used in earlier releases.

The header properties can include, for example, `jca.apps.ecx.TransactionType`, `jca.apps.ecx.TransactionSubtype`, `jca.apps.ecx.PartySiteId`, `jca.apps.ecx.MessageType`, `jca.apps.ecx.MessageStandard`, `jca.apps.ecx.DocumentNumber` that you need to populate for the XML transaction to complete successfully.

These header property values can be defined through the Properties tab of an Invoke

activity. See Configure the Invoke activity, page 4-26 for more information.

Setting Up XML Gateway Trading Partner

You need to ensure that you have defined a valid inbound XML Gateway trading partner in the Trading Partner Setup form through XML Gateway responsibility.

For example, a Trading Partner (such as 'Business World' with Site information '2391 L Street San Jose CA 95106' and partner type 'Customer') has the following details for an inbound transaction:

- Transaction type: ONT
- Transaction sub type: POI
- Standard Code: OAG
- External transaction type: PO
- External transaction subtype: PROCESS
- Direction: IN
- Map: ONT_3A4R_OAG72_IN
- Source Trading partner location code: BWSANJOSE

Ensuring Agent Listeners Are All Up and Running

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. Select the **Workflow Administrator Web Applications** link from the Navigator.
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**, **ECX Transaction Agent Listener**, and the **Workflow Deferred Agent Listener**. Select the listener, and select Start from the **Actions** box. Click **Go**.

Based on the XML Gateway Inbound Process PO XML Transaction business scenario, the following design-time tasks are discussed in this chapter:

1. Create a new BPEL project, page 4-9.
2. Create a partner link, page 4-12.

3. Add partner links for file adapter, page 4-22.
4. Configure Invoke activities, page 4-26.
5. Configure an Assign activity, page 4-32.

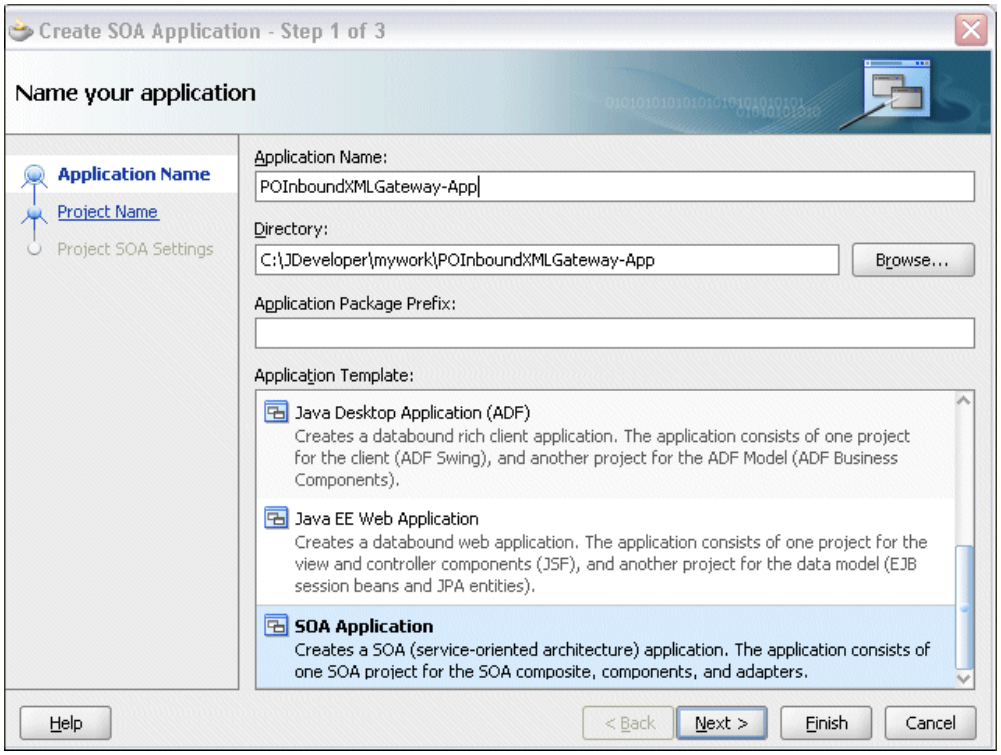
Creating a New BPEL Project

To create a new BPEL project:

1. Launch Oracle JDeveloper.
2. Click **New Application** in the Application Navigator.

   The Create SOA Application - Name your application page is displayed.

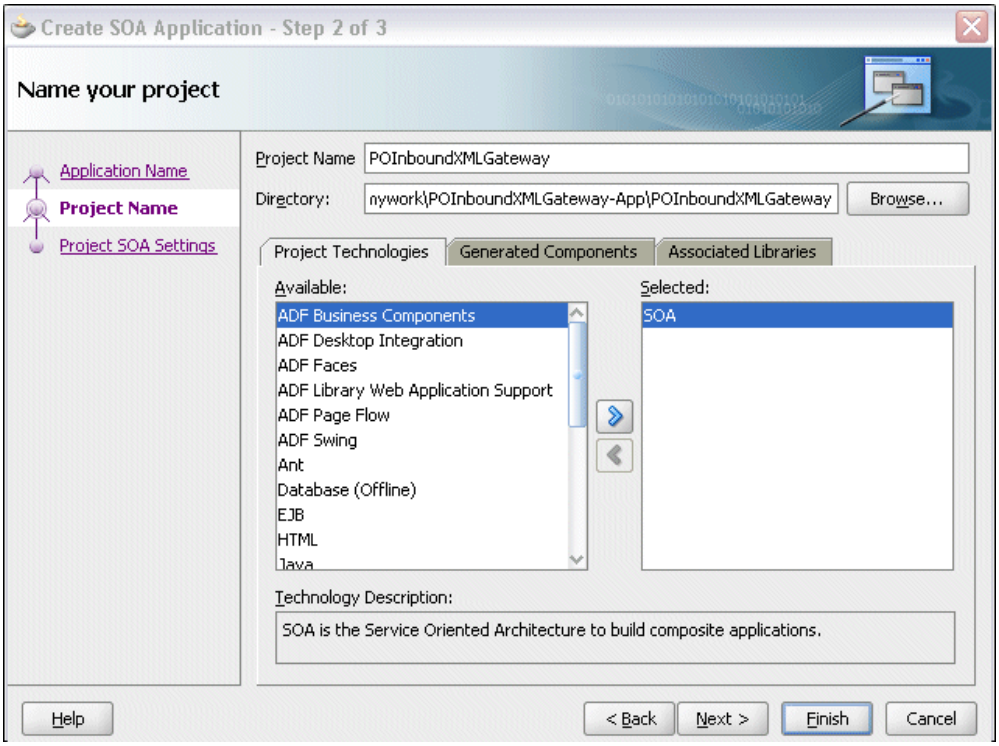
The Create SOA Application - Name your application Page



3. Enter an appropriate name for the application in the **Application Name** field and select **SOA Application** from the Application Template list.

   Click **Next**. The Create SOA Application - Name your project page is displayed.

The Create SOA Application - Name your project Page

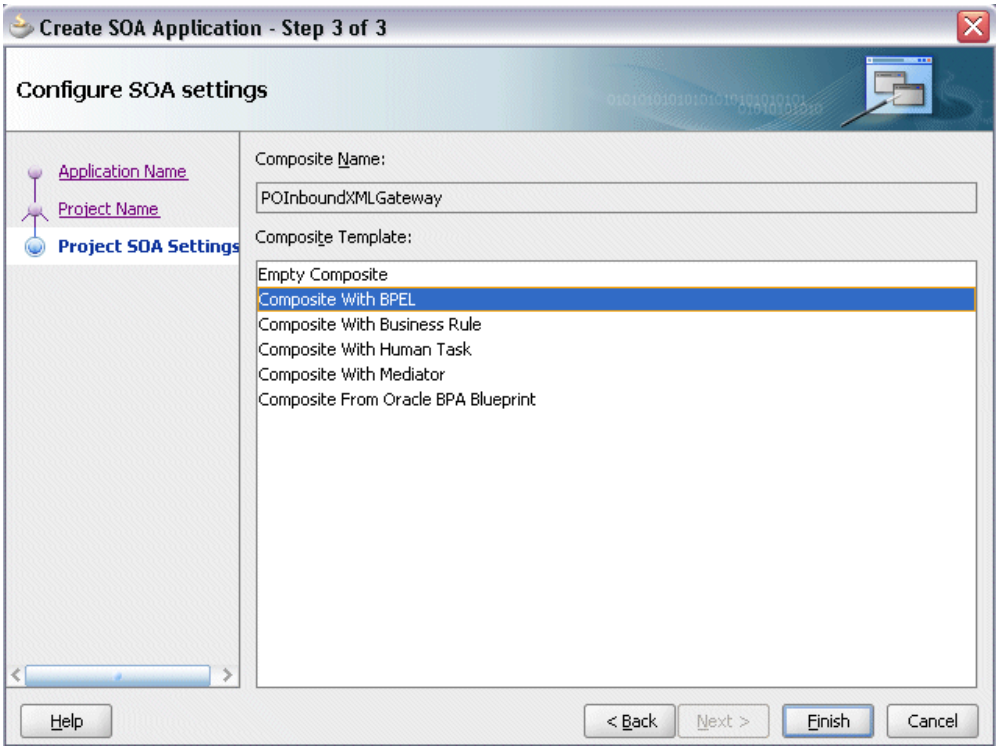


4. Enter an appropriate name for the project in the **Project Name** field. For example, `POInboundXMLGateway`.

5. In the Project Technologies tab, ensure that **SOA** is selected from the Available technology list to the Selected technology list.

   Click **Next**. The Create SOA Application - Configure SOA settings page is displayed.

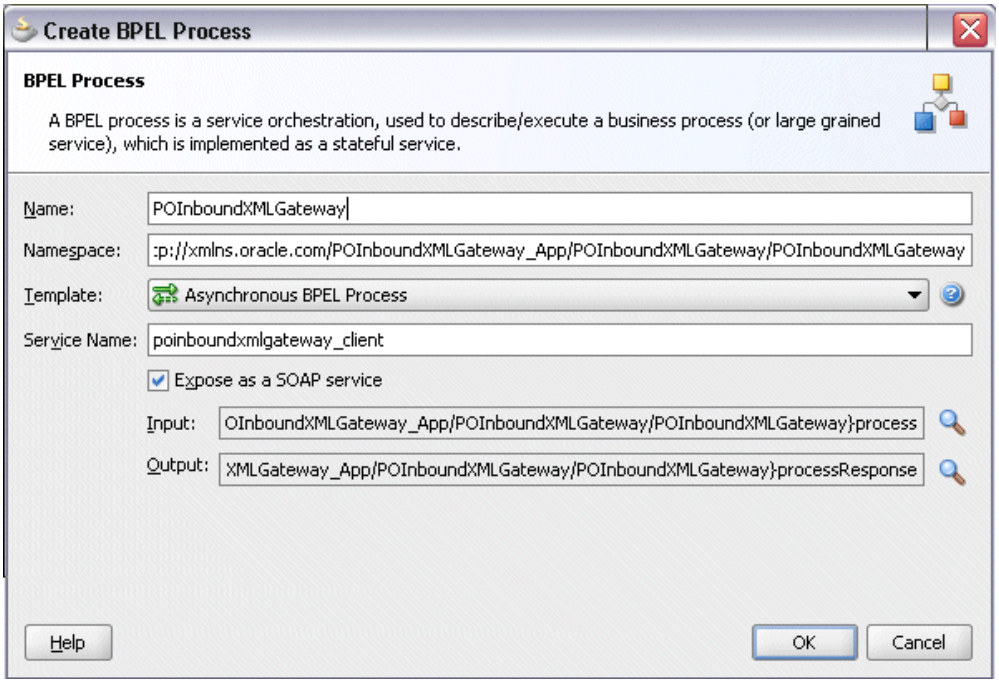
The Create SOA Application - Configure SOA settings Page



6. Select **Composite With BPEL** from the Composite Template list, and then click **Finish**. You have created a new application, and an SOA project. This automatically creates an SOA composite.

   The Create BPEL Process page is displayed.

The Create BPEL Process Page



7. Enter an appropriate name for the BEPL process in the **Name** field. For example, `POInboundXMLGateway`.

   Select **Asynchronous BPEL Process** in the **Template** field. Click **OK**.

   An asynchronous BPEL process is created with the Receive and Reply activities. The required source files including bpel and wsdl, using the name you specified (for example, `POInboundXMLGateway.bpel` and `POInboundXMLGateway.wsdl`) and `composite.xml` are also generated.

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.

To add a partner link:

1. Click **BPEL Services** in the Component palette.

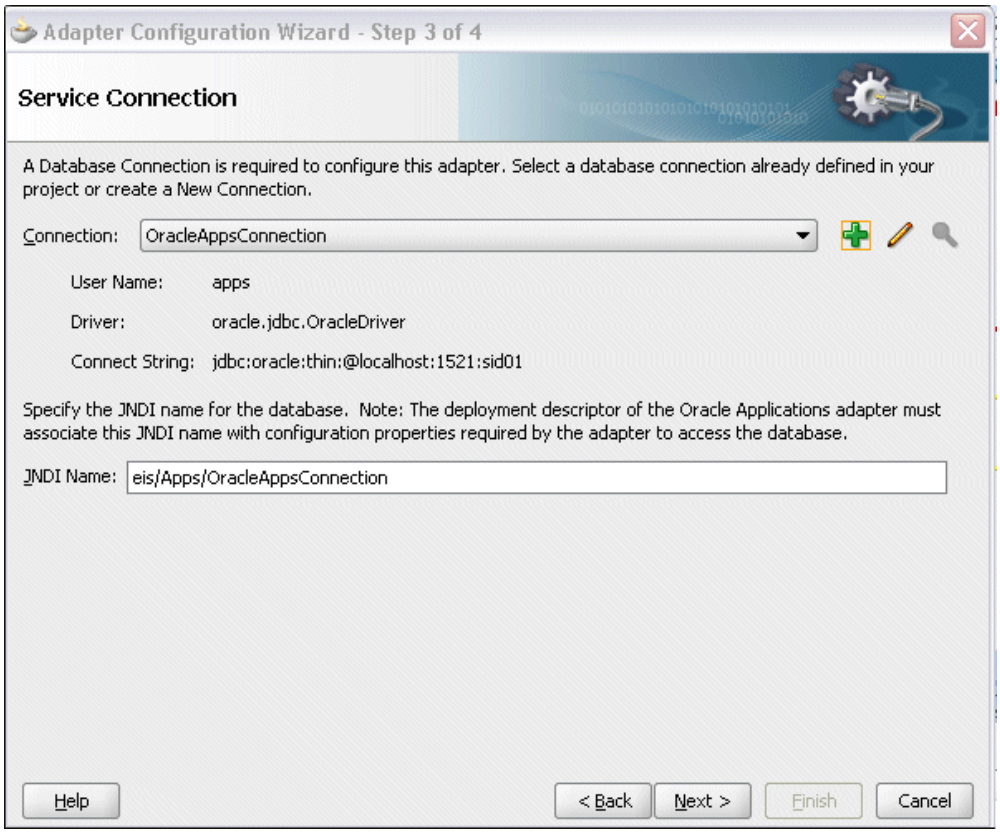
   Drag and drop **Oracle Applications** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears. Click **Next**.

2. Enter a service name in the **Service Name** field. For example,

`XMLGatewayOrderInbound`.

Click **Next**. The Service Connection dialog appears.

Specifying a Database Service Connection



3. You can either create a new database connection or use an existing connection.

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

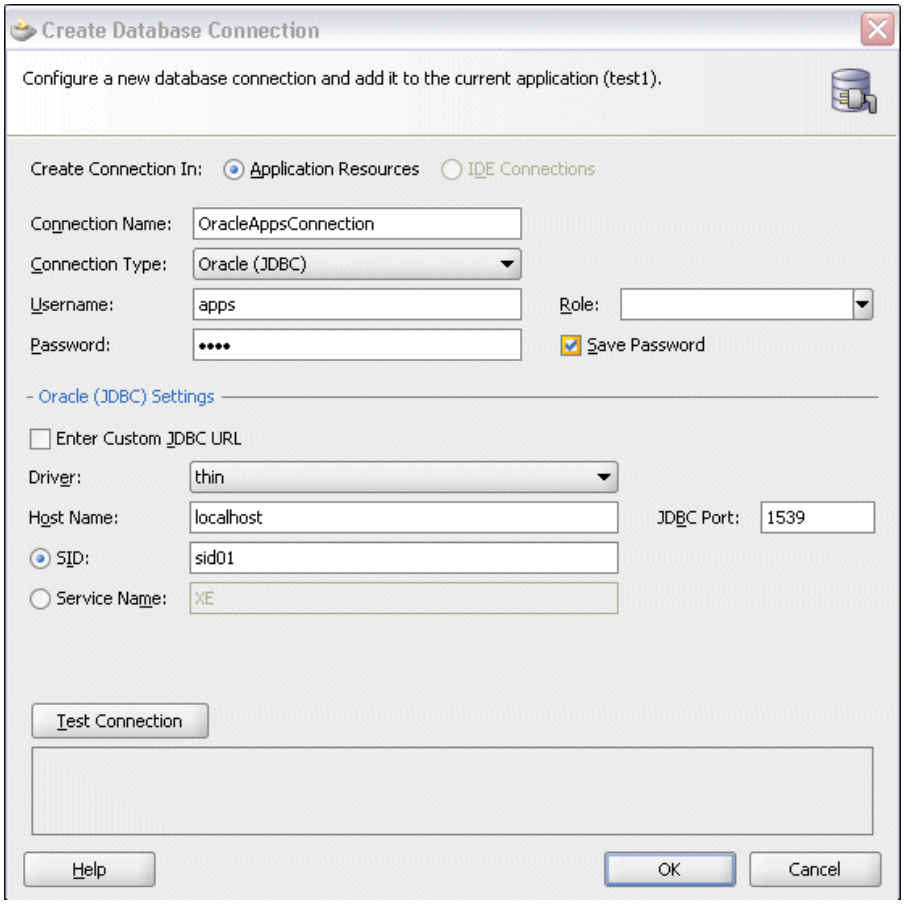
- **Creating a New Database Connection:**

  Complete the following steps to create a new database connection:

  1. From the Service Connection page, click the **Create a New Database Connection** icon.

     The Create Database Connection page appears.

Creating a Database Connection



2. Enter the following information:

    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 (such as `Oracle (JDBC)`).

    3. In the **UserName** field, specify a unique name for the database connection.

    4. In the **Password** field, specify a password for the database connection.

    5. In the Oracle (JDBC) Settings, from the **Driver** list, select **Thin**.

    6. In the **Host Name** field, specify the host name for the database connection.

7. In the **JDBC Port** field, specify the port number for the database connection.

8. In the **SID** field, specify the unique SID value for the database connection.

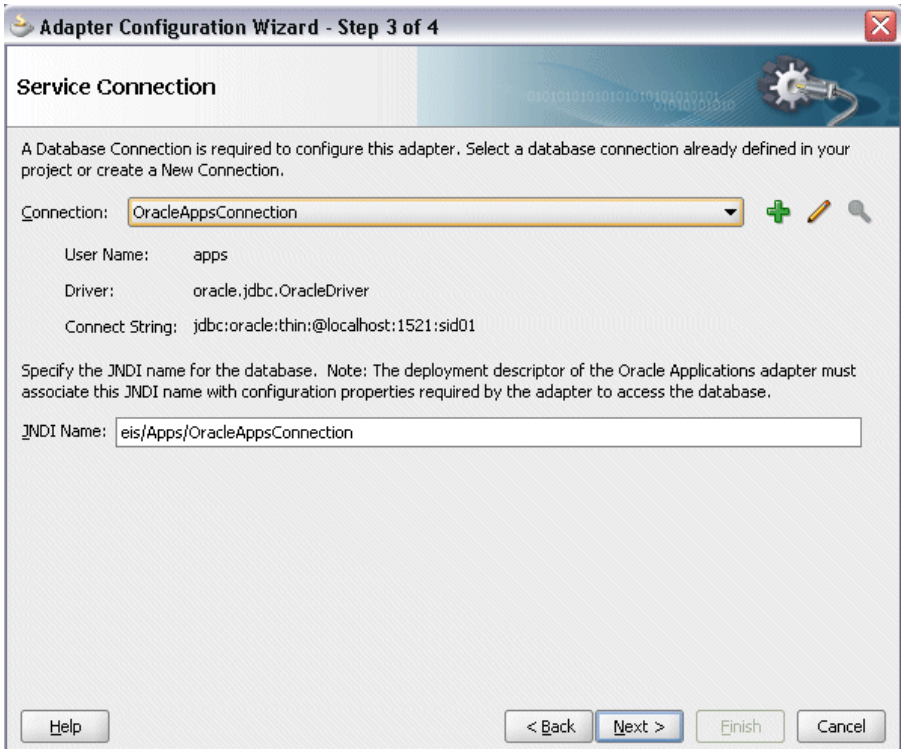
3. Click **Test Connection** to determine whether the specified information establishes a connection with the database.

   The status message "`Success!`" indicates a valid connection.

   Click **OK**.

4. The Service Connection dialog box appears, providing a summary of your database connection.

New Database Connection



- **Selecting an Existing Database Connection:**

  Instead of creating a new database connection, you can use an existing database connection that you have configured.

  1. From the Service Connection page, select an appropriate connection from the **Connection** drop-down list.

  2. The Service Connection page will be displayed with the selected connection information. The JNDI (Java Naming and Directory Interface) name corresponding to the database connection appears automatically in the **Database Server JNDI Name** field. Alternatively, you can specify a JNDI name.

     Note: When you specify a JNDI name, the deployment descriptor of the Oracle Applications adapter must associate this JNDI name with configuration properties required by the adapter to access the database.

     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.

> **Note:** For more information about JNDI concepts, refer to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

4. Once you have completed a new connection or selected an existing connection, you can add an XML Gateway inbound map by browsing through the list of APIs available in Oracle Applications.

   Click **Next**.

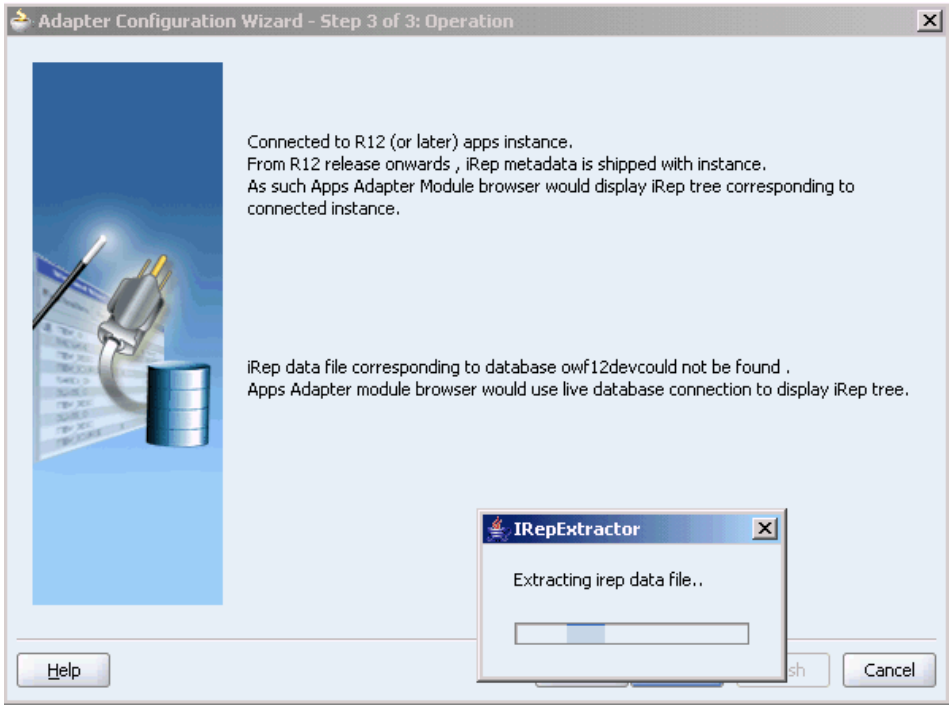
   For Oracle E-Business Suite Release 12:

   If you are connecting to Oracle E-Business Suite Release 12, then the **IREP File not present** dialog appears indicating that Adapter could not find the Oracle Integration Repository data file corresponding to the database you are connecting in your workspace. Absence of the data file would make browsing or searching of Integration Repository tree considerably slow. You can choose to extract the data file and create a local copy of the Integration Repository data file. Once it is created successfully, Adapter will pick it up automatically next time and retrieve data from your local Integration Repository.

   You can select one of the following options:

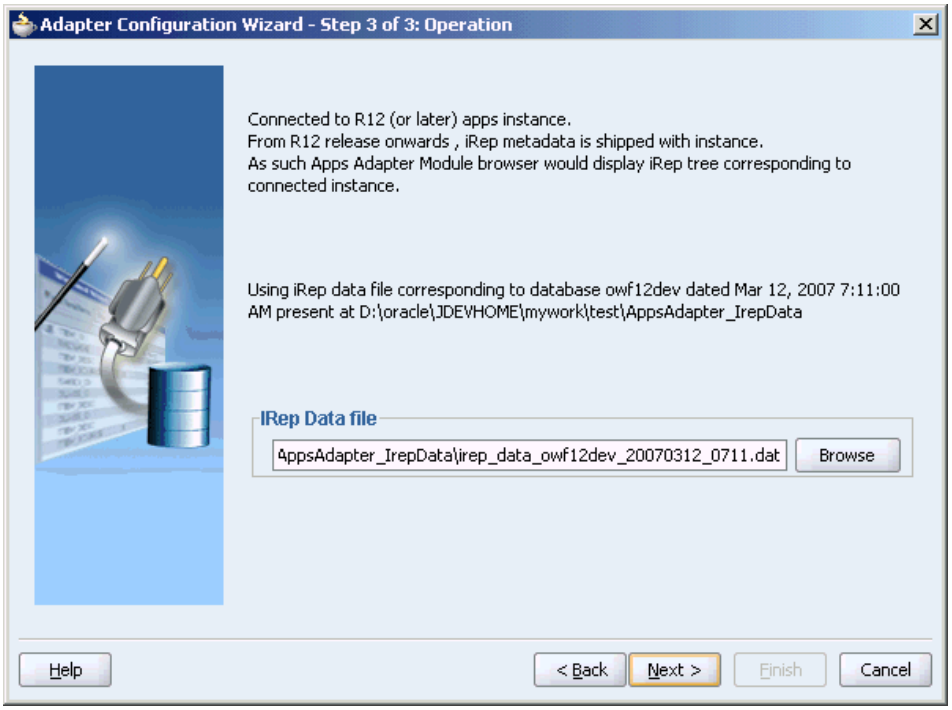
   - Click **Yes** to extract the Integration Repository data file.

Extracting Integration Repository Data File



After the system successfully creates a local copy of the Integration Repository data file, next time when you connect to the database, you will find the **IRep Data File** field appears in the Operation dialog indicating where your local copy exists with the creation date and time as part of the file name.

Using the Local Integration Repository Data File



- Click **No** to query the Integration Repository data file from the live database you are connecting to display the Integration Repository tree.

  Note: It is highly recommended that you create a local copy of the Integration Repository data file so that Adapter will query the data next time from the local copy in your workspace to enhance the performance.

  Click **Next** in the Operation page to open the Oracle Applications Module Browser.

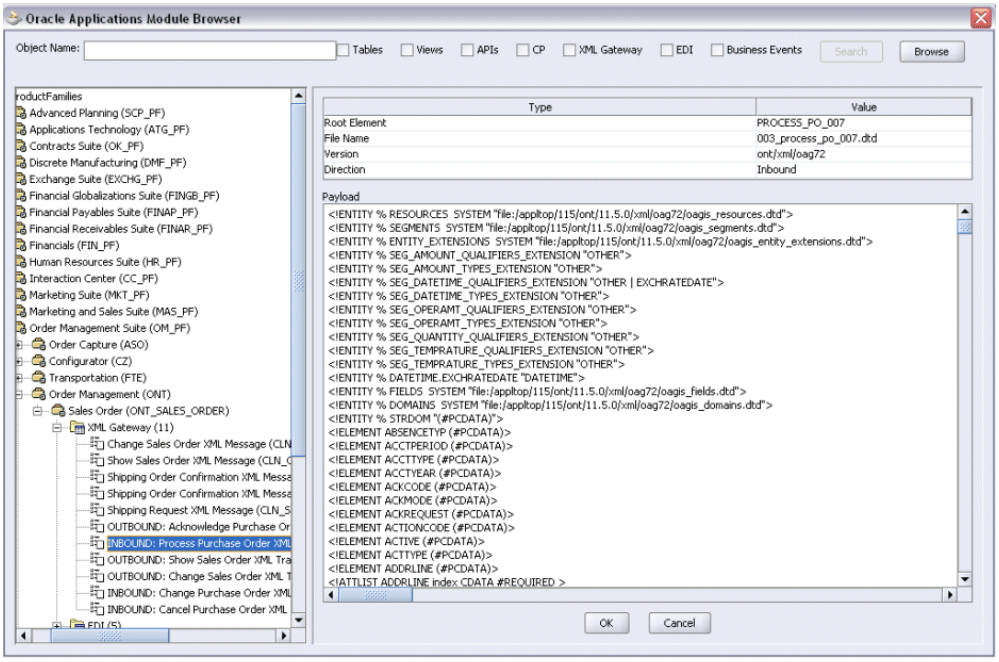
For Oracle E-Business Suite pre-Release 11.5.10:

If you are connecting to a pre-11.5.10 Oracle Applications instance, you must select the interface type in the Adapter Configuration Wizard. Select **XML Gateway** to proceed.

Click **Get Object** in the Application Interface dialog to open the Oracle Applications Module Browser.

5. The Oracle Applications Module Browser combines interface data from Oracle Integration Repository with information about the additional interfaces supported by Oracle Application Adapter, organized in a tree hierarchy.

Specify an Inbound XML Gateway Message Map from The Oracle Applications Module Browser

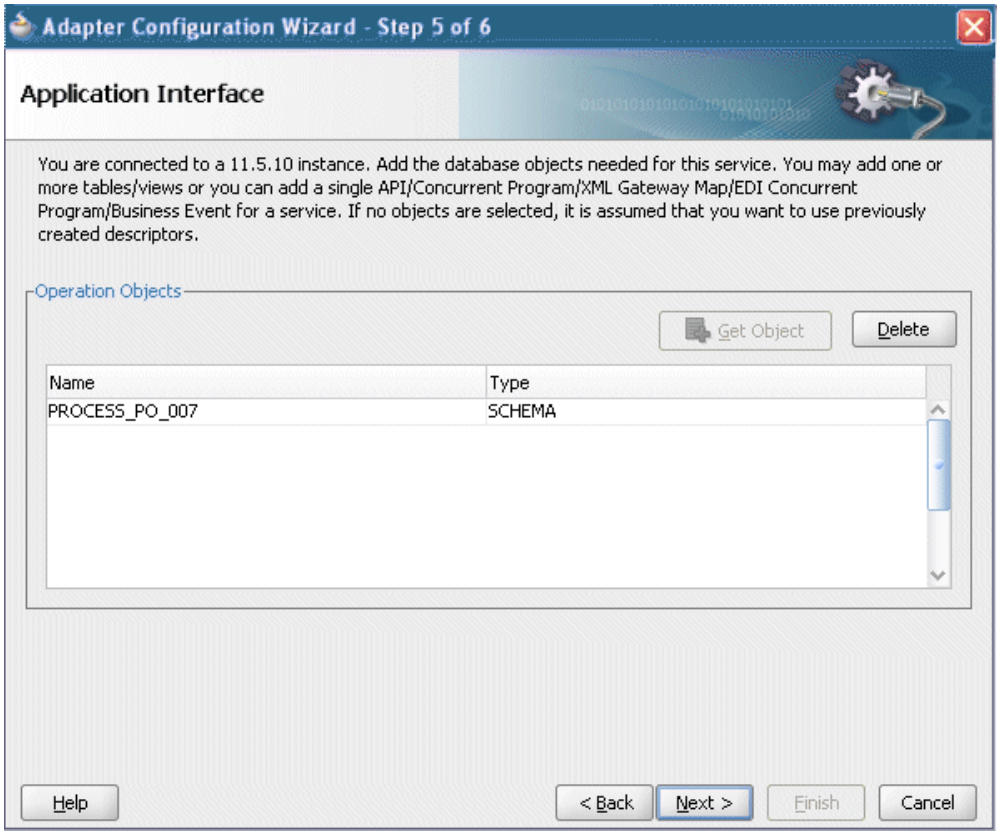


> **Note:** The Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. Each product family contains the individual products. Each product contains the business entities associated with the product. Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide.

Navigate to *Order Management Suite* > *Order Management* >*Sales Order* > *XML Gateway* to select `Inbound: Process Purchase Order XML Transaction (ONT_3A4R_OAG72_IN)`.

6. Click **OK**.

Adapter Configuration Wizard - Application Interface Page



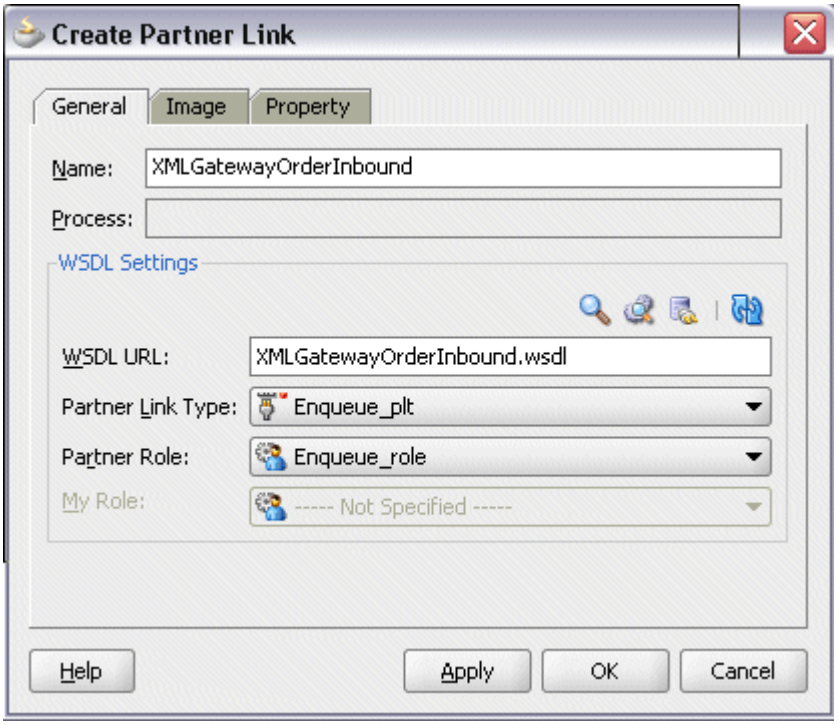
7. Click **Next**, then click **Finish** to complete the process of configuring Adapter for Oracle Applications.

   The wizard generates the WSDL file corresponding to the XML schema. This WSDL file is now available for the partner link.

8. Click **Apply** and then **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.

Partner Link Information



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.

To add a partner link for the file adapter to get the XML Message:

1. In JDeveloper BPEL Designer, click **BPEL Services** in the Component palette.

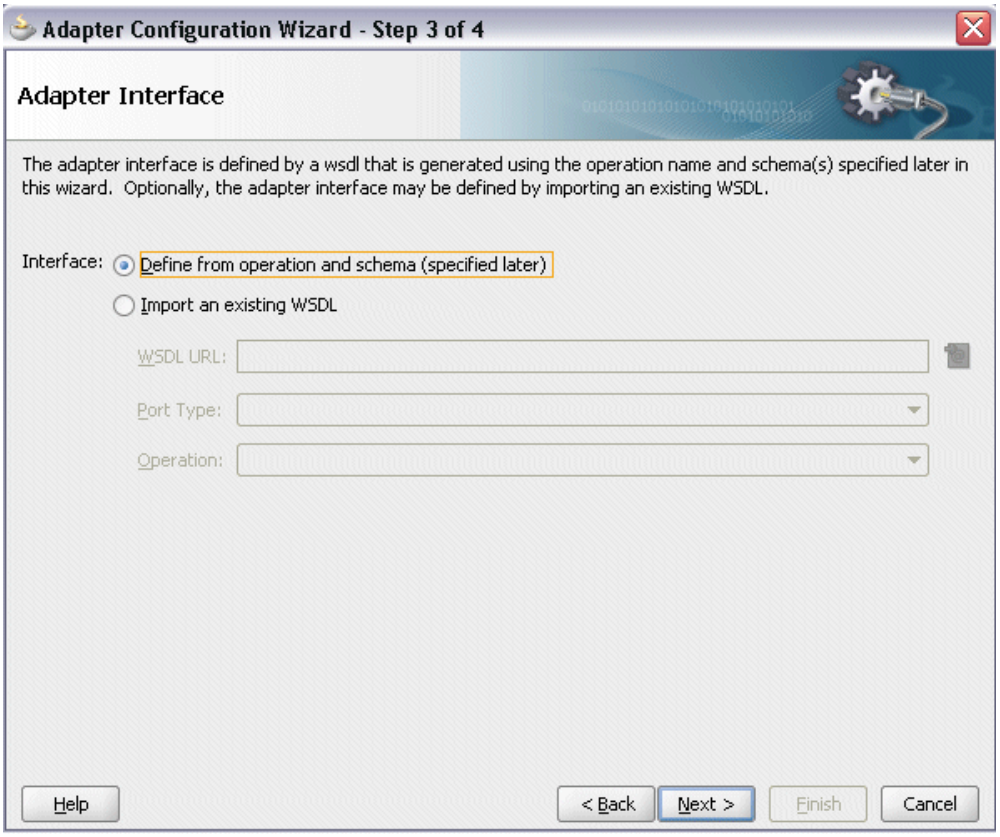
   Drag and drop **File Adapter** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears.

   Click **Next**.

2. In the Service Name page, enter a name for the file adapter service, for example, `getOrderXML`.

3. Click **Next**. The Adapter Interface page appears.

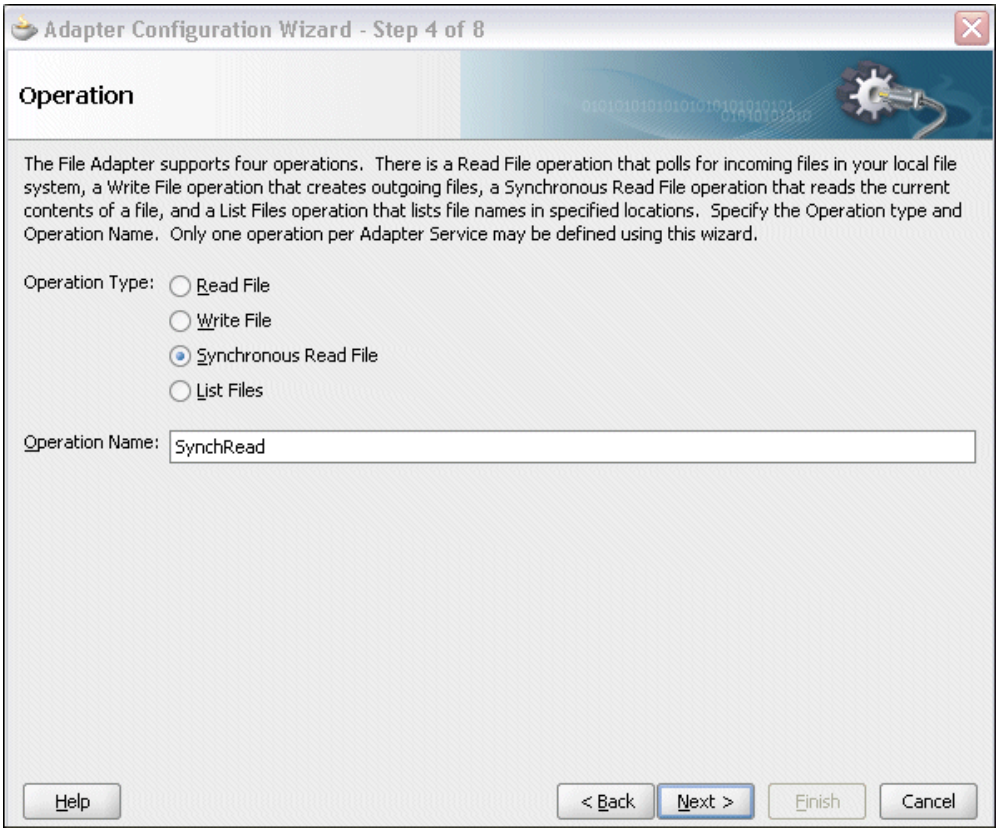
Specifying the Adapter Interface



Select the **Define from operation and schema (specified later)** radio button and click **Next**.

4. In the Operation page, specify the operation type. For example, select the **Synchronous Read File** radio button. This automatically populates the **Operation Name** field.

Specifying the Operation

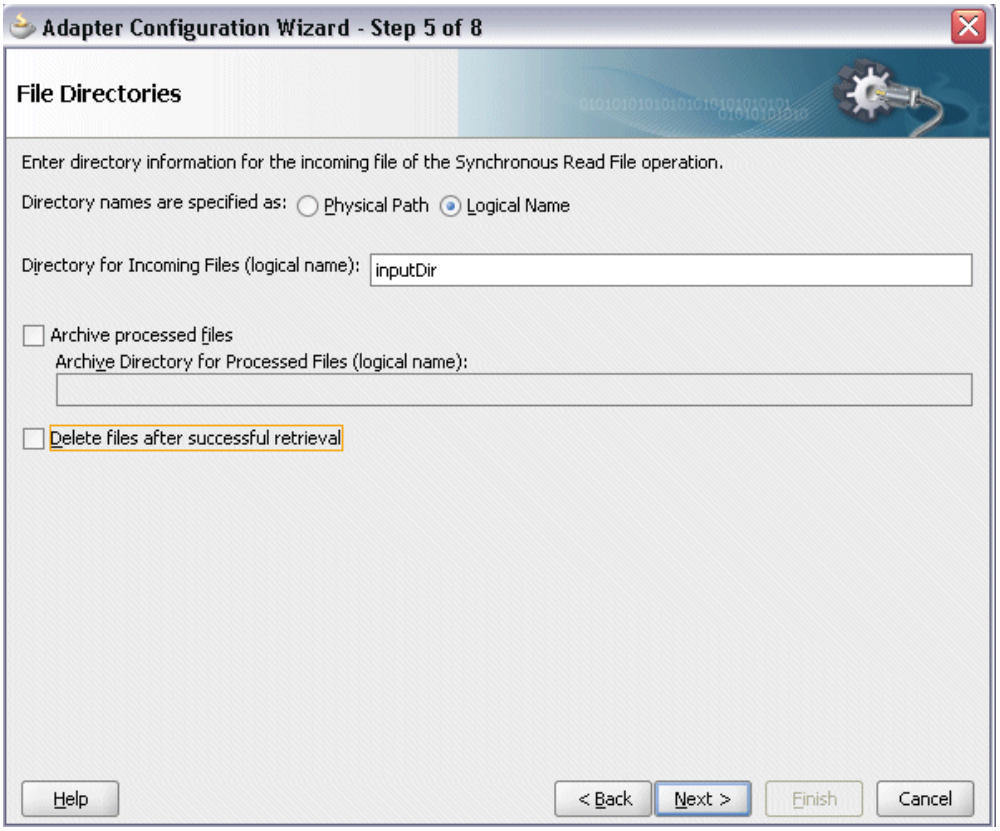


Click **Next** to access the File Directories page.

5. Select the **Logical Name** radio button and specify directory for incoming files, such as `inputDir`.

   Ensure the **Delete files after successful retrieval** check box is not selected.

Configuring the Input File



Click **Next** to open the File Name page.

6. Enter the name of the file for the synchronous read file operation. For example, enter `order_data_xmlg.xml`.

   Note: Use the following information to edit `composite.xml` to specify the physical directory for the File Adapter:

   ```
   <property name="inputDir" type="xs:string"
   many="false" override="may">/usr/tmp/</property>
   ```

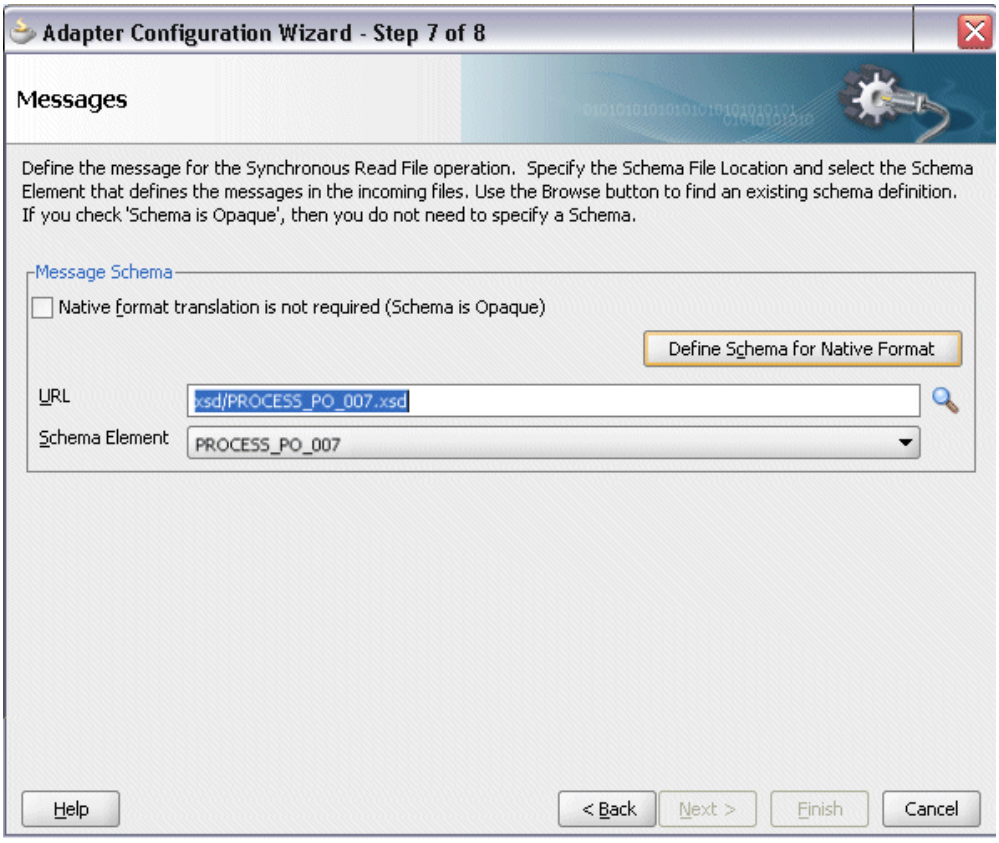
   Click **Next**. The Messages page appears.

7. Select the 'browse for schema file' icon to open the Type Chooser window.

   Expand the node by clicking **Project Schema Files > PROCESS_PO_007.xsd > PROCESS_PO_007**.

   The selected schema information will be automatically populated in the URL and Schema Element fields.

Specifying Message Schema



8. Click **Next** and then **Finish**. The wizard generates the WSDL file corresponding to the partner link. The main Create Partner Link dialog box appears, specifying the new WSDL file `getOrderXML.wsdl`.

   Click **Apply** and **OK** to complete the configuration and create the partner link with the required WSDL settings for the File Adapter service.

   The `getOrderXML` Partner Link appears in the BPEL process diagram.

Configuring the Invoke Activities

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

Based on the scenario described earlier, you need to configure the following two Invoke activities:

1. To get the XML message details from the input XML file through synchronous read operation by invoking the `getOrderXML` partner link.

2. To enqueue the purchase order information to the ECX_INBOUND queue by invoking `XMLGatewayOrderInbound` partner link in an XML file.

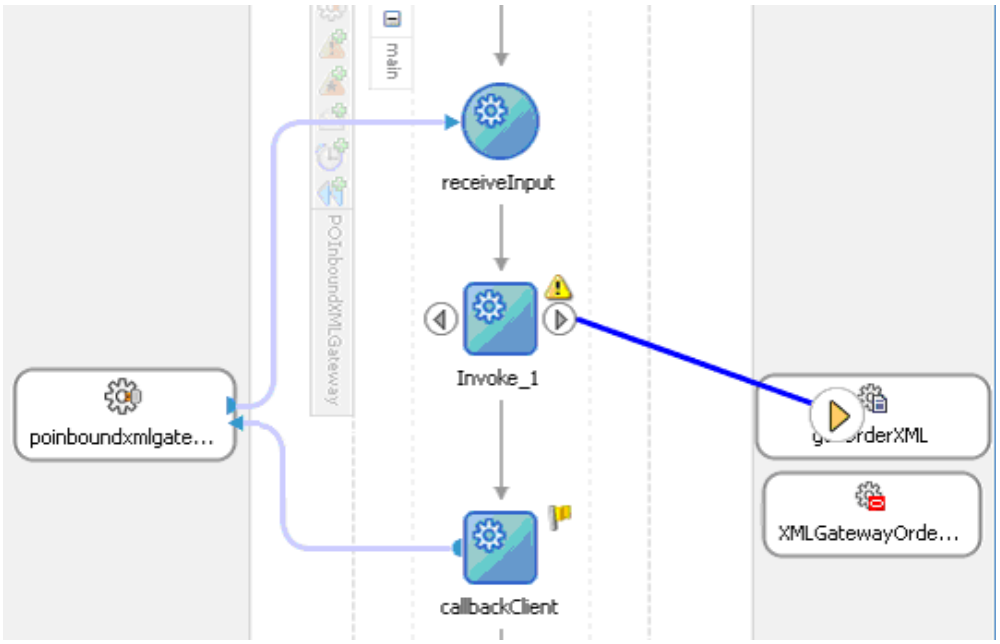
   The ECX Header properties for the XML Gateway inbound service can also be set through the Invoke activity.

To configure the first Invoke activity:

1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the first **Invoke** activity into the center swim lane of the process diagram, between the **receiveInput** and **callbackClient** activities.

2. Link the Invoke activity to the `getOrderXML` partner link service.

Creating an Invoke Activity



3. In the Edit Invoke dialog, enter a name for the Invoke activity in the General tab.

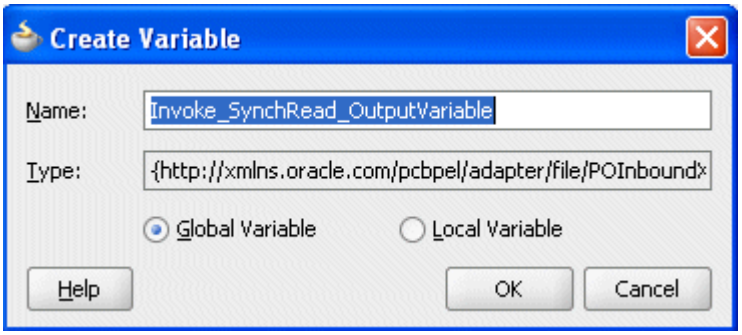
   The value of the Operation field is automatically selected based on the associated partner link. For example, this Invoke activity is to invoke a File Adapter service to synchronize read an input XML file; thus, the SynchRead is automatically populated in the Operation field.

4. Click the **Create** icon next to the **Input Variable** field. The Create Variable dialog appears.

   Select **Global Variable** and enter a name for the variable. You can also accept the default name. Click **OK**.

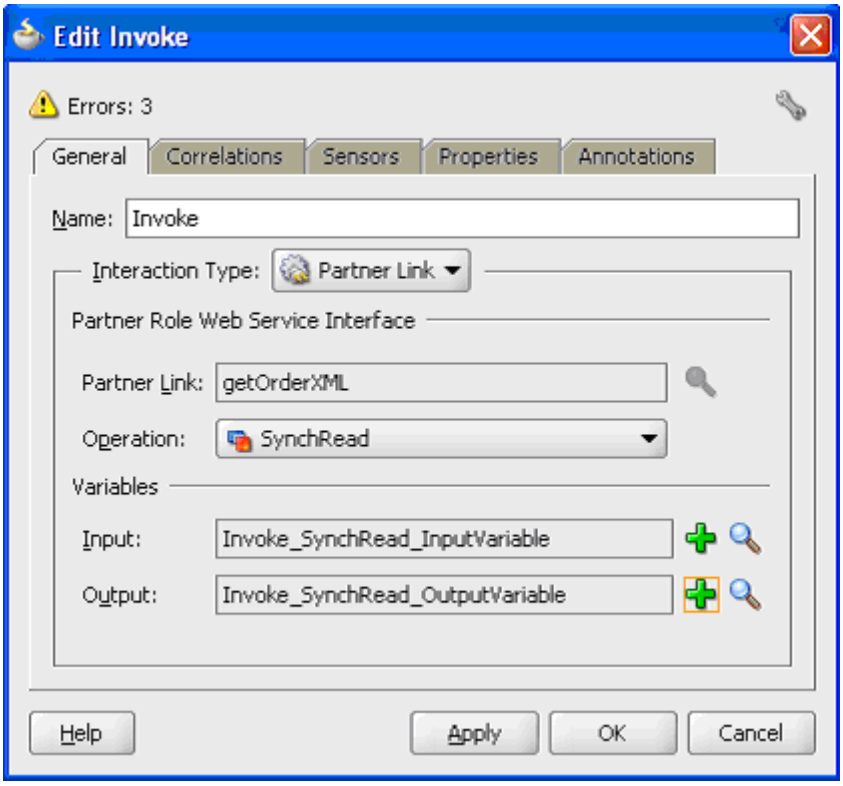
5. In the Edit Invoke dialog, click the **Create** icon next to the **Output Variable** field to create a new variable. The Create Variable dialog box appears.

Creating the Input Variable



Select **Global Variable** and enter a name for the variable. You can also accept the default name. Click **OK** to return to the Edit Invoke dialog.

The Edit Invoke Dialog



6. Click **Apply** and then **OK**.

To configure the second Invoke activity:

1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the first **Invoke** activity into the center swim lane of the process diagram, between the first **Invoke** activity and **callbackClient** activity.

2. Link the Invoke activity to the `XMLGatewayOrderInbound` partner link service.

3. In the Edit Invoke dialog, enter a name for the Invoke activity in the General tab.

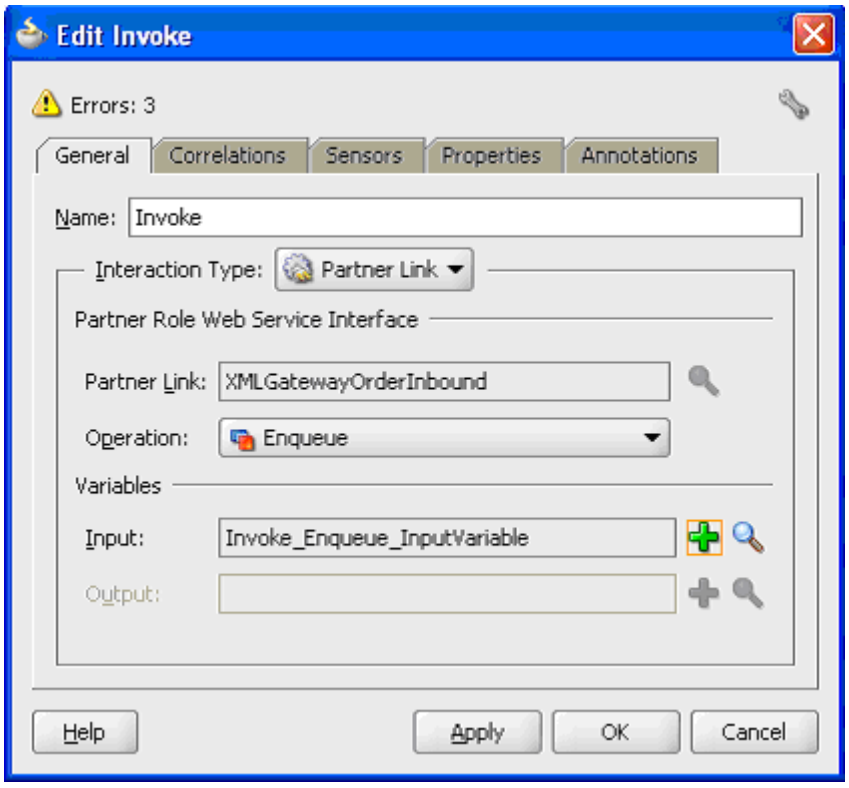
   The value of the Operation field is automatically selected based on the associated partner link. Since this Invoke activity is associated with an inbound message map partner link, the Enqueue operation is selected.

4. Click the **Create** icon next to the **Input Variable** field. The Create Variable dialog appears.

   Select **Global Variable** and enter a name for the variable. You can also accept the default name. Click **OK**.

   Click **Apply**.

The Edit Invoke Dialog



5. **Setting ECX Header Message Properties**

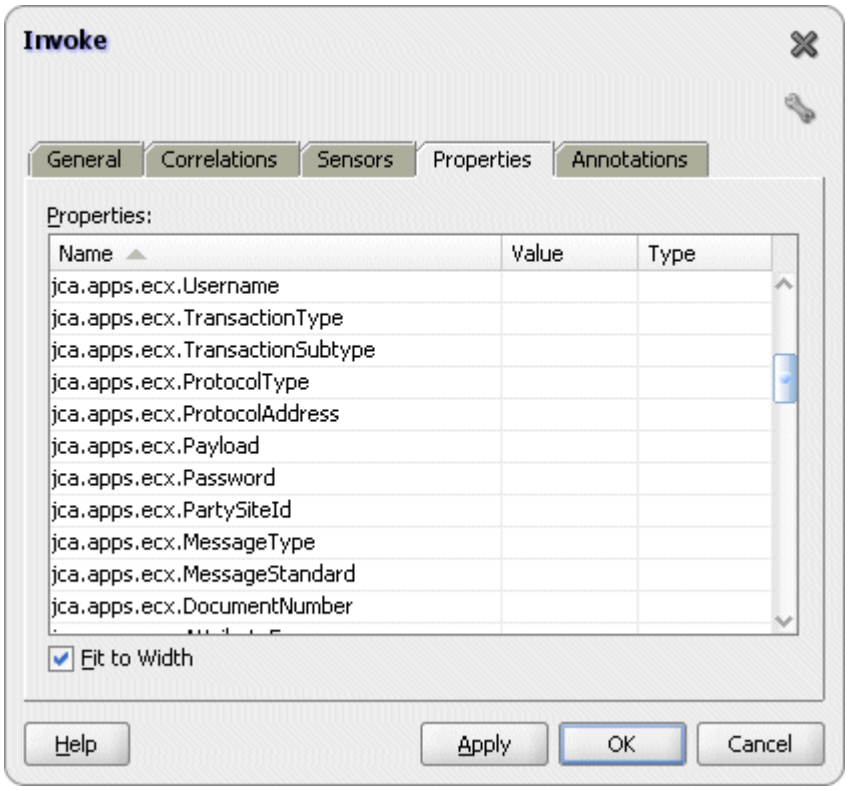
You need to enter appropriate XML Gateway header property values in order to pass individual message properties required for XML transaction to complete successfully.

Use the following steps to set ECX header message properties:

1. Click the **Properties** tab in the Invoke dialog box. You will find all predefined normalized message properties.

   Locate XML Gateway specific properties with prefix `jca.apps.ecx` from the Properties tab.

Locating XML Gateway Related Properties



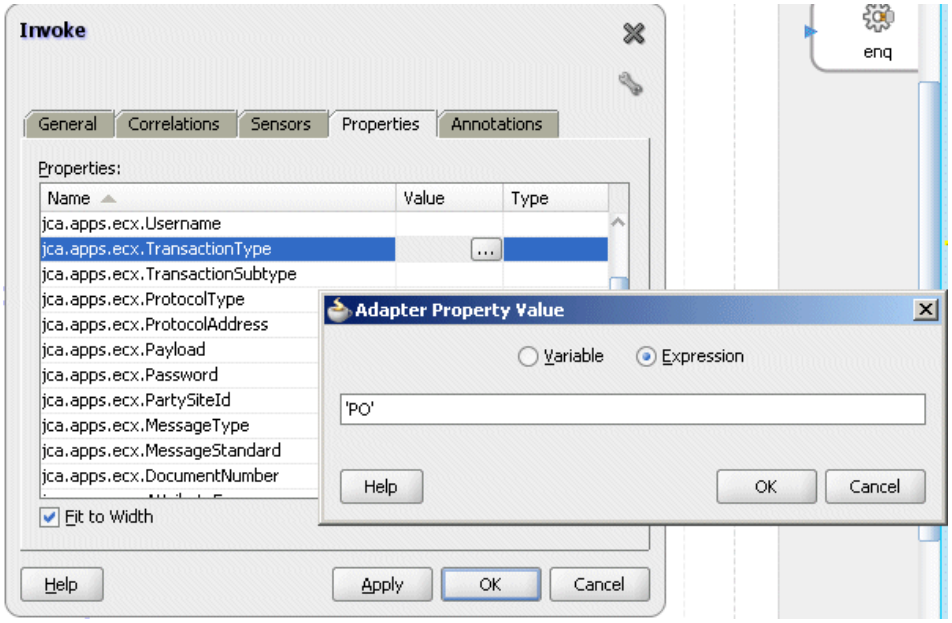
2. Click a property first (such as `jca.apps.ecx.TransactionType`) from the Properties tab to enable the **Adapter Property Value** icon.

3. Click the icon to open the Adapter Property Value dialog.

4. Select the 'Expression' button as the message type.

5. Enter a property value for the selected property. For example, enter `'PO'` for the property `jca.apps.ecx.TransactionType`.

Entering a Selected Property Value



Click **OK**.

6. Repeat Step 2 to Step 5 to set the following property values:

   - `jca.apps.ecx.MessageType: 'XML'`
   - `jca.apps.ecx.TransactionSubType: 'PROCESS'`
   - `jca.apps.ecx.PartySiteId: 'BWSANJOSE'`
   - `jca.apps.ecx.MessageType: 'XML'`
   - `jca.apps.ecx.MessageStandard: 'OAG'`
   - `jca.apps.ecx.DocumentNumber: 'order_xml_01'`

   Note: In earlier releases, XML Gateway header variables including `MESSAGE_TYPE`, `MESSAGE_STANDARD`, `TRANSACTION_TYPE`, `TRANSACTION_SUBTYPE`, `DOUCMENT_NUMBER`, and `PARTY_SITE_ID` are contained in the `SYSTEM.ECXMSG` object. In this release, Adapter for Oracle Applications normalize the `SYSTEM.ECXMSG` object to create and set each individual message property through the Properties tab of the Invoke activity. These header properties are required for the XML transaction to complete successfully.

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

Configuring the Assign Activity

Use the Assign activity to pass the output of `getOrderXML` service as an input to the `XMLGatewayOrderInbound` service.

To add an Assign activity:

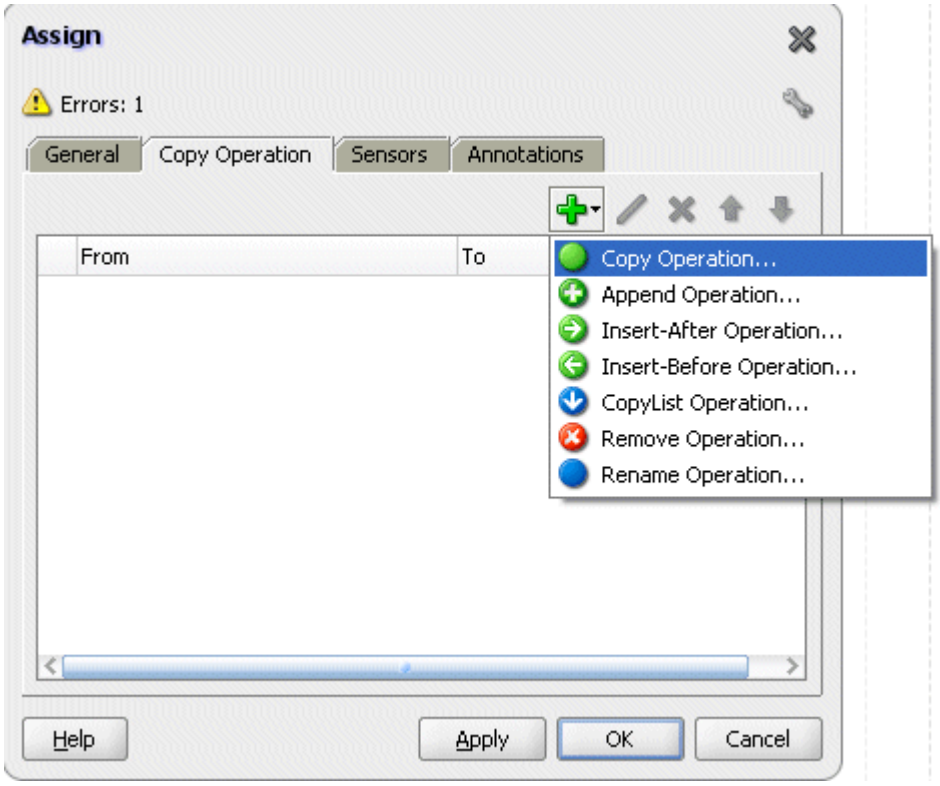
1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the **Assign** activity into the center swim lane of the process diagram between the two **Invoke** activities you just created earlier.

2. Double-click the **Assign** activity to access the Edit Assign dialog.

   Click the General tab to enter a name for the Assign activity. For example, `SetOrderXML`.

3. Select the Copy Operation tab, click the 'Plus' sign icon and choose **Copy Operation** from the menu. The Create Copy Operation window appears.

Specifying a Copy Operation Action

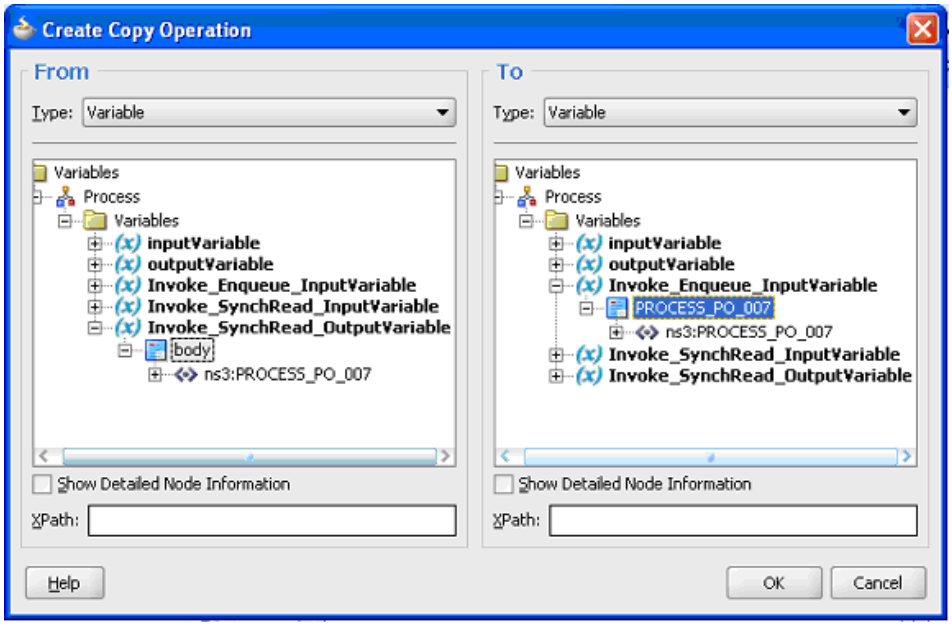


4. Enter the following information:

   - In the From navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_SynchRead_OutputVariable > body** and select **ns3:PROCESS_PO_007**.

   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_Enqueue_InputVariable > PROCESS_PO_007** and select **ns3:PROCESS_PO_007**. The XPath field should contain your selected entry.

Create Copy Operation Dialog

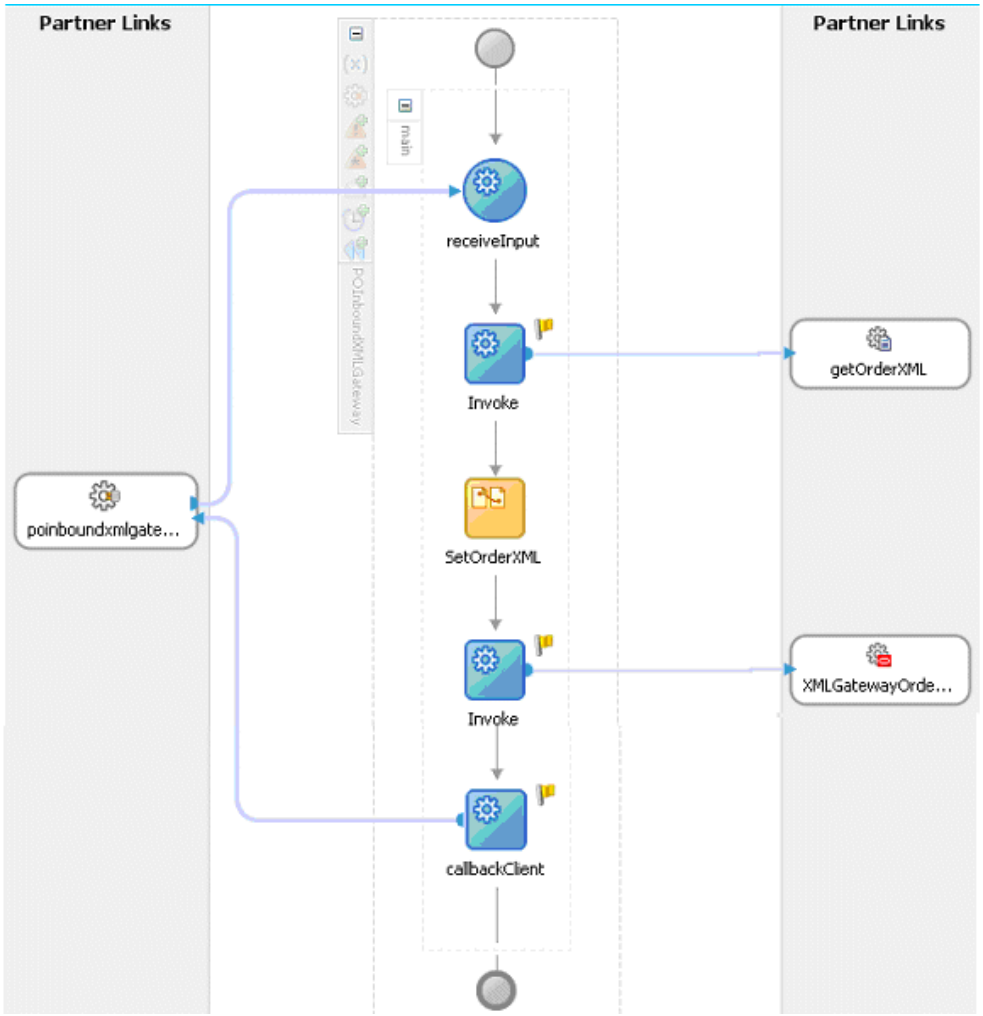


- Click **OK**. The Edit Assign dialog appears.

5. Click **Apply** and **OK** to complete the configuration of the Assign activity.

   The following diagram illustrates the complete BPEL process:

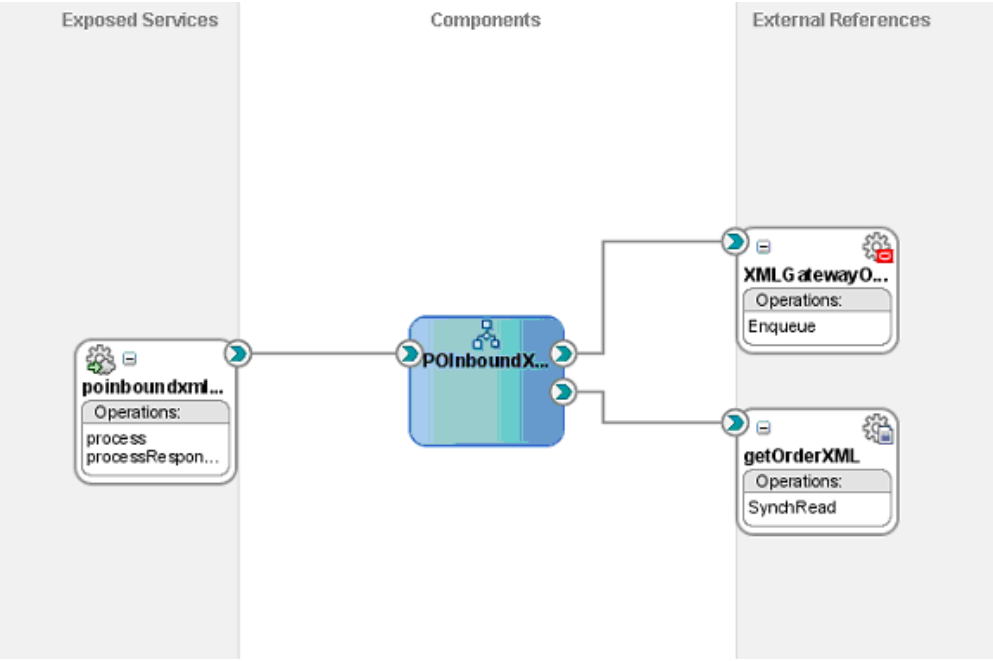
BPEL Process Diagram



Click the `composite.xml` to display the Oracle JDeveloper composite diagram:

> **Note:** Click the Source tab of `composite.xml` to enter a value for the `inputDir` (such as `/usr/tmp`).
>
> ```
> <property name="inputDir" type="xs:string"
> many="false" override="may">/usr/tmp</property>
> ```

Oracle JDeveloper composite Diagram



Run-Time Tasks for XML Gateway Inbound Messaging

After designing the BPEL process, the next step is to deploy, run and monitor it.

1. Deploy the BPEL process., page 4-36
2. Test the BPEL process., page 4-38
3. Verify records in Oracle Applications., page 4-40

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 application server (Oracle WebLogic Server) that you have established the connection.

Prerequisites

Before deploying the BPEL process using Oracle JDeveloper, you must ensure the following:

- You must have established the connectivity between the deign-time environment and an application server.

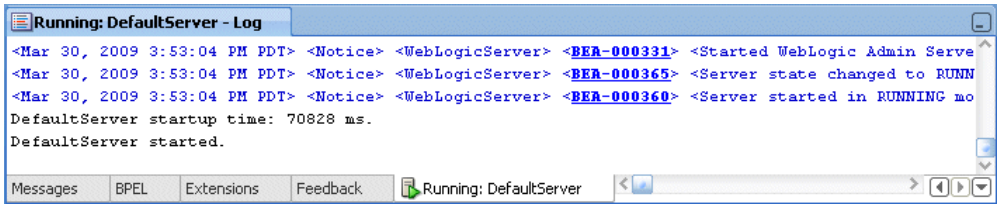
For more information, see Configuring the Data Source in Oracle WebLogic Server, page A-3 and Creating an Application Server Connection, page A-8.

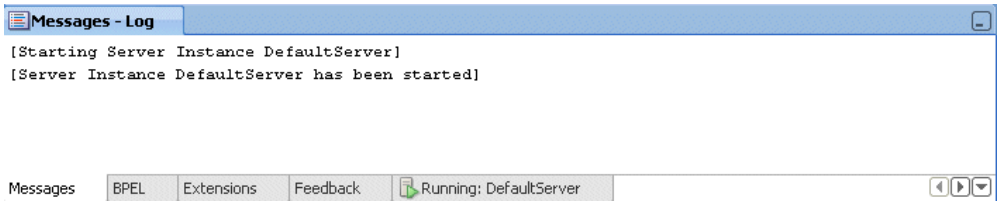
- Oracle WebLogic Server has been started.

  Before deploying the BPEL process, you need to start the Oracle WebLogic Server that you have established the connection.

  If a local instance of the WebLogic Server is used, start the WebLogic Server by selecting **Run > Start Server Instance** from Oracle JDeveloper.

  Once the WebLogic Admin Server "DefaultServer" instance is successfully started, you should find that <Server started in Running mode> and DefaultServer started message in the Running:DefaultServer and Messages logs.



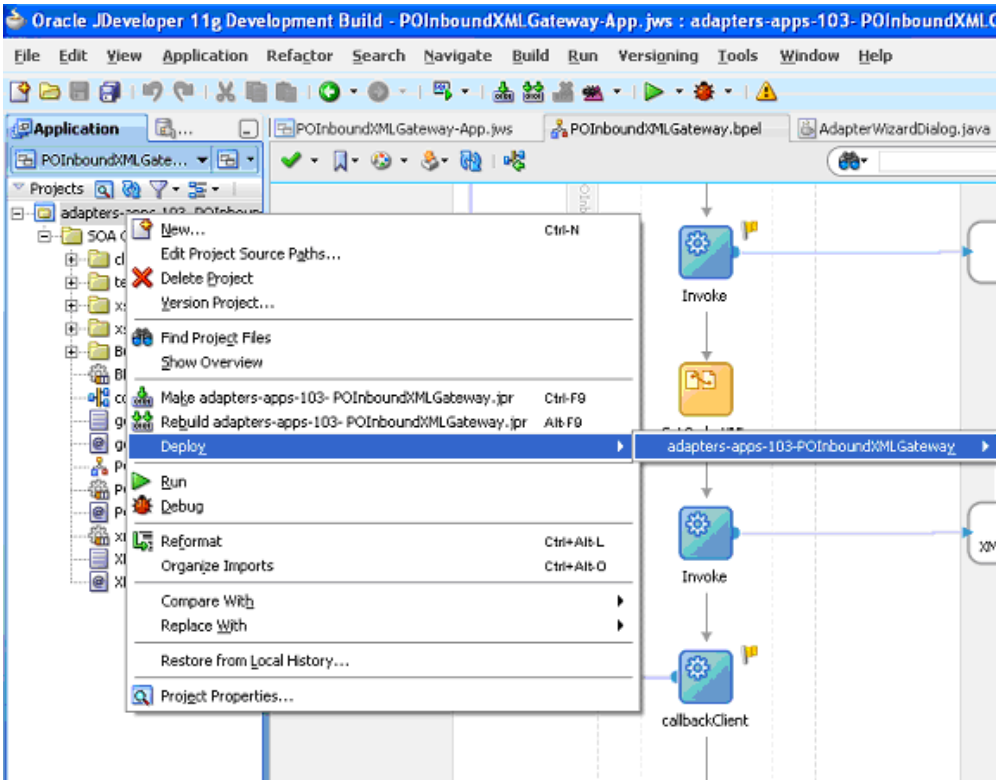


To deploy the BPEL process:

1. Select the BPEL project in the Applications Navigator.

2. Right-click the project name. Select **Deploy > [project name] > [serverConnection]** from the menu that appears.

Deploying the BPEL Process



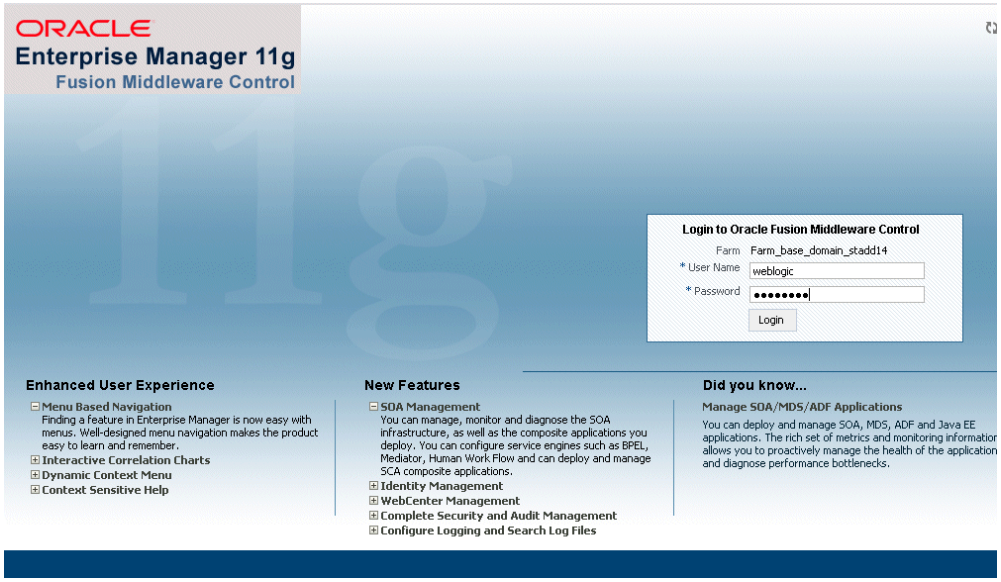
3. The BPEL process is compiled and deployed. You can check the progress in the Messages window.

Testing the BPEL Process

Once the BPEL process is deployed, you can manage and monitor the process from the Oracle Enterprise Manager Fusion Middleware Control Console. You can also test the process and the integration interface by manually initiating the process.

To test the BPEL process:

1. Navigate to Oracle Enterprise Manager Fusion Middleware Control Console (`http://<servername>:<portnumber>/em`). The composite you deployed is displayed in the Applications Navigation tree.



2. Enter username (such as `weblogic`) and password and click **Login** to log in to a farm.

   You may need to select an appropriate target instance farm if there are multiple target Oracle Enterprise Manager Fusion Middleware Control Console farms.

3. From the Farm base domain, expand the **SOA** >**soa-infra** to navigate through the SOA Infrastructure home page and menu to access your deployed SOA composite applications running in the SOA Infrastructure for that managed server.

   > **Note:** The Farm menu always displays at the top of the navigator. As you expand the SOA folder in the navigator and click the links displayed beneath it, the SOA Infrastructure menu becomes available at the top of the page.

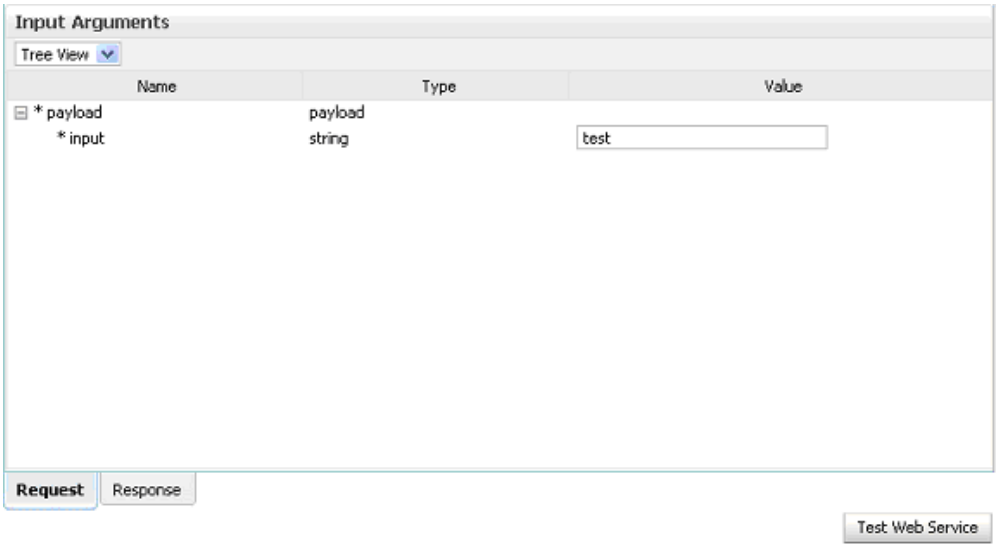
   Click the SOA composite application that you want to initiate (such as 'POInboundXMLGateway') from the SOA Infrastructure.

   Click **Test** at the top of the page.

4. The Test Web Service page for initiating an instance appears. You can specify the XML payload data to use in the Input Arguments section.

   Enter the input string required by the process and click **Test Web Service** to initiate the process.

Testing Web Service



The test results appear in the Response tab upon completion.

5. Click the Instances tab. The SOA composite application instance ID, name, conversation ID, most recent known state of each instance since the last data refresh of the page are displayed.

   In the Instance ID column, click a specific instance ID to show the message flow through the various service components and binding components. The Flow Trace page is displayed.

   In the Trace section, you should find the sequence of the message flow for the service binding component (`poinboundxmlgateway_client_ep`), BPEL component (`POInboundXMLGateway`), and reference binding components (`getOrderXML` and `XMLGatewayOrderInbound`). All involved components have successfully received and processed messages.

   If any error occurred during the test, you should find it in the Faults section.

6. Click your BPEL service component instance link (such as `POInboundXMLGateway`) to display the Instances page where you can view execution details for the BPEL activities in the Audit Trail tab.

   Click the Flow tab to check the BPEL process flow diagram. Click an activity of the process diagram to view the activity details and flow of the payload through the process.

Verifying Records in Oracle Applications

Once the BPEL process is successfully initiated and completed, you can validate it

through the relevant module in Oracle Applications.

For example, you can validate it using Oracle Transaction Monitor to ensure the process is successfully completed, and then import the order to Order Management application. You can then verify it in Oracle Order Management to ensure the order does exist.

To validate it Using Oracle Transaction Monitor

Additionally, you can also validate it from the 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. It 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:

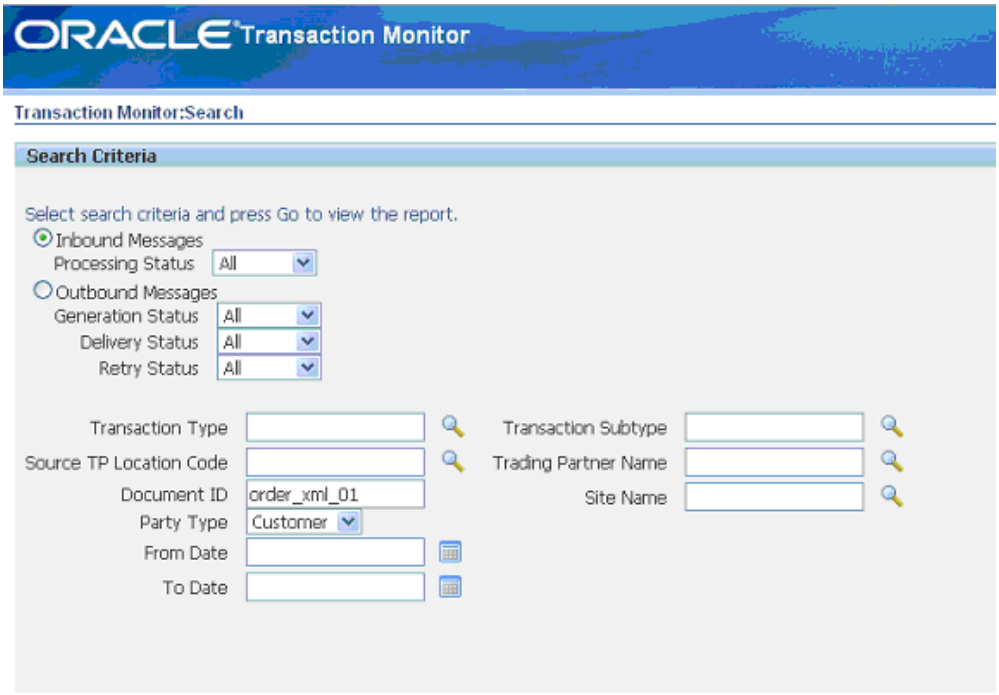
- Flexible search criteria to support access to a specific document or group of documents
- Search results at the document header level with drill down by document ID
- Resend capability for outbound messages
- Viewing capability of the XML message content

> **Note:** For details on using the Transaction Monitor, see *Oracle XML Gateway User's Guide.* This guide is 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

Use the following steps to validate in Oracle Transaction Monitor:

1. Log on to Oracle Applications with the Workflow Administrator Web Applications responsibility. Select Transaction Monitor to open the search window to search for the order.
2. Select the **Inbound Messages** radio button.

Searching from the Transaction Monitor



Enter the following information in the search window:

- Party Type: Customer
- Document ID: Enter the document ID you defined for the header message property. For example, enter `oder_xml_01`.

3. Click **Go** to retrieve all XML inbound messages listed in the Inbound Search Results region.

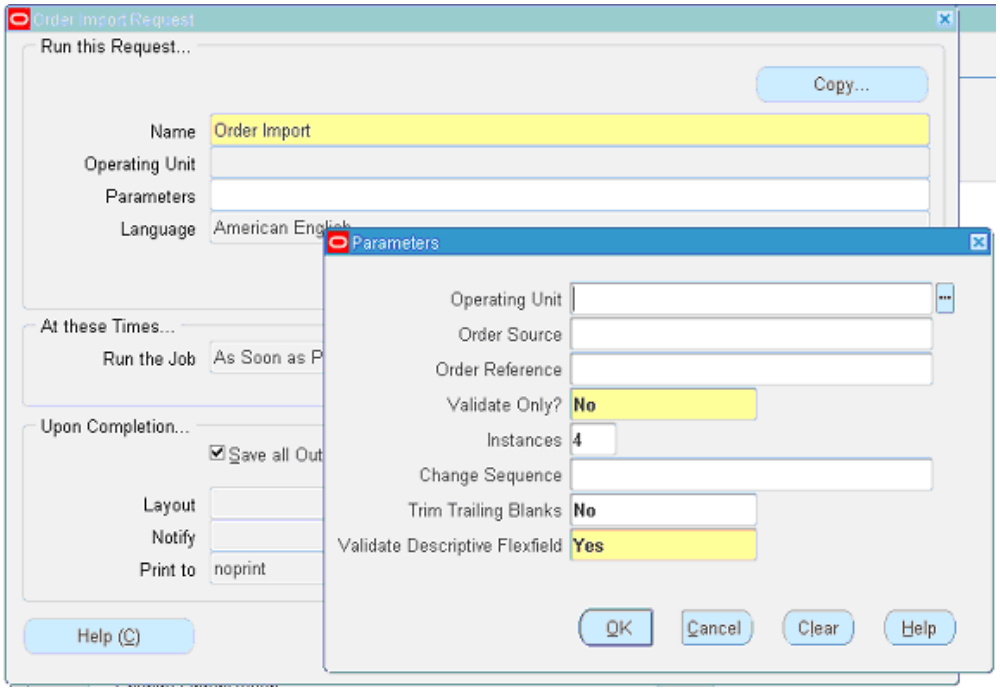
   Confirm that transaction for 'order_xml_01' has the 'Success' status.

To import the order to Oracle Order Management:

1. Log on to the Forms-based Oracle Applications with the Order Management Super User, Vision Operations (USA) responsibility.

2. Select Orders, Returns : Import Orders > Order Import Request. The Order Import Request window is displayed along with the Parameters dialog.

Order Import Request



3. Click **OK** in the parameters dialog. The Order Import request name is populated automatically in the Import Request window.

4. Click **Submit** to submit the request. This displays a concurrent request number. Record the request number, but click **No** in the Decision dialog that you will not submit another request.

5. From the application menu, select View >Requests to open the Find Requests window.

6. Enter the request number you recorded earlier and click **Find**.

   This would show the status of Order Import request. It should be success and the order should be created in Order Management application.

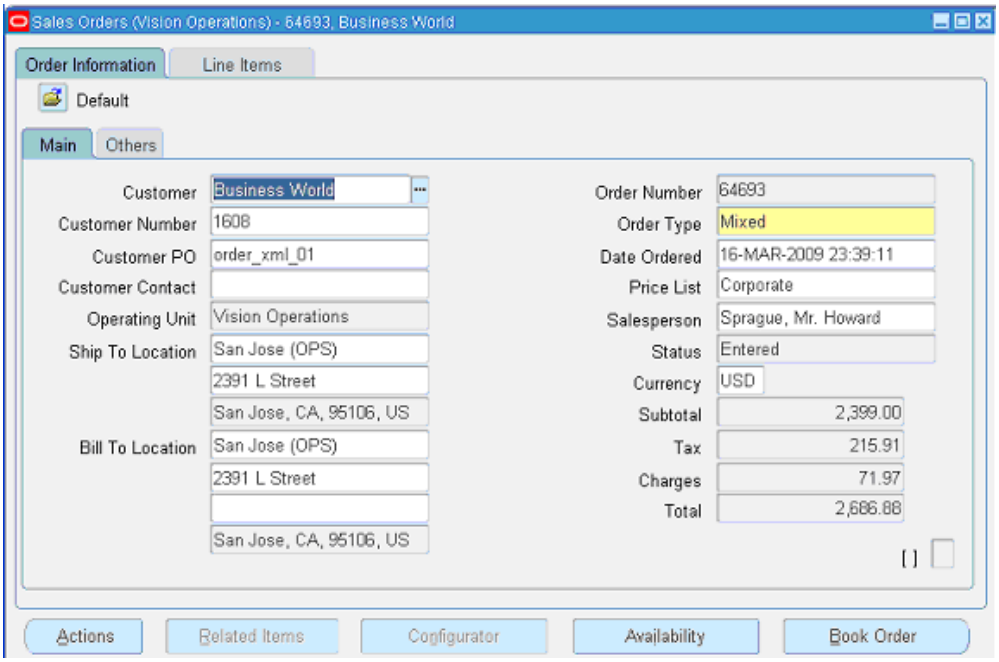
To validate it in Oracle Order Management:

1. Log on to the Forms-based Oracle Applications with the Order Management, Super User responsibility.

2. Select Order Returns > Sales Order. Sales Order Forms would open up.

3. Search for an order by entering the order number (`order_xml_01`) in the Customer PO field. This would bring up the details of a newly created order using XML Gateway inbound map.

Sales Orders



You can also select the Items tab for item details.

Design-Time Task for XML Gateway Outbound Messaging

For an outbound XML Gateway Map interface, since an outbound message is first enqueued to the ECX_OUTBOUND queue, Oracle BPEL PM listens to ECX_OUTBOUND queue for the message with the same correlation Id BPEL. The message will then be dequeued to retrieve outbound data and then the outbound map will be invoked to update Oracle Applications.

BPEL Process Scenario

Take XML Gateway outbound interface PO acknowledgement XML Transaction as an example. The XML Gateway outbound interface is exposed as a Web service through `ECX_CBODO_OAG72_OUT_CONFIRM` outbound map.

When a purchase order is created and approved, on approval of the purchase order, a workflow will be triggered which creates the Purchase Order Acknowledgement flow and sends out the PO Acknowledgement as an XML file. The workflow delivers the Confirm BOD as the PO Acknowledgement to ECX_OUTBOUND queue for delivery to the other system.

The correlation Id for this message is set to "BPEL" and the Oracle BPEL PM listens to ECX_OUTBOUND queue for the message with the correlation Id "BPEL". Confirm BOD as the PO Acknowledgement is written as an output XML file using File Adapter.

Prerequisites to Configure XML Gateway Outbound

Setting Up Correlation Identifier

For invoking an outbound message map, 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.

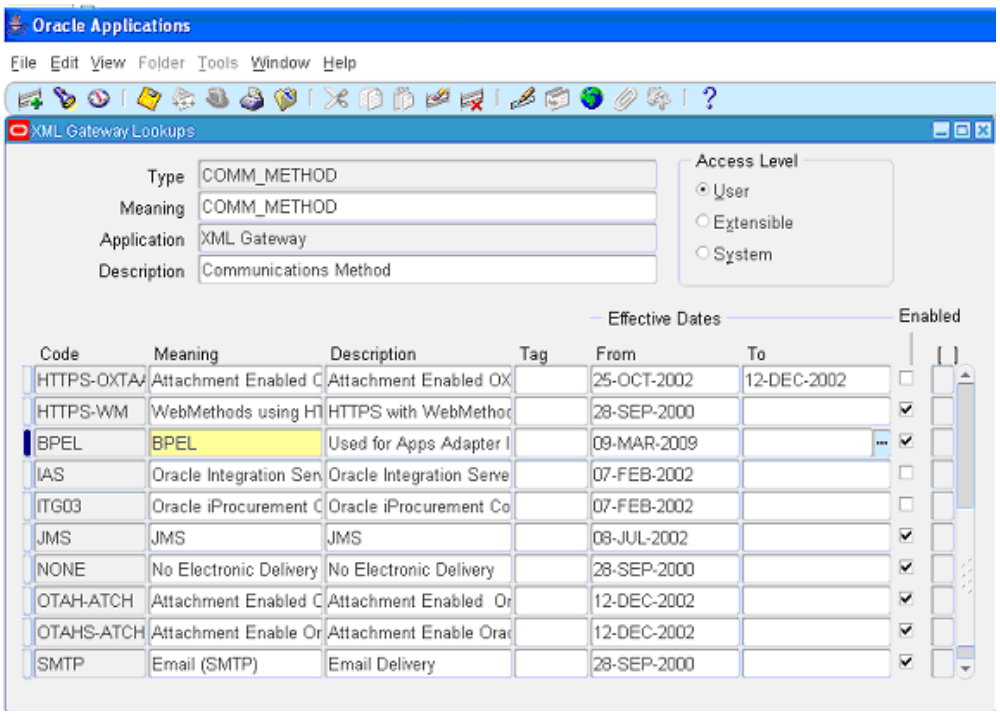
To set up the correlation identifier:

1. Log in to Oracle Applications with the XML Gateway responsibility. The Navigator page appears.

2. Click the **XML Gateway** link.

3. Click the **Define Lookup Values** link under XML Gateway to open the XML Gateway Lookups form.

XML Gateway Lookups



4. Search for `COMM_METHOD` in the **Type** field to see if it exists in the system.

5. Add a new record to the `COMM_METHOD` type by entering `BPEL` for the **Code** field and **Meaning** field. Enter description information and save the record.

   Oracle XML Gateway puts the correlation of BPEL when enqueueing the message

on the ECX_OUTBOUND queue.

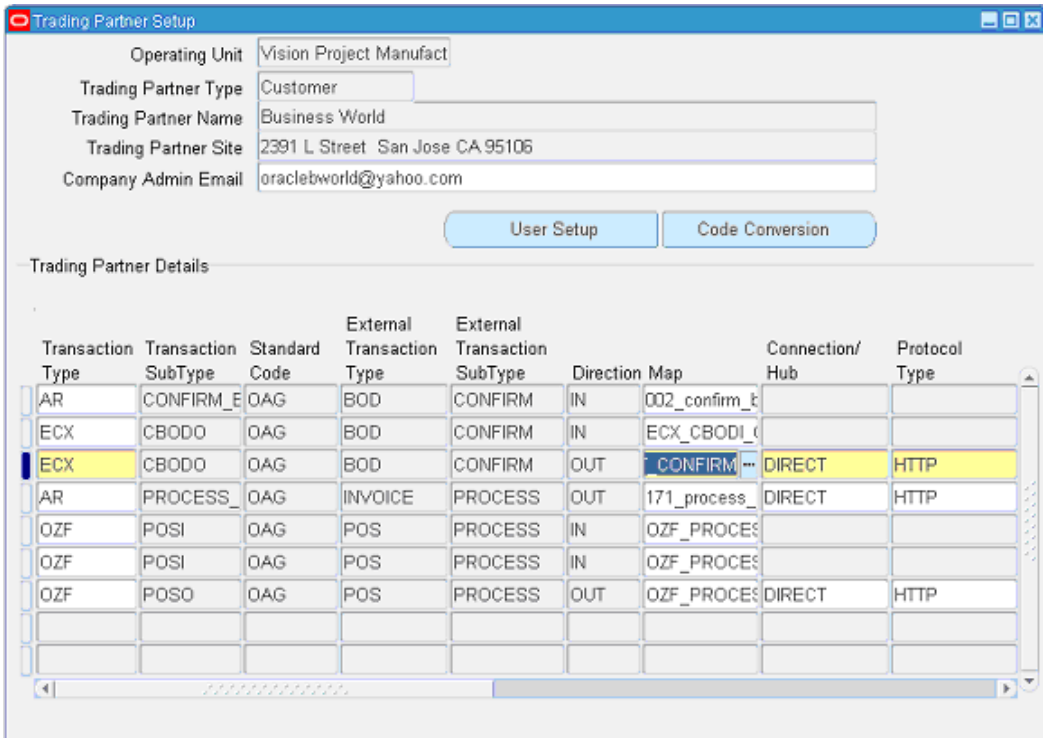
Setting Up XML Gateway Trading Partner

Once you have the correlation identifier set up correctly, you also need to ensure a valid outbound XML Gateway trading partner in the Trading Partner Setup form through XML Gateway responsibility. You want to have the Protocol Type field set to `BPEL`.

For example, a Trading Partner (such as 'Business World' with Site information '2391 L Street San Jose CA 95106' and partner type 'Customer') has the following details for an outbound transaction:

- Transaction type: ECX
- Transaction sub type: CBODO
- Standard Code: OAG
- External transaction type: BOD
- External transaction subtype: CONFIRM
- Direction: OUT
- Map: ECX_CBODO_OAG72_OUT_CONFIRM
- Connection/Hub: DIRECT
- Protocol Type: BPEL
- Source Trading partner location code: BWSANJOSE

Trading Partner Setup



BPEL Process Creation Flow

Based on the PO acknowledgement XML Transaction scenario, the following design-time tasks are discussed in this chapter:

1. Create a new BPEL project, page 4-47
2. Create a Partner Link, page 4-51
3. Add a Receive activity, page 4-56
4. Add a Partner Link for File Adapter, page 4-59
5. Add an Invoke activity, page 4-63
6. Add an Assign activity, page 4-65

Creating a New BPEL Project

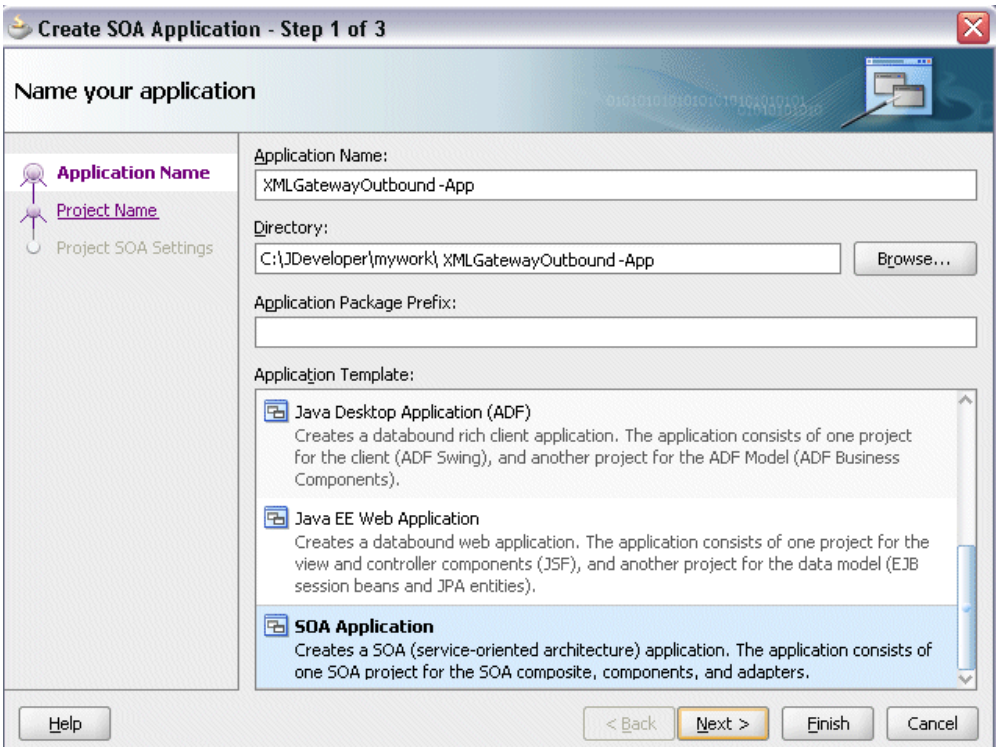
To create a new BPEL project:

1. Launch Oracle JDeveloper.

2. Click **New Application** in the Application Navigator.

   The Create SOA Application - Name your application page is displayed.

The Create SOA Application - Name your application Page

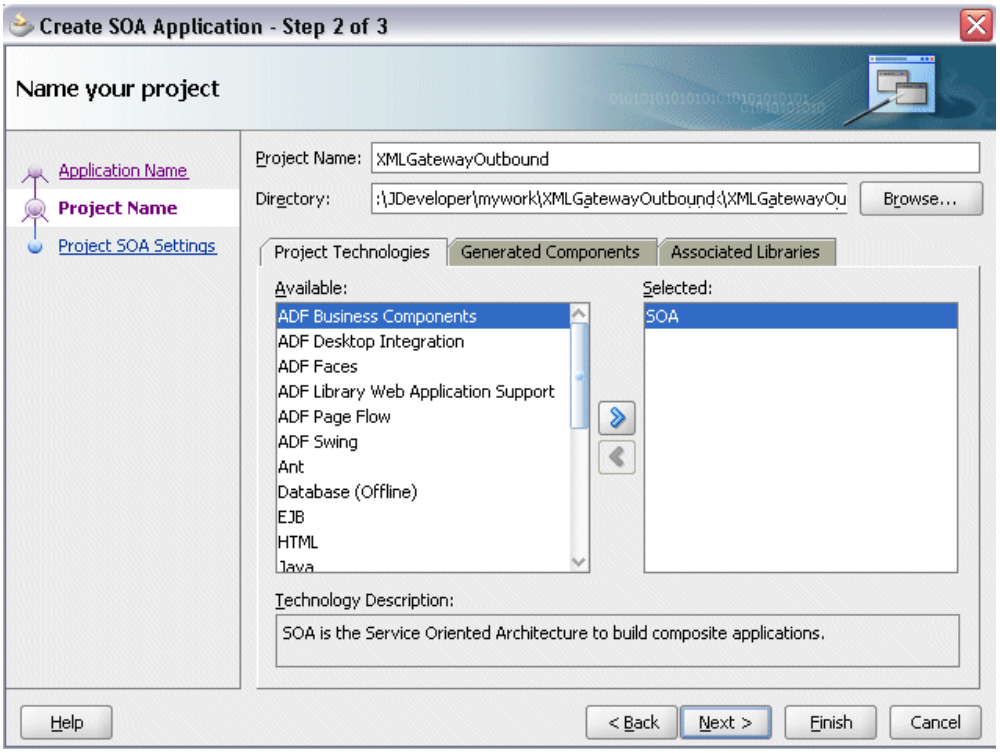


3. Enter an appropriate name for the application in the **Application Name** field and select **SOA Application** from the Application Template list.

   Click **Next**. The Create SOA Application - Name your project page is displayed.

4. Enter an appropriate name for the project in the **Project Name** field. For example, `XMLGatewayOutbound`.

The Create SOA Application - Name your project Page



5. In the Project Technologies tab, ensure that **SOA** is selected from the Available technology list to the Selected technology list.

   Click **Next**. The Create SOA Application - Configure SOA settings page is displayed.

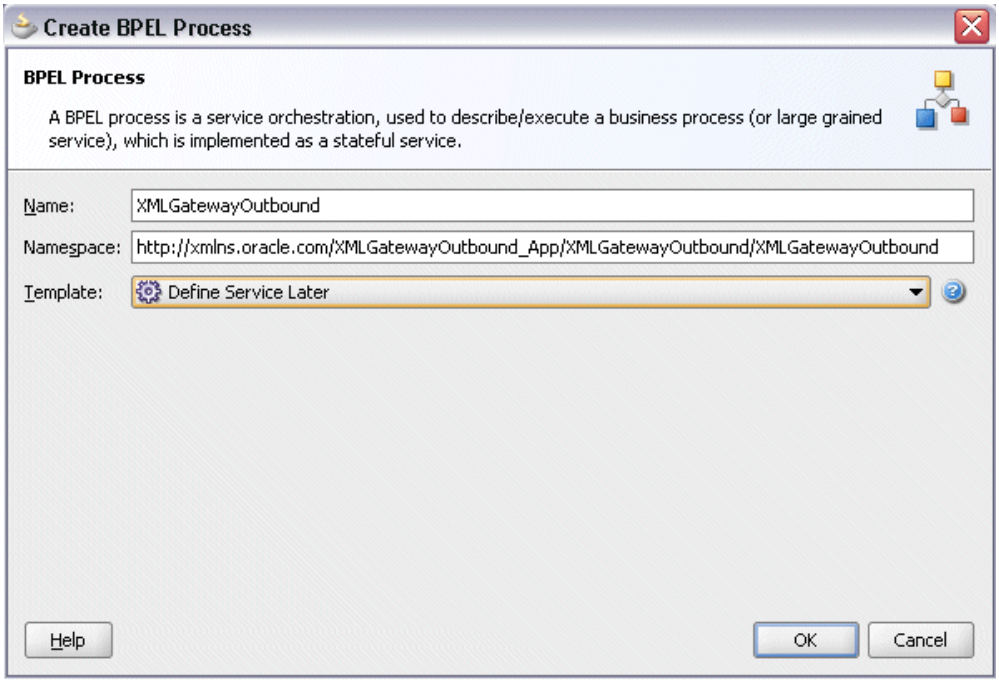
6. Select **Composite With BPEL** from the Composite Template list, and then click **Finish**. You have created a new application, and an SOA project. This automatically creates an SOA composite.

   The Create BPEL Process page is displayed.

7. Enter an appropriate name for the BEPL process in the **Name** field. For example, `XMLGatewayOutbound`.

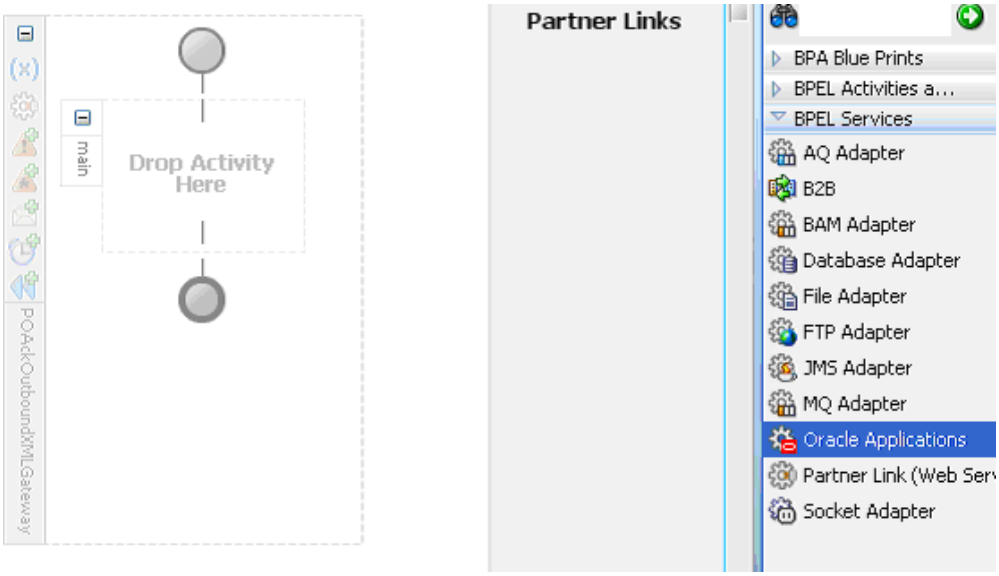
   Select **Define Service Later** in the **Template** field. Click **OK**.

Create BPEL Process



An empty BPEL process is created. The required source files including bpel and wsdl, using the name you specified (for example, `XMLGatewayOutbound.bpel` and `XMLGatewayOutbound.wsdl`) and `composite.xml` are also generated.

An Empty BPEL Process



Adding a Partner Link

This section describes how to create an Oracle Applications 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.

You need to add a partner link for the outbound XML message in order for the Receive activity to dequeue it later.

To add a partner link:

1. Click **BPEL Services** in the Component palette.

   Drag and drop **Oracle Applications** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears. Click **Next**.

2. Enter a service name in the **Service Name** field. For example, `GetAck`.

   Click **Next**. The Service Connection dialog appears.

3. You can perform either one of the following options for your database connection:

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

- You can create a new database connection by clicking the **Create a New Database Connection** icon.

  How to define a new database connection, see Create a New Database Connection, page 4-13.

- You can select an existing database connection that you have configured earlier from the **Connection** drop-down list.

  The Service Connection page will be displayed with the selected connection information. The JNDI (Java Naming and Directory Interface) name corresponding to the database connection appears automatically in the **Database Server JNDI Name** field. Alternatively, you can specify a JNDI name.

  > **Note:** When you specify a JNDI name, the deployment descriptor of the Oracle Applications adapter must associate this JNDI name with configuration properties required by the adapter to access the database.

  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.

  > **Note:** For more information about JNDI concepts, refer to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

4. Once you have completed creating a new connection for the service, you can add an outbound message map by browsing through the list of APIs available in Oracle Applications.

   Click **Next**.

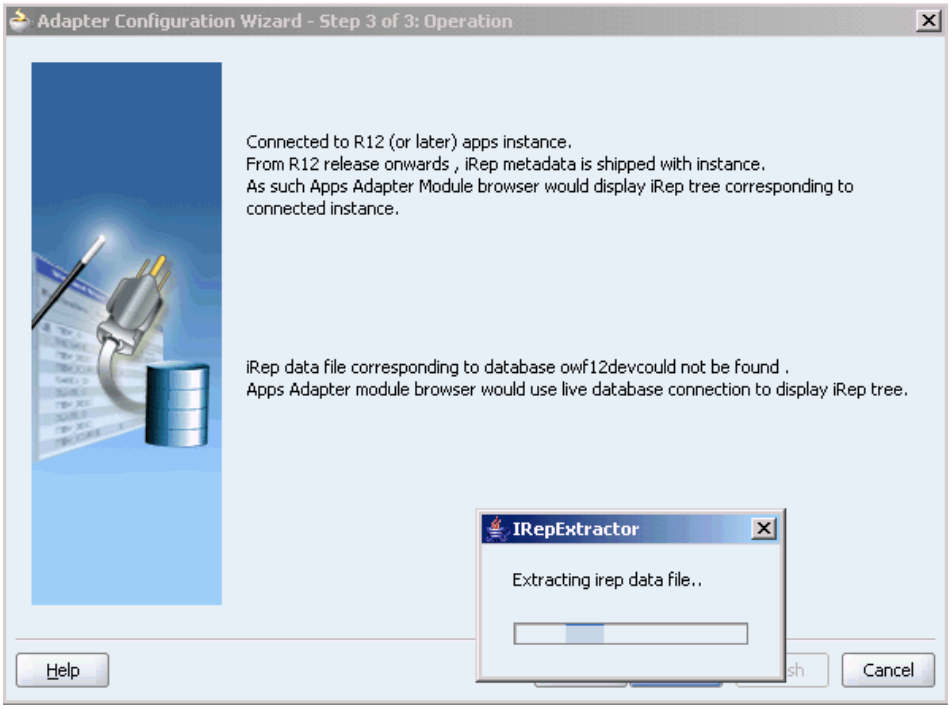
   For Oracle E-Business Suite Release 12:

   If you are connecting to Oracle E-Business Suite Release 12, then the **IREP File not present** dialog appears indicating that Adapter could not find the Oracle Integration Repository data file corresponding to the database you are connecting in your workspace. Absence of the data file would make browsing or searching of Integration Repository tree considerably slow. You can choose to extract the data file and create a local copy of the Integration Repository data file. Once it is created successfully, Adapter will pick it up automatically next time and retrieve data from your local Integration Repository.

   You can select one of the following options:

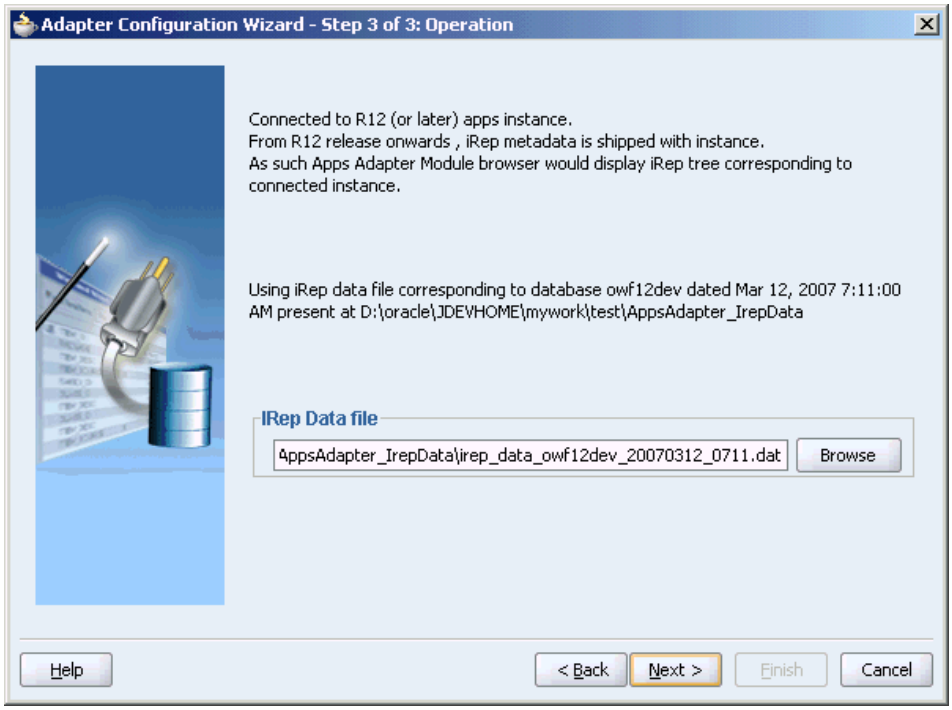
   - Click **Yes** to extract the Integration Repository data file.

Extracting Integration Repository Data File



After the system successfully creates a local copy of the Integration Repository data file, next time when you connect to the database, you will find the **IRep Data File** field appears in the Operation dialog indicating where your local copy exists with the creation date and time as part of the file name.

Using the Local Integration Repository Data File



- Click **No** to query the Integration Repository data file from the live database you are connecting to display the Integration Repository tree.

  Note: It is highly recommended that you create a local copy of the Integration Repository data file so that Adapter will query the data next time from the local copy in your workspace to enhance the performance.

  Click **Next** in the Operation page to open the Oracle Applications Module Browser.

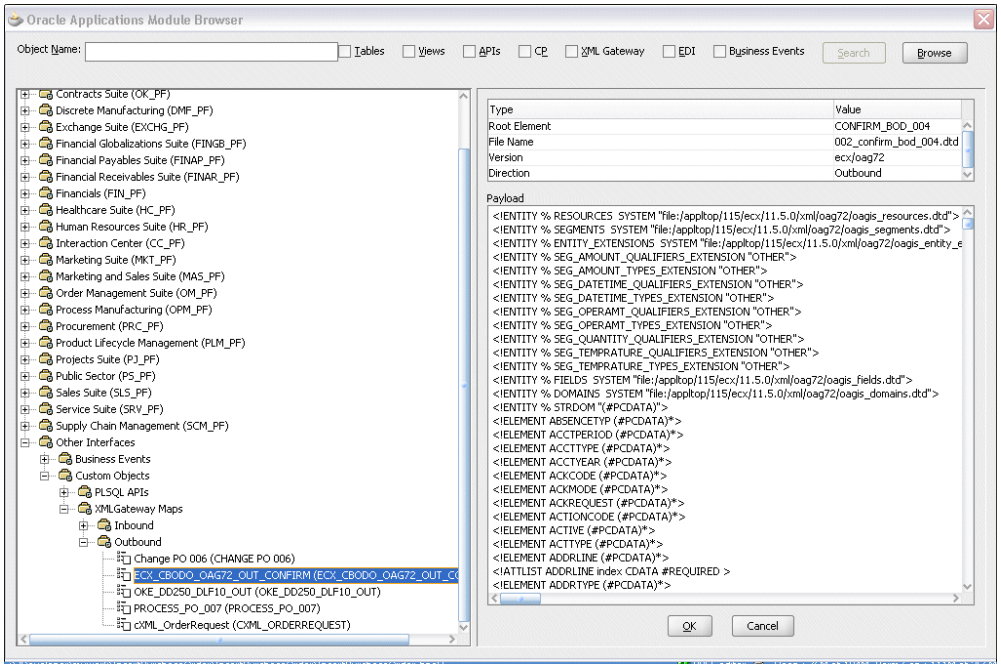
For Oracle E-Business Suite pre-Release 11.5.10:

If you are connecting to a pre-11.5.10 Oracle Applications instance, you must select the interface type in the Adapter Configuration Wizard. Select **XML Gateway** to proceed.

Click **Get Object** in the Application Interface dialog to open the Oracle Applications Module Browser.

5. The Oracle Applications Module Browser combines interface data from Oracle Integration Repository with information about the additional interfaces supported by Oracle Application Adapter, organized in a tree hierarchy.

Specify an API from The Oracle Applications Module Browser

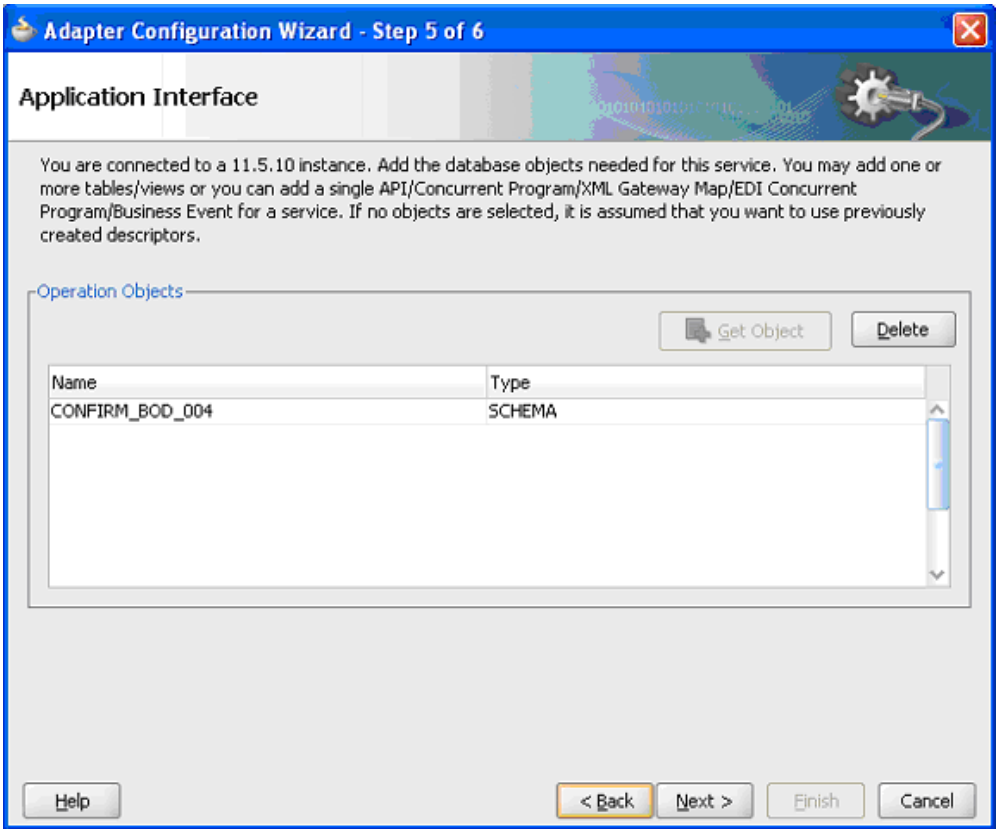


> **Note:** The Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. Each product family contains the individual products. Each product contains the business entities associated with the product. Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide.

Navigate to *Other Interfaces* > *Custom Objects* >*XML Gateway Maps*> *Outbound* to select an outbound map `ECX_CBODO_OAG72_OUT_CONFIRM`.

6. Click **OK**.

Adapter Configuration Wizard - Application Interface Page



7. Click **Next**, then click **Finish** to complete the process of configuring 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.

8. Click **Apply** and then **OK**. The partner link is created with the required WSDL settings.

Adding a Receive Activity

When configuring the 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.

To configure the Receive activity:

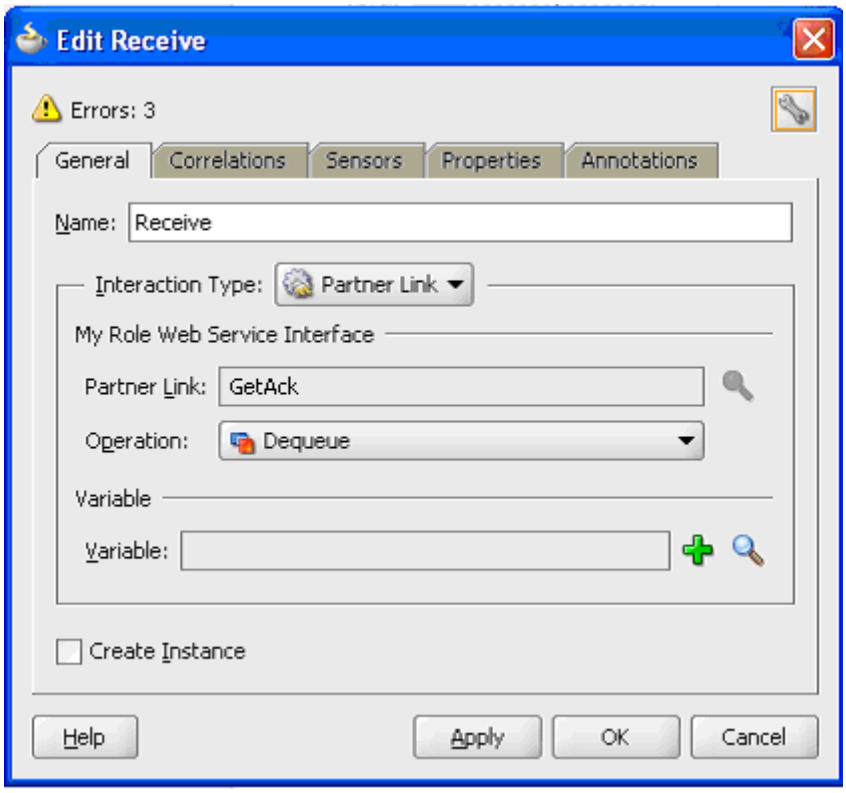
1. In JDeveloper BPEL Designer, click **BPEL Services and Components** in the Component palette.

Drag and drop **Receive** from the BPEL activity list into the center swim lane of the process diagram.

2. Link the **Receive** activity to the partner link `GetAck` you created earlier.

   The Receive dialog appears.

Configuring the Receive Activity

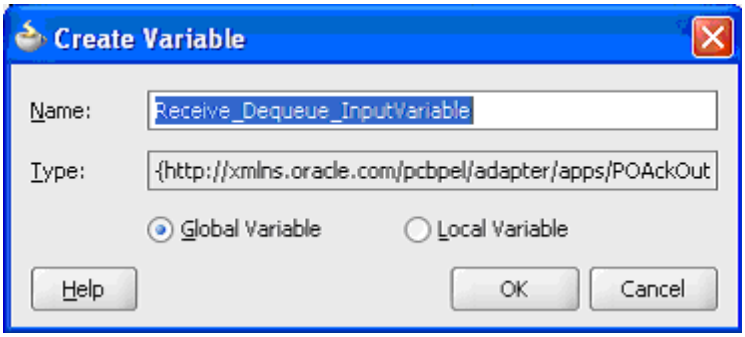


3. Enter an appropriate name for the Receive activity.

   The Dequeue **Operation** is automatically selected since the partner link has been configured with an outbound XML Gateway map.

4. Specify a **Variable** to receive the message data from the partner link by clicking the **Create** icon to the right of the Variable field. The **Create Variable** dialog box appears.

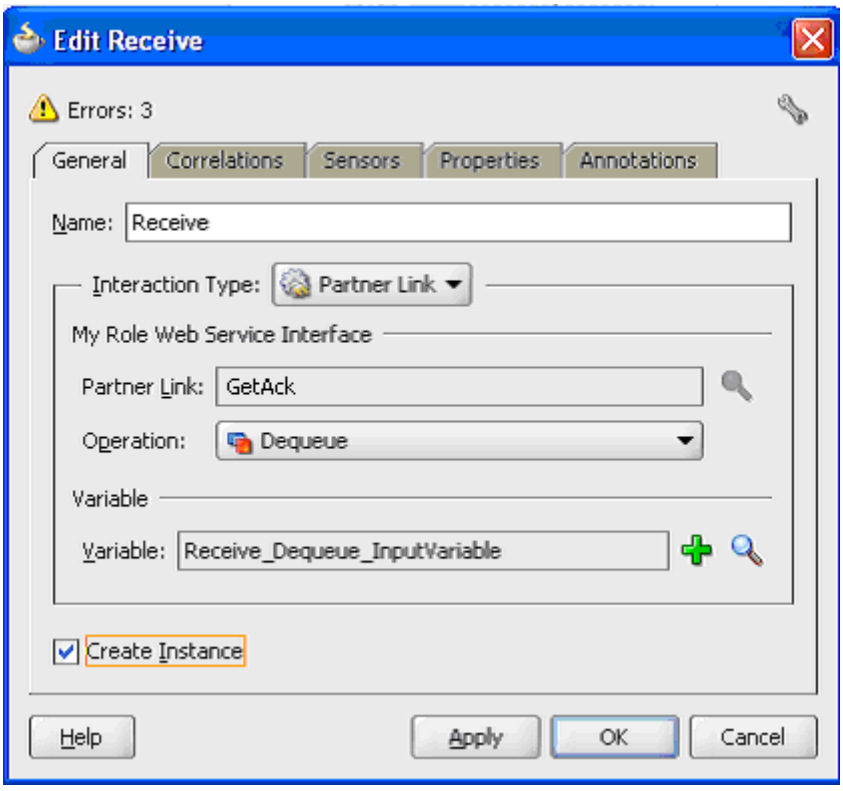
Creating a Variable to Receive Message Data



5. Click **OK** to accept the default name.

6. Select the **Create Instance** check box.

Configuring the Receive Activity



7. Click **Apply** in the Receive dialog, then click **OK**.

Adding a Partner Link for File Adapter

Use this step to write PO acknowledgement details in an XML file as an output file.

To add a Partner Link for File Adapter:

1. In JDeveloper BPEL Designer, click **BPEL Services** in the Component palette.

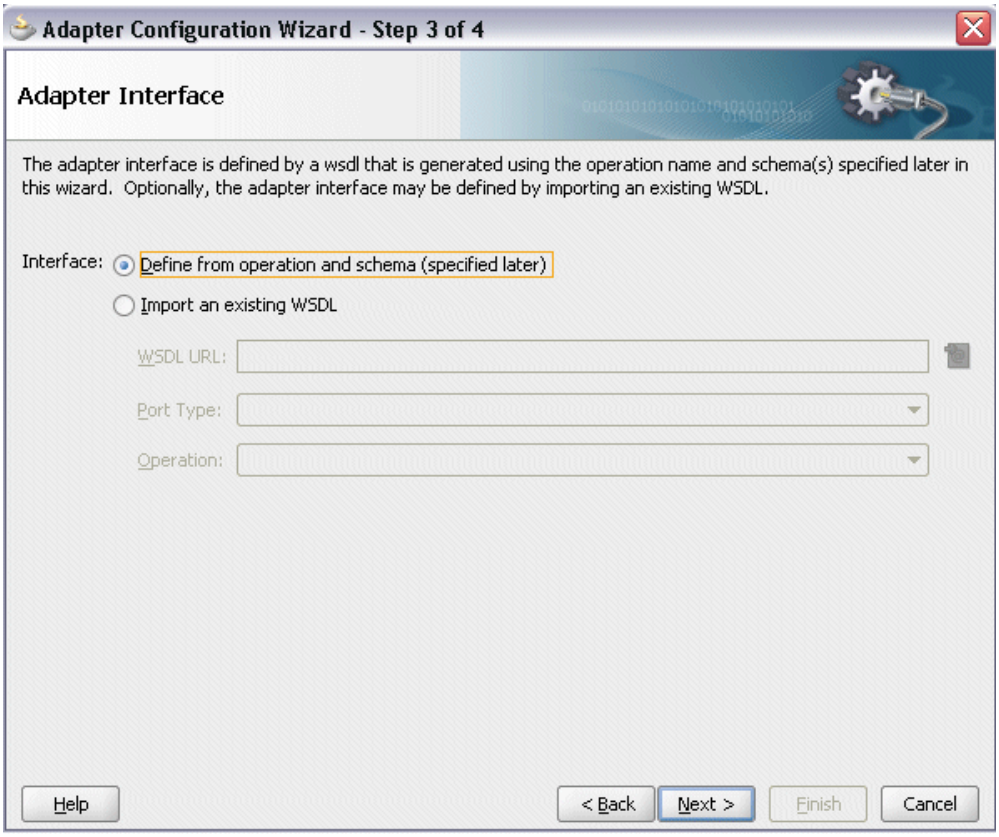
   Drag and drop **File Adapter** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears.

   Click **Next**.

2. In the Service Name dialog, enter a name for the file adapter service, for example, `WriteAck`.

3. Click **Next**. The Adapter Interface page appears.

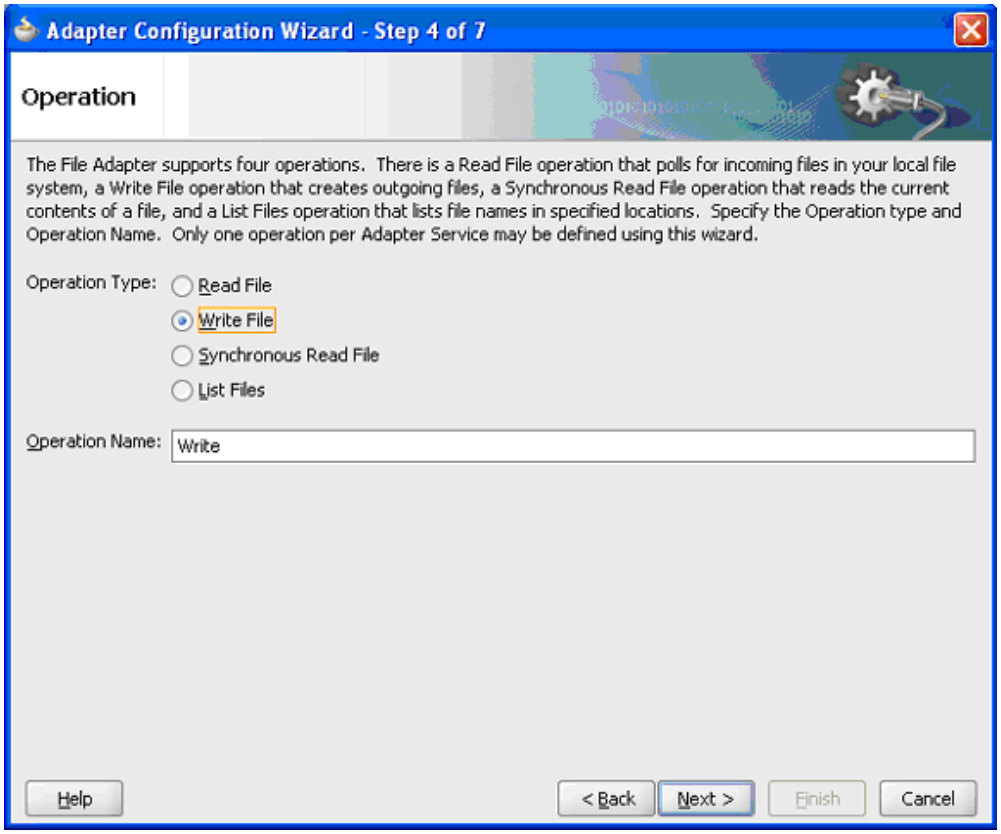
Specifying the Adapter Interface



Select the **Define from operation and schema (specified later)** radio button and click **Next**.

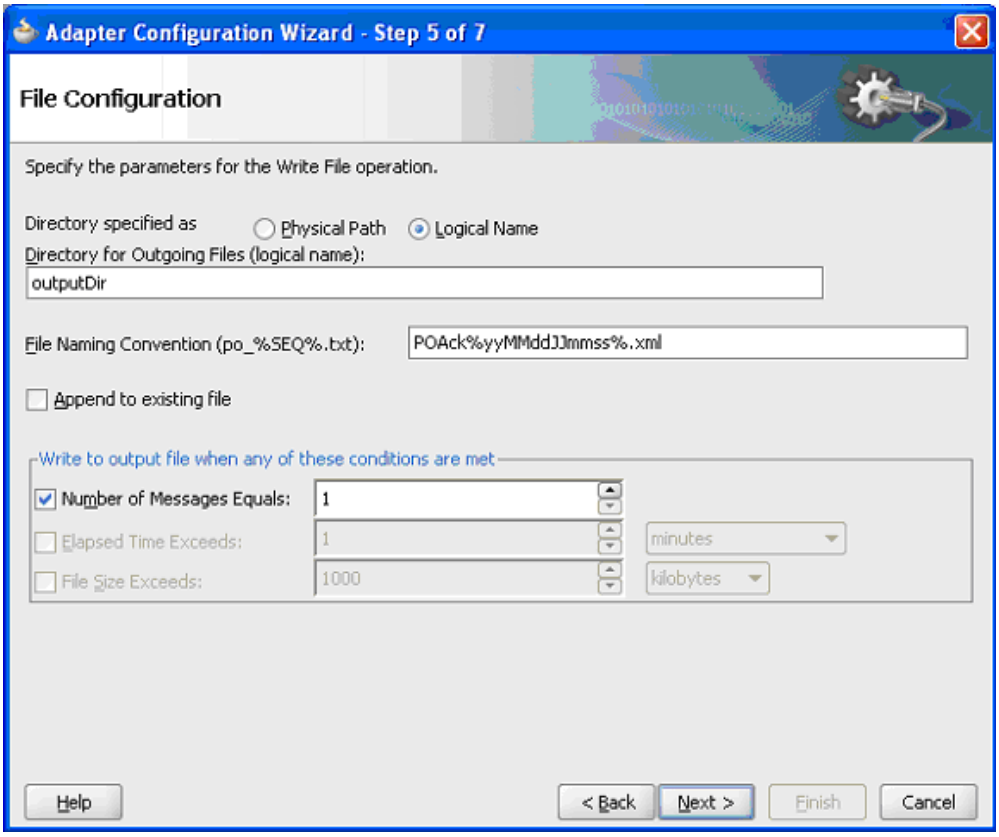
4. In the Operation page, specify the operation type. For example, select the **Write File** radio button. This automatically populates the **Operation Name** field.

Specifying the Operation



Click **Next** to access the File Configuration page.

Adapter Configuration Wizard - File Configuration Page



5. For the Directory specified as field, select **Logical Name**. Enter directory name in the Directory for Outgoing Files (logical name) field, for example, `outputDir`.

   Specify a naming convention for the output file, for example, `POAck%yyMMddJJmmss%.xml`.

   Select the **Number of Messages Equals** check box and set it to '1'.

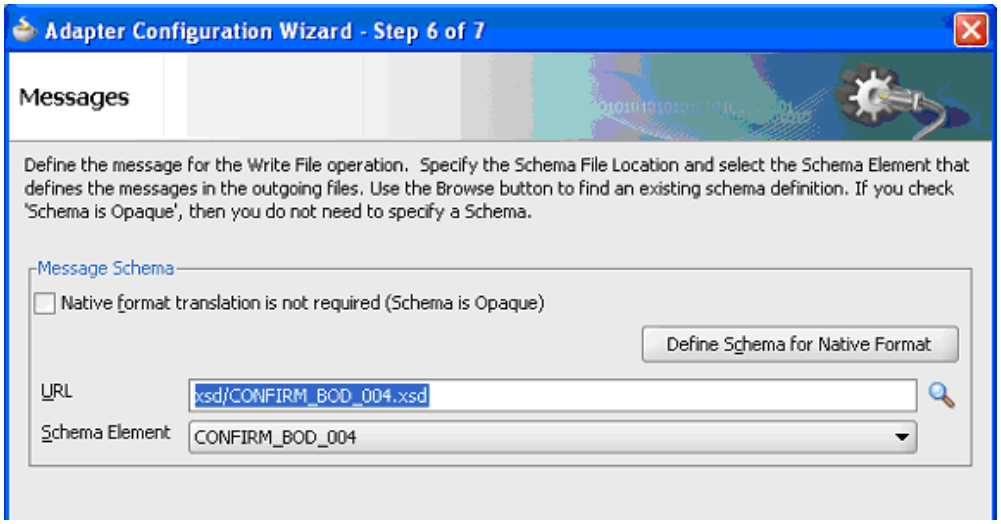
   Click **Next** to open the Messages page.

6. Select the 'browse for file schema' icon next to the URL field to locate the schema location and schema element.

   The Type Chooser dialog box appears. Expand the node by clicking **Project Schema Files > CONFIRM_BOD_004.xsd**. Select **CONFIRM_BOD_004** and click **OK**.

   The selected schema information will be automatically populated in the URL and Schema Element fields.

Adapter Configuration Wizard - Messages Page



7. Click **Next** and then **Finish**. The wizard generates the WSDL file corresponding to the partner link. The main Create Partner Link dialog box appears, specifying the new WSDL file `WriteAck.wsdl`.

   Click **Apply** and **OK** to complete the configuration and create the partner link with the required WSDL settings for the File Adapter service.

   The `WriteAck` Partner Link appears in the BPEL process diagram.

Adding an Invoke Activity

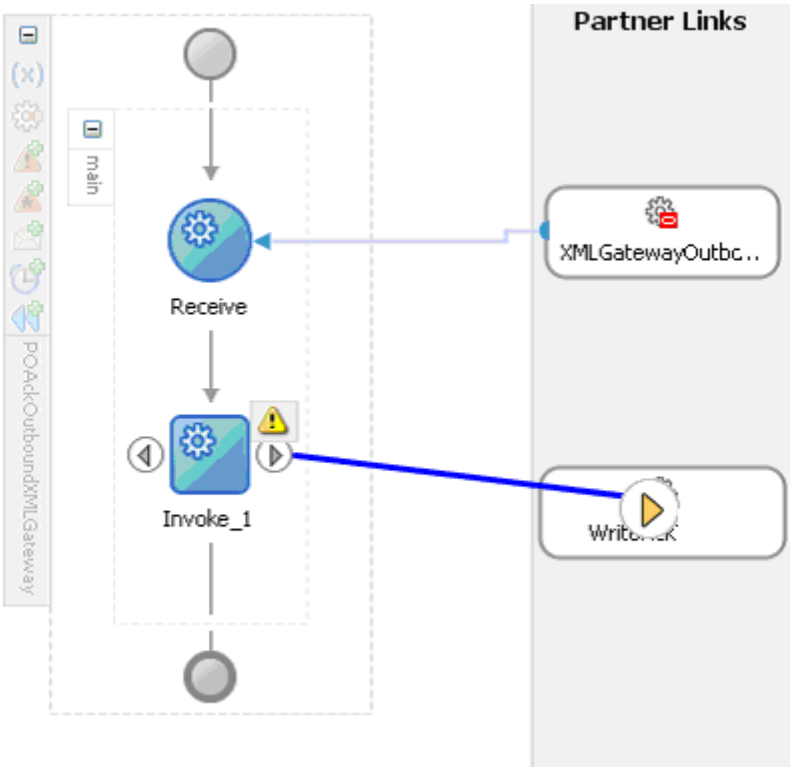
This step is to configure an Invoke activity to write PO acknowledgement information to an XML file through invoking the partner link for File Adapter.

To add an Invoke activity:

1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the **Invoke** activity into the center swim lane of the process diagram after the **Receive** activity.

2. Link the Invoke activity to the `WriteAck` service.

Adding an Invoke Activity

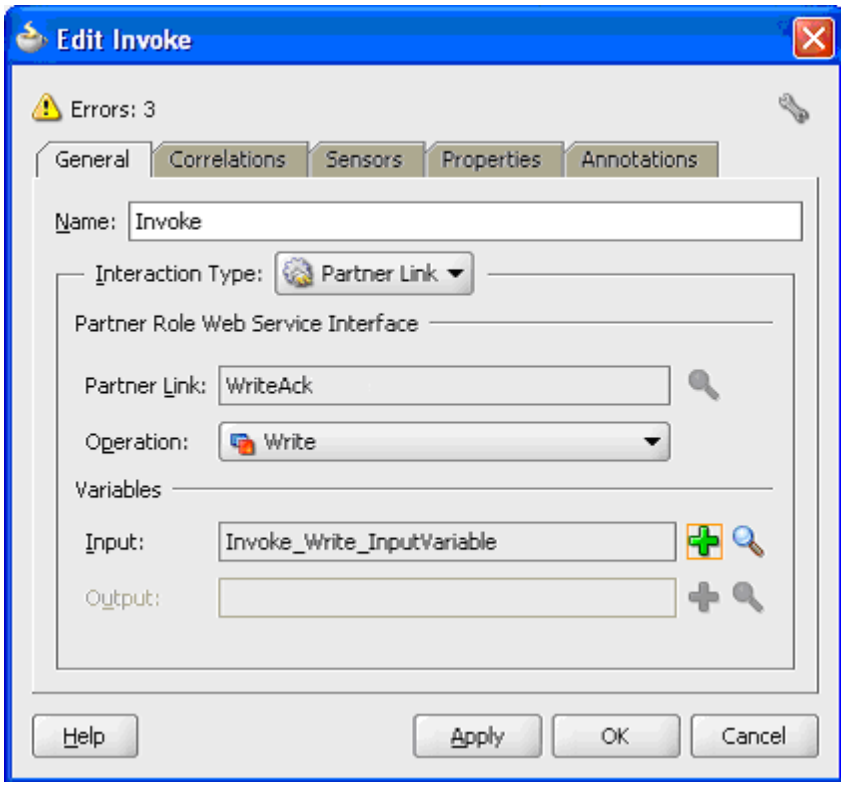


The Edit Invoke dialog appears.

3. Enter a name for the Invoke activity, then click the **Create** icon next to the **Input Variable** field to create a new variable. The Create Variable dialog box appears.

4. Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK**.

   Click **Apply** and then click **OK** in the Edit Invoke dialog to finish configuring the Invoke activity.



The Invoke activity appears in the process diagram.

Adding an Assign Activity

Use the Assign activity to take the output from the Receive activity and to provide input to the Invoke activity.

To add an Assign activity:

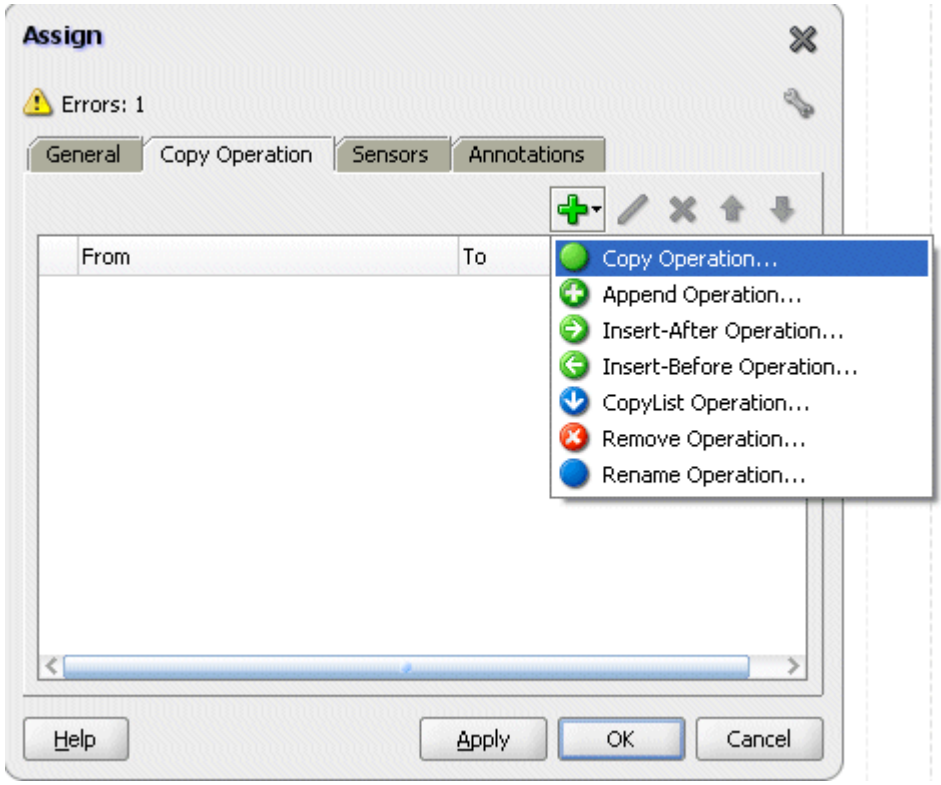
1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the **Assign** activity into the center swim lane of the process diagram, between the **Receive** and **Invoke** activities that you just created earlier.

2. Double-click the **Assign** activity to access the Edit Assign dialog.

   Click the General tab to enter a name for the Assign activity.

3. Select the Copy Operation tab, click the 'Plus' sign icon and select **Copy Operation** from the menu. The Create Copy Operation window appears.

Specifying a Copy Operation Action



4. Enter the assign parameters:

   - In the From navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Receive_DEQUEUE_InputVariable > CONFIRM_BOD_004** and select **ns3:CONFIRM_BOD_004**.

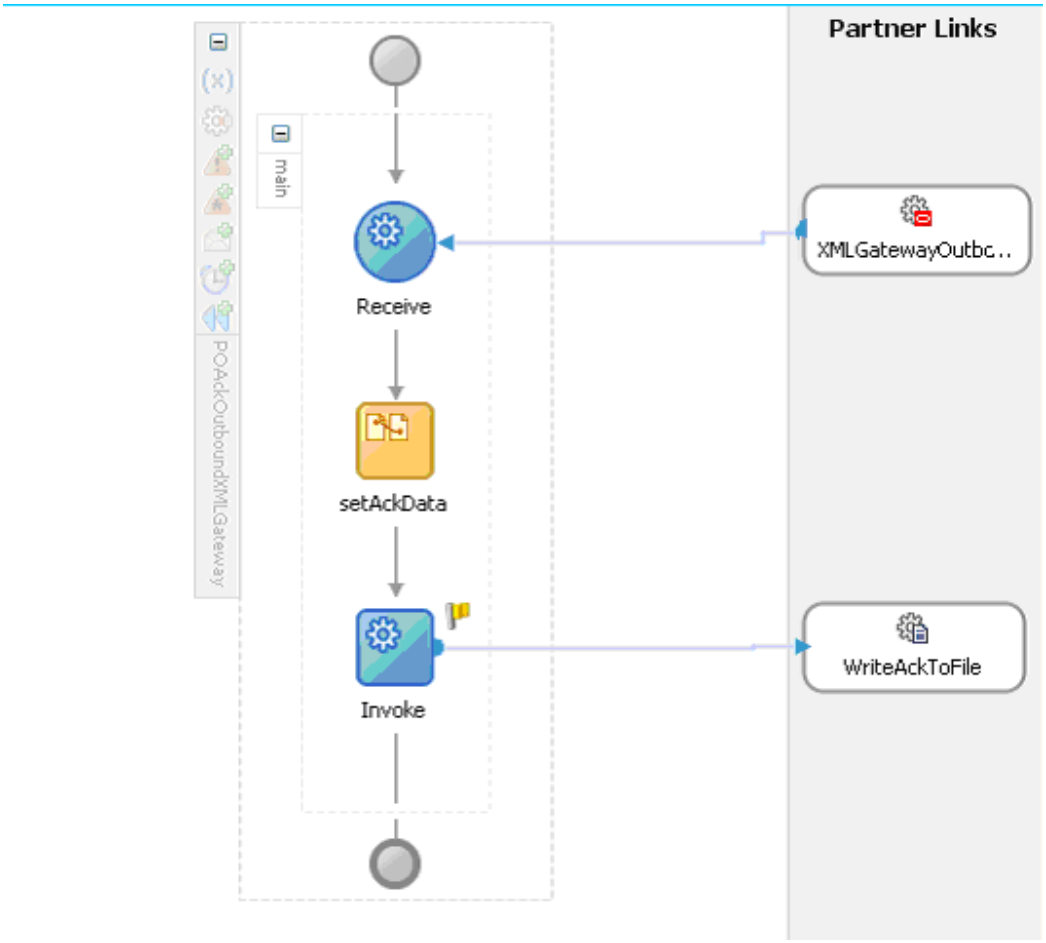
     The XPath field should contain your selected entry.

   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_Write_InputVariable > body** and select **ns3:CONFIRM_BOD_004**. The XPath field should contain your selected entry.

   - Click **OK**. The Edit Assign dialog box appears.

5. Click **Apply** and then click **OK** to complete the configuration of the Assign activity.

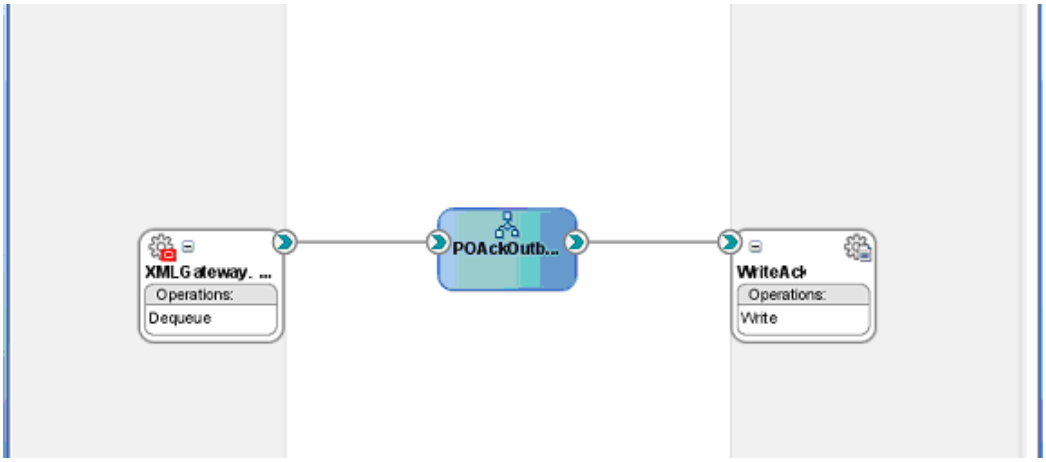
The following diagram illustrates the complete BPEL process diagram:



Click the `composite.xml` to display the Oracle JDeveloper composite diagram:

> **Note:** Click the Source tab of `composite.xml` to enter a value for the physical directory `outputDir` for the reference WriteEventData (such as `/usr/tmp`).
>
> ```
> <property name="outputDir" type="xs:string"
> many="false" override="may">/usr/tmp</property>
> ```

Oracle JDeveloper composite Diagram



Run-Time Task for XML Gateway Outbound Messaging

After designing the BPEL process, you can compile, deploy and test it.

1. Deploy the BPEL Process., page 4-68
2. Test the BPEL process., page 4-71

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 application server (Oracle WebLogic Server) that you have established the connection.

Prerequisites

Before deploying the BPEL process using Oracle JDeveloper, you must ensure the following:

- You must have established the connectivity between the deign-time environment and an application server.

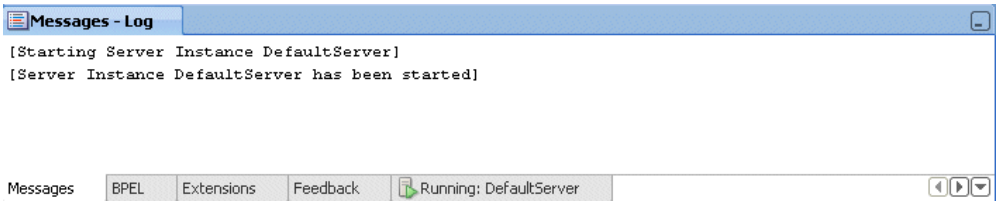
  For more information, see Configuring the Data Source in Oracle WebLogic Server, page A-3 and Creating an Application Server Connection, page A-8.

- Oracle WebLogic Server has been started.

  Before deploying the BPEL process, you need to start the Oracle WebLogic Server that you have established the connection.

  If a local instance of the WebLogic Server is used, start the WebLogic Server by selecting **Run > Start Server Instance** from Oracle JDeveloper.

Once the WebLogic Admin Server "DefaultServer" instance is successfully started, you should find that <Server started in Running mode> and DefaultServer started message in the Running:DefaultServer and Messages logs.

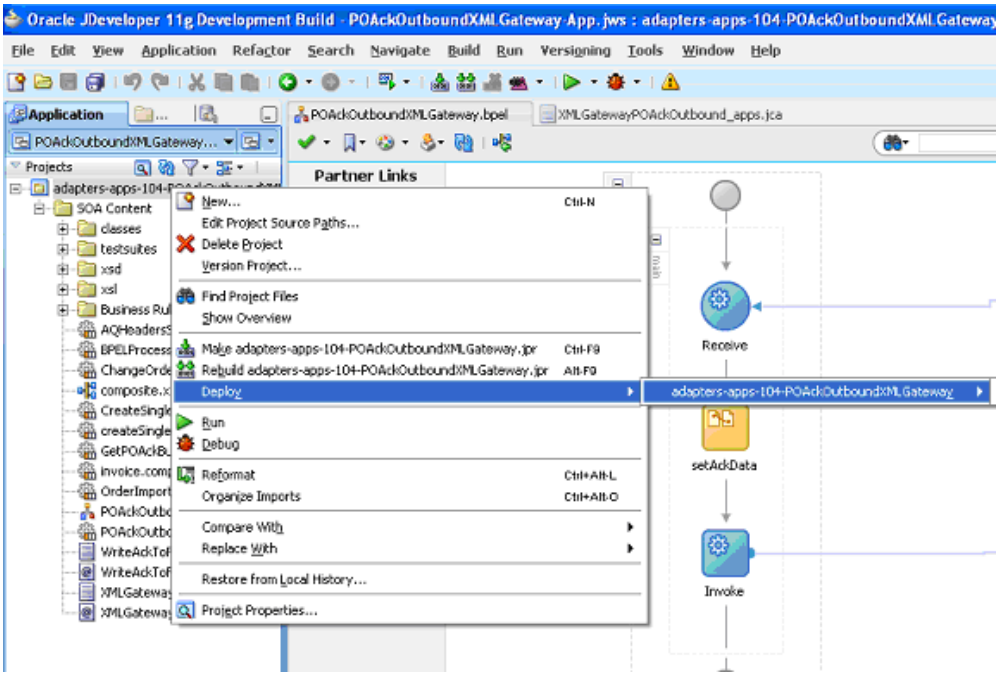


To deploy the BPEL process:

1. Select the BPEL project in the Applications Navigator.

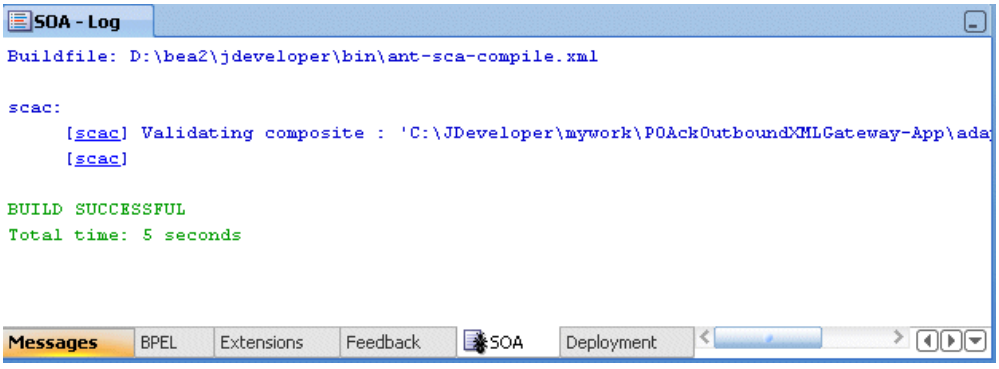
2. Right-click the project name. Select **Deploy > [project name] > [serverConnection]** from the menu that appears.

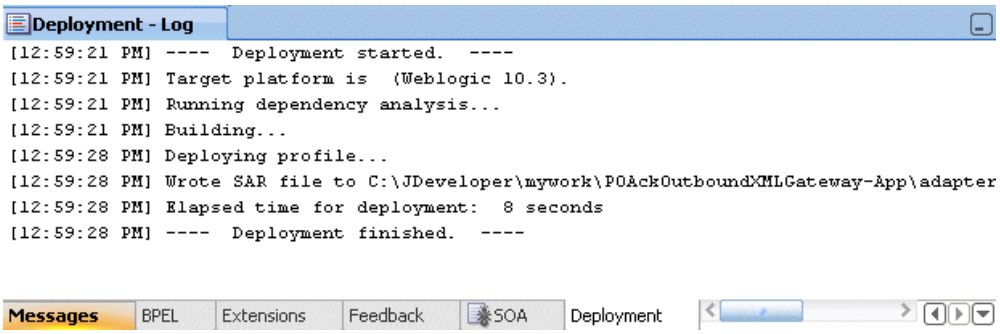
Deploying the BPEL Process



The BPEL project is compiled and successfully deployed. You can check the progress in the Messages window.

Compilation and Deployment Message Logs





Testing the BPEL Process

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.

> **Note:** For details on using the Transaction Monitor, see *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

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

To validate the design-time tasks created earlier, you can log in to Oracle Applications to manually create and book the order as well as generate the order acknowledgement by submitting a Workflow Background Process concurrent request.

To manually test the BPEL process:

1. Log in to Oracle Applications with the XML Gateway responsibility.

   This is to ensure that the XML Gateway trading partner is set up correctly so that a purchase order can have a valid customer that has been defined.

2. Select Define Trading Partner from the navigation menu to access the Trading Partner Setup window.

3. Enter the header values on the Trading Partner Setup form as follows:

   - Trading Partner Type: Customer
   - Trading Partner Name: For example, Business World

- Trading Partner Site: Enter a trading partner site information. For example, 2391 L Street, San Jose, CA 95106
- Company Admin Email: Enter a valid e-mail address.

4. Enter the following trading partner details:
   - Transaction Type: ECX
   - Transaction SubType: CBODO
   - Standard Code: OAG
   - External Transaction Type: BOD
   - External Transaction SubType: CONFIRM
   - Direction: Out
   - Map: ECX_CBODO_OAG72_OUT_CONFIRM
   - Connection / Hub: DIRECT
   - Protocol Type: BPEL
   - Username: 'operation'
   - Password: enter 'welcome' twice
   - Protocol Address: 'http://ebssoa.sample.com'
   - Source Trading Partner Location Code: BWSANJOSE
5. Save your work.

To successfully generated PO Acknowledgement, you need to create an order and then manually book the order through Order Management.

Use the following steps to create an order and then manually book the order:

1. Switch to Order Management Super User, Vision Operations (USA) responsibility and select Customer > Standard from the navigation menu to open the Enter Customer form.
2. Search on the 'Business World' in the Name field and click **Find**.
3. Select the Business World with the following information from the search results.
   - Account Name: Business World

- Registry ID: 2813
- Identifying check box: checked
- Address: 2391 L Street, San Jose, CA 95106
- Country: United States of America

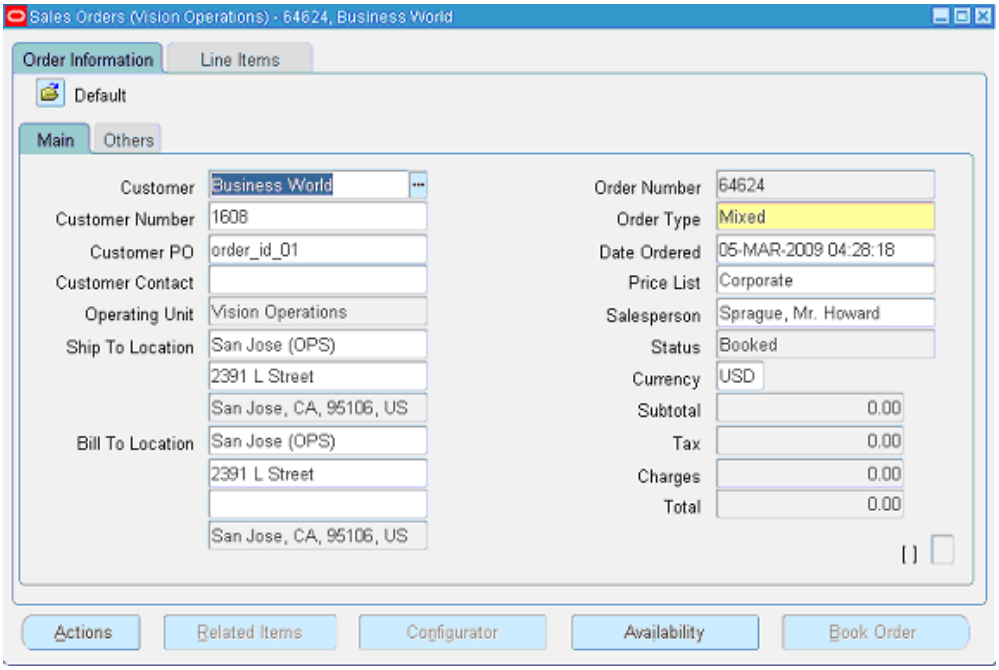
4. Select row with the following entries:
   - Account Number:1608
   - Account Description: Business World
   - Status: Active
5. Click **Details** to open the Customer Information page.
6. Click **Details** in the row with the Business World with Address '2391 L Street, San Jose, CA 95106' and Country 'United States of America'. This opens the Customer Account page.
7. Enter 'BWSANJOSE' in the EDI Location field.
8. In the Business Purposes tab, create a new row with the following values:
   - Usage: Sold To
   - Check on 'Primary' Check box

   Save your work.

Use the following steps to generate acknowledgement for already created order:

1. Log in to Oracle Applications with the Order Management Super User, Vision Operations (USA) responsibility. Select Order Returns > Sales Order to open the Sales Orders form.
2. Retrieve the order that you have created earlier by entering the order ID in the Customer PO field. For example, enter `order_id_01`.
3. Click **Book Order** to book the order.

Booking an Order

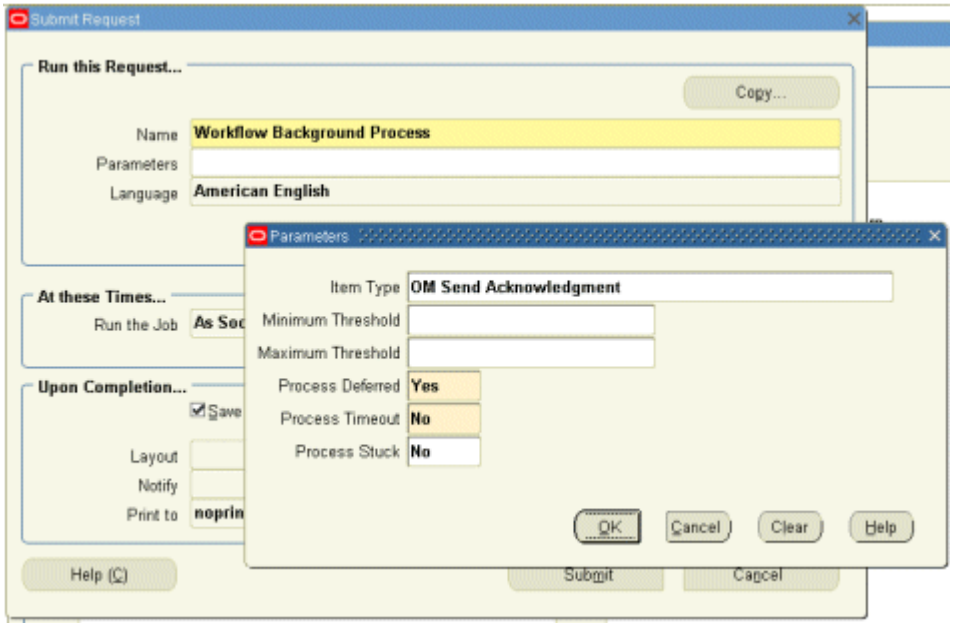


4. Switch to the System Administrator responsibility and select Request > Run.

5. Select **Single Request** and click **OK**.

6. Enter the following information in the Submit Request form:

Specifying Parameters



- Name: Workflow Background Process
- Enter the following parameters:
  - Item Type: OM Send Acknowledgement
  - Process Deferred: Y
  - Process Timeout: N
  - Process Stuck: N
- Click **OK**.

7. Click **Submit** to submit 'send acknowledgement' request.
8. View your request by entering the request ID to ensure its status is 'Success'.

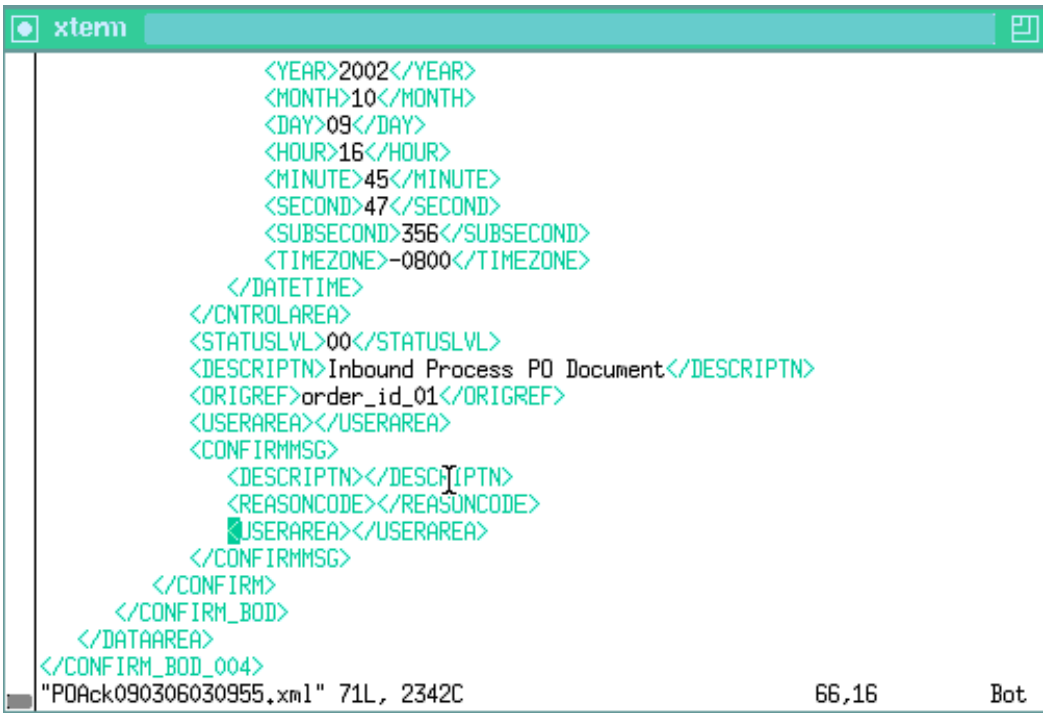
On approval of the order, Oracle E-Business Suite triggers the workflow that creates the Purchase Order Acknowledgement flow and sends out the PO Acknowledgement as an XML file. The workflow delivers the Confirm BOD as the PO Acknowledgement to the ECX_OUTBOUND queue for delivery to the other system.

On the other hand, Oracle BPEL PM listens to the ECX_OUTBOUND queue for the message with the correlation Id = "BPEL" which is the same id for this outbound message. The Confirm BOD as the PO Acknowledgement is written as XML file using

File Adapter.

Since the BPEL process is deployed, the process is continuously polling the ECX_OUTBOUND queue for PO acknowledgement. It also writes PO Acknowledgement in the physical directory mentioned in `composite.xml` after receiving from XML Gateway ECX_OUTBOUND queue.

You can open file in text editor and search for ORIGREF element. Its value should be same as order id (`order_id_01` whose order was booked.



Troubleshooting and Debugging

If you experience problems with your Oracle XML Gateway integration, you can take the following troubleshooting steps:

- Confirm that you have the correct settings for the following elements of the trading partner setup:
  - Standard Code
  - Transaction Type
  - Transaction Subtype
  - Source Trading Partner Location Code (Party Site ID)

- Confirm that the correct transaction is enabled for the trading partner.
- Check the status of the XML transaction in Transaction Monitor.
- Ensure that the document number is unique within this transaction type.
- For inbound transactions, confirm that ECX Listeners are running.
- For outbound transactions, confirm that Background Engine is running.
- In the trading partner setup, ensure that the Protocol Type is set to `BPEL`.
- Specify the same correlation ID for Oracle Applications as for the Adapter.

  The Adapter Configuration wizard of Adapter for Oracle Applications does not specify a correlation ID for XML Gateway transactions for inbound or outbound interfaces. Instead, a default correlation ID of `BPEL` is automatically set in the WSDL file. To make this configuration work, you must configure Oracle Applications to set the same correlation ID value of `BPEL` for the corresponding XML Gateway transactions.

  If you want the Adapter to use a different correlation ID than the default, you need to configure a correlation ID in Oracle Applications, then edit the `Correlation="BPEL"` line contained in the `<jca:operation>` section of the adapter service WSDL. Replace `BPEL` with the string value of the correlation ID you specified in Oracle Applications.

If you still experience problems with your integration, you can enable debugging.

Enabling Debugging

You can enable debugging for XML gateway using the BPEL Process Manager.

To enable debugging:

1. Log into your BPEL Process Manager domain.
2. Select *`yourdomain`*`.collaxa.cube.ws`
3. Select **Debug.**
4. Enable FND Logging to debug XML Gateway Transactions.

Debugging information is output to the log file for your domain. To examine the log file in the BPEL Process Manager, navigate to **Home > BPEL Domains >*yourdomain* > Logs.** The log file is ***yourdomain*.log**.

5

Using Business Events

This chapter covers the following topics:

- Overview of Business Events
- Design-Time Tasks for Outbound Business Events
- Creating a New BPEL Project
- Creating a Partner Link
- Configuring the Receive Activity
- Adding a Partner Link for the File Adapter
- Configuring the Invoke Activity
- Configuring the Assign Activity
- Run-Time Tasks for Outbound Business Events
- Deploying the BPEL Process
- Testing the BPEL Process
- Troubleshooting and Debugging

Overview of Business Events

The *Oracle Workflow Business Event System* (BES) is an application service that leverages the Oracle Advanced Queuing (AQ) infrastructure to communicate business events between systems. The Business Event System consists of the Event Manager and workflow process event activities.

The Event Manager contains a registry of business events, systems, named communication agents within those systems, and subscriptions indicating that an event is significant to a particular system. Events can be raised locally or received from an external system or the local system through AQ. When a local event occurs, the subscribing code is executed in the same transaction as the code that raised the event, unless the subscriptions are deferred.

Subscriptions can include the following types of processing:

- Executing custom code on the event information
- Sending event information to a workflow process
- Sending event information to other queues or systems

Business events are represented within workflow processes by event activities. By including event activities in a workflow process, you can model complex processing or routing logic for business events beyond the options of directly running a predefined function or sending the event to a predefined recipient.

Each business event represents a ready to use integration or extension point. Oracle E-Business Suite currently ships preconfigured with over 900 business events.

The uses of the Business Event System include:

- **System integration messaging hubs** - Oracle Workflow with the Business Event System can serve as a messaging hub for complex system integration scenarios. The Event Manager can be used to "hard–wire" routing between systems based on event and originator. Workflow process event activities can be used to model more advanced routing, content–based routing, transformations, error handling, and so on.
- **Distributed applications messaging** - Applications can supply Generate and Receive event message handlers for their business entities. For example, message handlers can be used to implement Master/Copy replication for distributed applications.
- **Message-based system integration** - You can set up subscriptions, which cause messages to be sent from one system to another when business events occur. In this way, you can use the Event Manager to implement point–to–point messaging integration.
- **Business–event based workflow processes** - You can develop sophisticated workflow processes that include advanced routing or processing based on the content of business events.
- **Non-invasive customization of packaged applications** - Analysts can register interesting business events for their Internet or intranet applications. Users of those applications can register subscriptions to those events to trigger custom code or workflow processes.

Business Events Concepts

Event

A business event is an occurrence in an Internet or intranet application or program that might be significant to other objects in a system or to external agents. For instance, the creation of a purchase order is an example of a business event in a purchasing application.

Event Key

A string that uniquely identifies an instance of an event. Together, the event name, event key, and event data fully communicate what occurred in the event.

Event Message

A standard Workflow structure for communicating business events, defined by the datatype WF_EVENT_T. The event message contains the event data as well as several header properties, including the event name, event key, addressing attributes, and error information.

Event Activity

A business event modeled as an activity so that it can be included in a workflow process.

Event Data

A set of additional details describing an event. The event data can be structured as an XML document. Together, the event name, event key, and event data fully communicate what occurred in the event.

Event Subscription

A registration indicating that a particular event is significant to a system and specifying the processing to perform when the triggering event occurs. Subscription processing can include calling custom code, sending the event message to a workflow process, or sending the event message to an agent.

Deferred Subscription Processing

If you do not want subscriptions for an event to be executed immediately when the event occurs, you can defer the subscriptions. In this way you can return control more quickly to the calling application and let the Event Manager execute any costly subscription processing at a later time.

Agent

An agent is a named point of communication within a system. Communication within and between systems is accomplished by sending a message from one agent to another. A single system can have several different agents representing different communication alternatives. For example, a system may have different agents to support inbound and outbound communication, communication by different protocols, different propagation frequencies, or other alternatives.

Design-Time Tasks for Outbound Business Events

Adapter for Oracle Applications is deployed at design-time using Oracle JDeveloper and at run-time using the BPEL Process Manager.

This section discusses the process of configuring Adapter for Oracle Applications to create business event outbound subscriptions. It describes the tasks required to configure Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper.

Multiple BPEL Processes Consuming the Same Business Event

Please note that Adapter for Oracle Applications can handle multiple BPEL processes consuming the same business event. Adapter for Oracle Applications creates only single subscription for a particular business event regardless of the number of BPEL process consuming it. Internally, this subscription forwards business event message to a multi-consumer AQ. Since each BPEL process is an unique consumer for the event, when the message is placed in the queue, all BPEL processes are notified. Therefore, as a user you do not need to create a separate subscription for each BPEL process. All you need to do is to create the service for the event, and Adapter for Oracle Applications will take care of message delivery to each BPEL process.

For example, if there are three BPEL processes (BPEL1, BPEL2, and BPEL3) that want to consume the same business event (such as BE1 event). For each BPEL process, you create a service for the BE1 event using Adapter for Oracle Applications. Adapter for Oracle Applications in turn creates a single subscription for all the three BPEL processes - BPEL1, BPEL2, and BPEL3. This subscription puts BE1 event message in multi-consumer AQ.

At run time, when a BE1 event is raised, since the subscription is applicable to all the three BPEL processes, all these three deployed BPEL processes will be activated and would receive the same BE1 event message.

BPEL Process Scenario

Take a PO XML Raise business event as an example.

When a purchase order is created and approved, a purchase order approved business event `oracle.apps.po.evnt.xmlpo` is raised. The subscription to this event is created in the background to listen to the business event and get event details. The event data will be passed through BPEL process activities and then written in XML file

as an output file.

If the BPEL process is successfully executed after deployment, you should get the same purchase order information from the output file once a purchase order is approved.

Prerequisites to Configure Outbound Business Events

- A valid XML Gateway trading partner must be set up first.
- The WF_Deferred Agent Listener must be up and running on the target instance.
- The event should be enabled for BPEL to subscribe to it. The event should not be in the disabled mode.

BPEL Process Creation Flow

Based on the PO XML Raise business event scenario, the following design-time tasks are discussed in this chapter:

1. Create a new BPEL project, page 5-5.
2. Create a partner link, page 5-10.
3. Configure a Receive activity, page 5-19.
4. Add a partner link for the file adapter, page 5-21.
5. Configure an Invoke activity, page 5-27.
6. Configure an Assign activity, page 5-29.

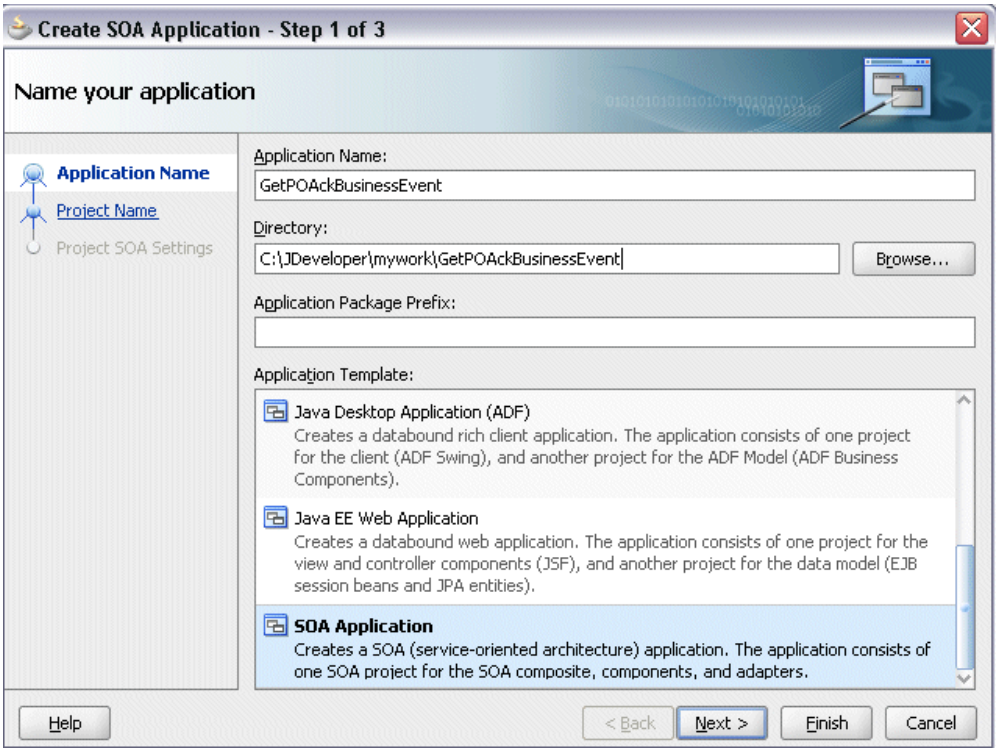
Creating a New BPEL Project

To create a new BPEL project:

1. Launch Oracle JDeveloper.
2. Click **New Application** in the Application Navigator.

   The Create SOA Application - Name your application page is displayed.

The Create SOA Application - Name your application Page

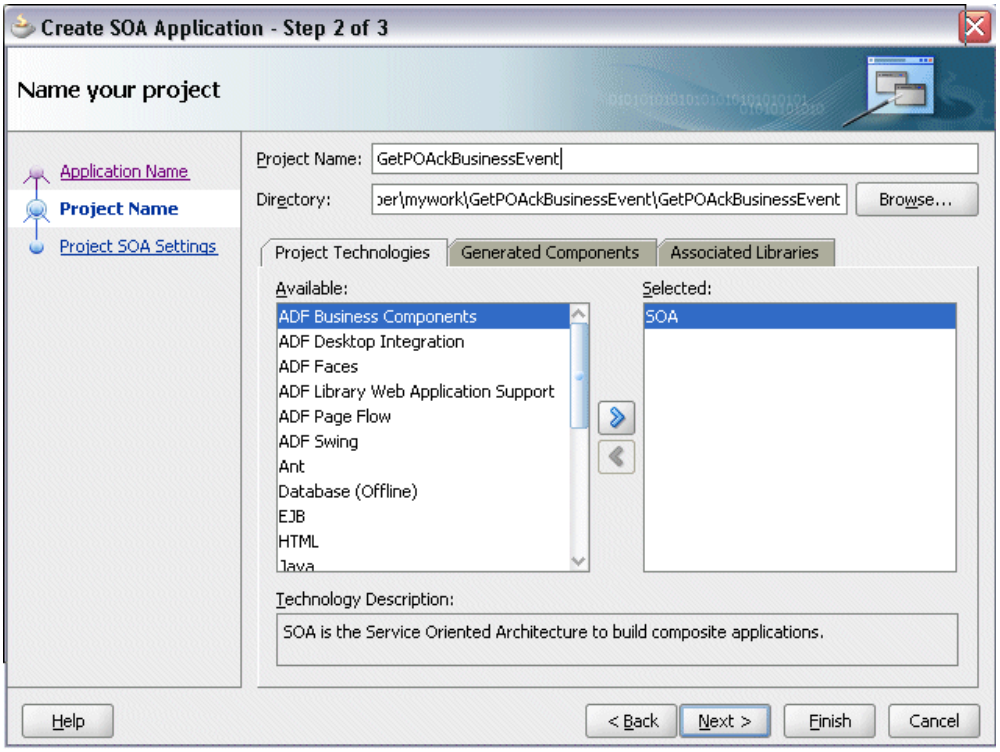


3. Enter an appropriate name for the application in the **Application Name** field and select **SOA Application** from the Application Template list.

   Click **Next**. The Create SOA Application - Name your project page is displayed.

4. Enter an appropriate name for the project in the **Project Name** field. For example, `GetPOAckBusinessEvent`.

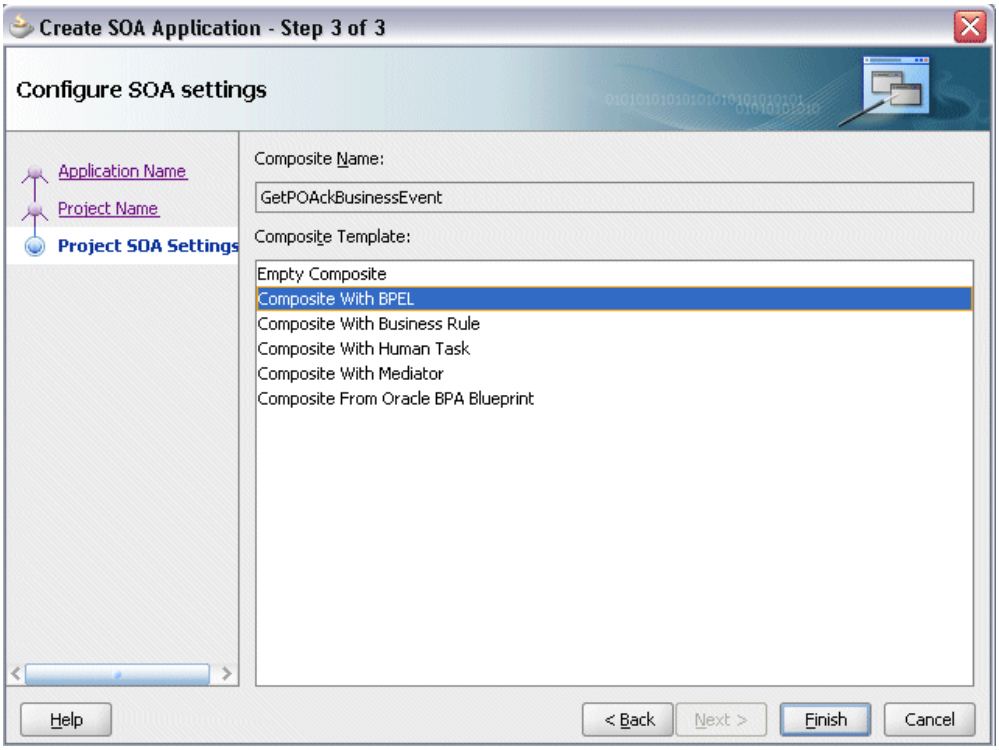
The Create SOA Application - Name your project Page



5. In the Project Technologies tab, ensure that **SOA** is selected from the Available technology list to the Selected technology list.

   Click **Next**. The Create SOA Application - Configure SOA settings page is displayed.

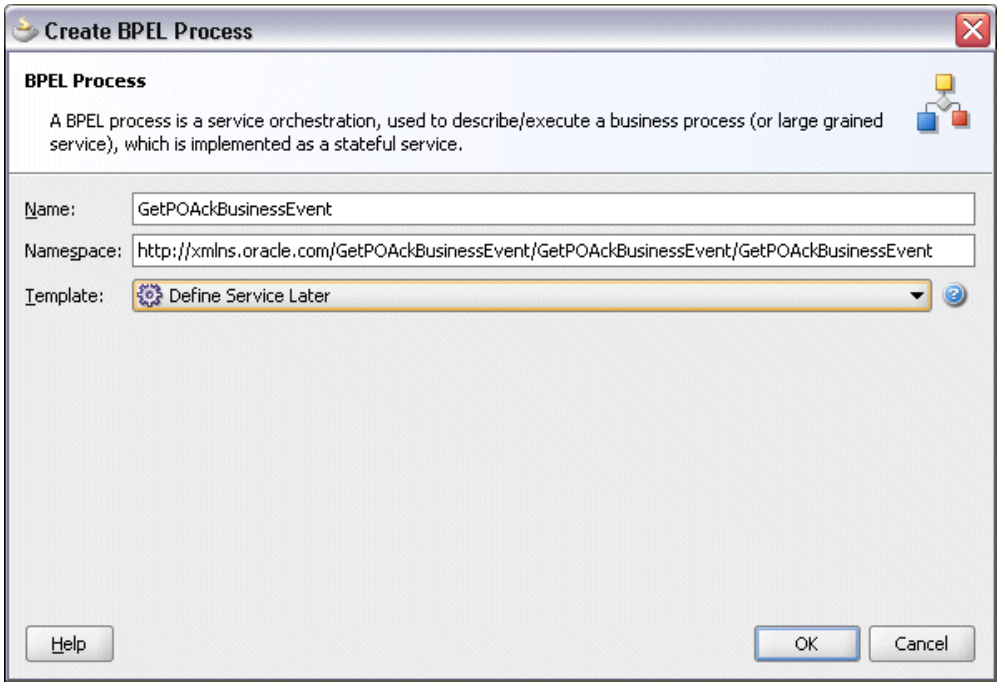
The Create SOA Application - Configure SOA settings Page



6. Click **Finish**. You have created a new application, and an SOA project. This automatically creates an SOA composite.

   The Create BPEL Process page is displayed.

The Create BPEL Process Page

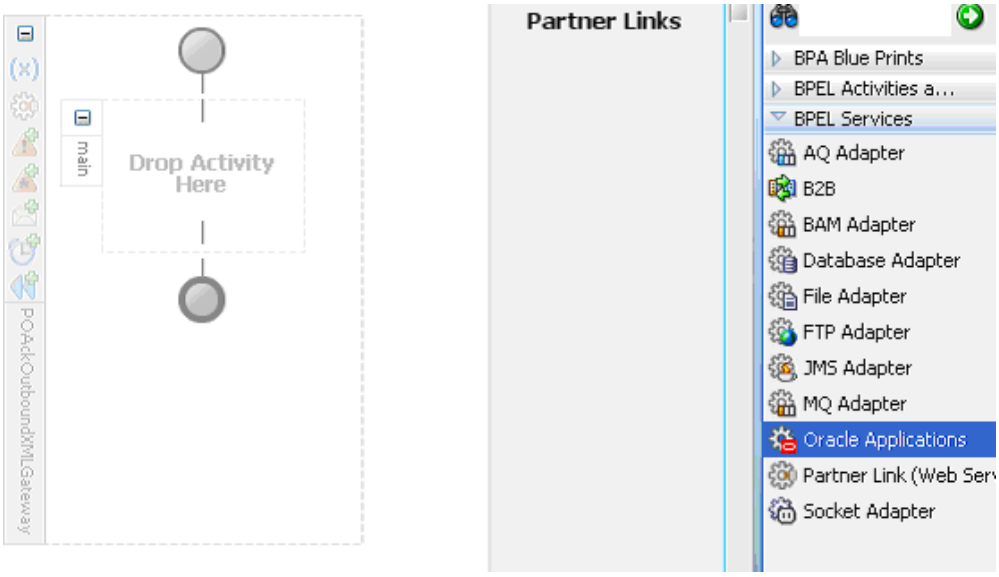


7. Enter an appropriate name for the BEPL process in the **Name** field. For example, `GetPOAckBusinessEvent`.

   Select **Define Service Later** from the **Template** field. Click **OK**.

   An empty BPEL process is created. The required source files including bpel and wsdl, using the name you specified (for example, `GetPOAckBusinessEvent.bpel` and `GetPOAckBusinessEvent.wsdl`) and `composite.xml` are also generated.

An Empty BPEL Process



Creating a Partner Link

Configuring an outbound business event requires creating a partner link to allow the outbound event to be published.

This task adds 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.

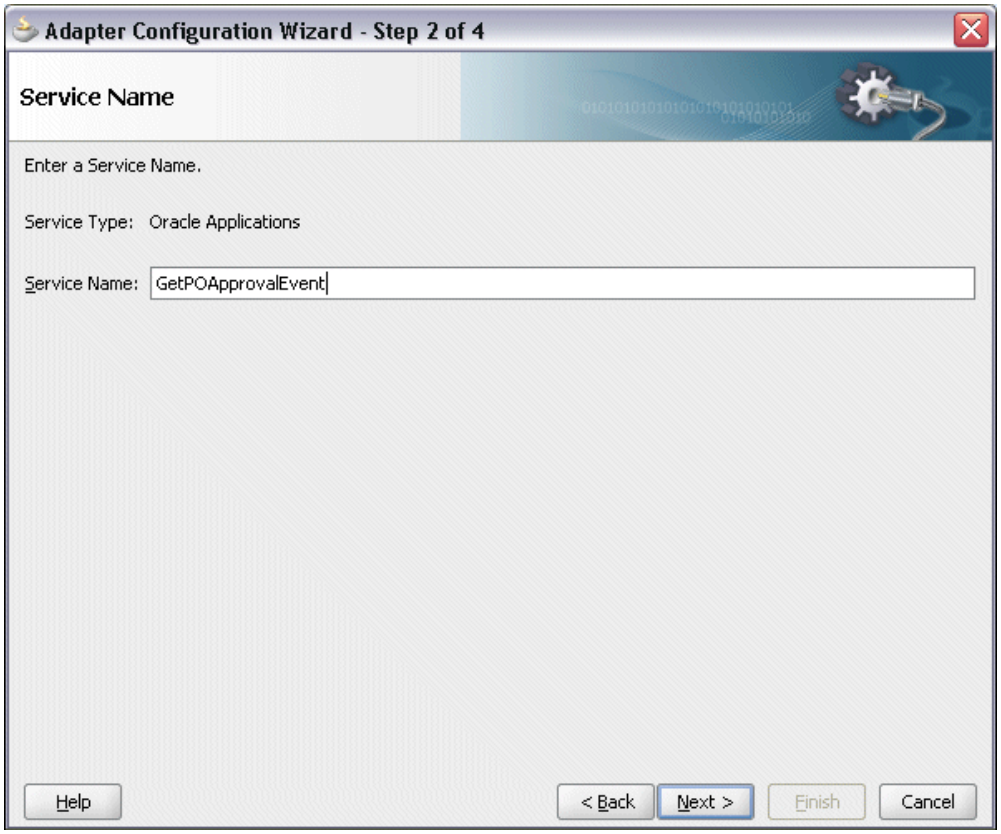
To add a partner link:

1. Click **BPEL Services** in the Component palette.

   Drag and drop **Oracle Applications** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears. Click **Next**.

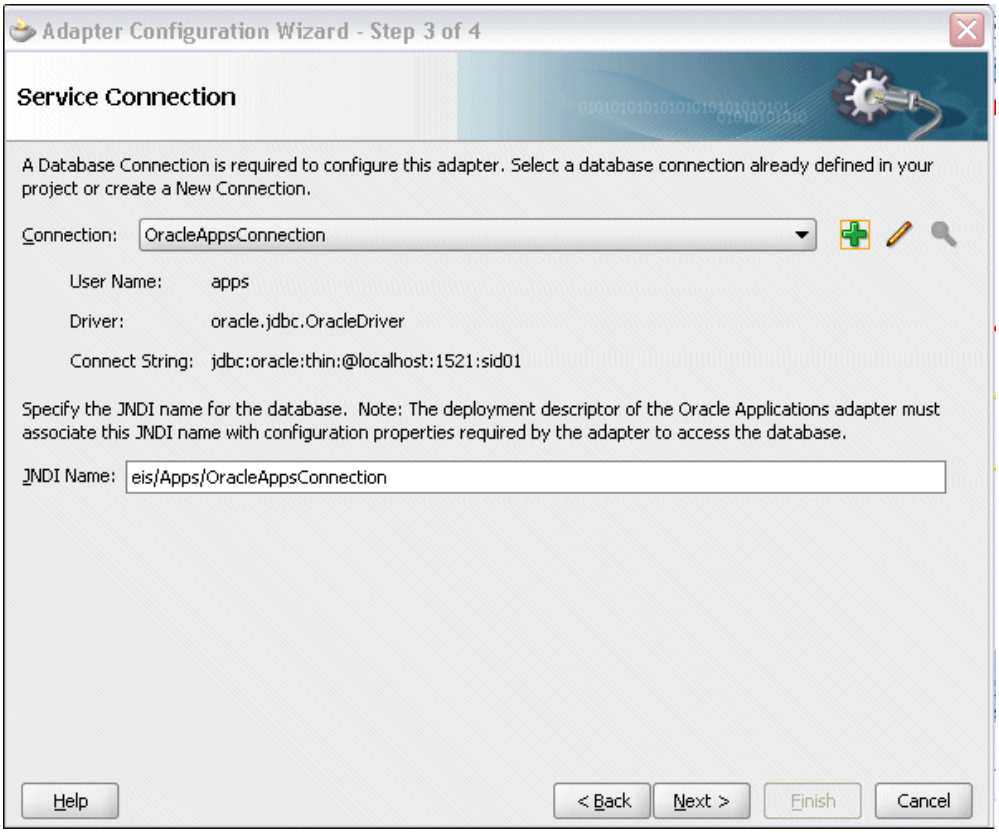
2. Enter a service name in the **Service Name** field. For example, `GetPOApprovalEvent`.

Enter the Service Name



Click **Next**. The Service Connection dialog appears.

Specifying a Database Service Connection



3. You can perform either one of the following options for your database connection:

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

   - You can create a new database connection by clicking the **Create a New Database Connection** icon.

     How to define a new database connection, see Create a New Database Connection, page 4-13.

   - You can select an existing database connection that you have configured earlier from the **Connection** drop-down list.

     The Service Connection page will be displayed with the selected connection information. The JNDI (Java Naming and Directory Interface) name corresponding to the database connection appears automatically in the **Database Server JNDI Name** field. Alternatively, you can specify a JNDI name.

Note: When you specify a JNDI name, the deployment descriptor of the Oracle Applications adapter must associate this JNDI name with configuration properties required by the adapter to access the database.

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.

Note: For more information about JNDI concepts, refer to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

4. Once you have completed creating a new connection for the service, you can add a business event by browsing through the list available in Oracle Applications.

   Click **Next**.

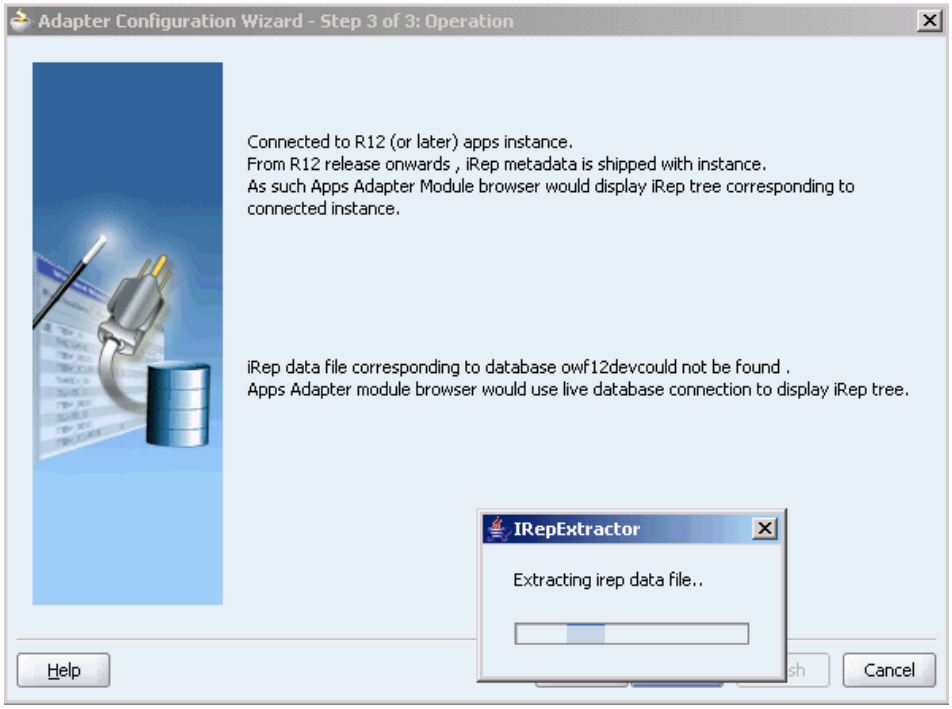
   For Oracle E-Business Suite Release 12:

   If you are connecting to Oracle E-Business Suite Release 12, then the **IREP File not present** dialog appears indicating that Adapter could not find the Oracle Integration Repository data file corresponding to the database you are connecting in your workspace. Absence of the data file would make browsing or searching of Integration Repository tree considerably slow. You can choose to extract the data file and create a local copy of the Integration Repository data file. Once it is created successfully, Adapter will pick it up automatically next time and retrieve data from your local Integration Repository.

   You can select one of the following options:

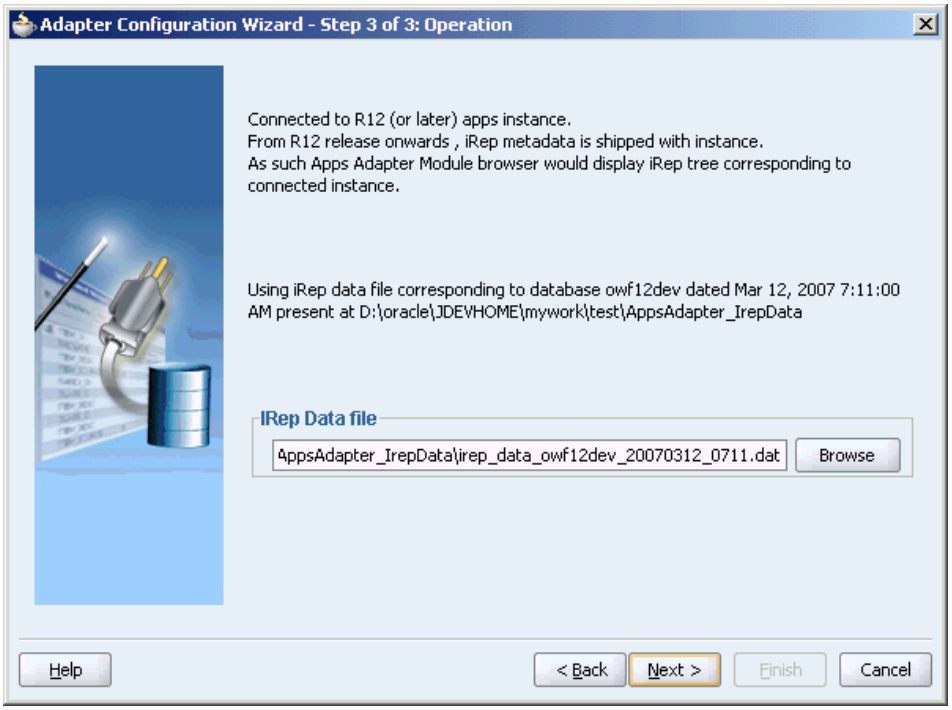
   - Click **Yes** to extract the Integration Repository data file.

Extracting Integration Repository Data File



After the system successfully creates a local copy of the Integration Repository data file, next time when you connect to the database, you will find the **IRep Data File** field appears in the Operation dialog indicating where your local copy exists with the creation date and time as part of the file name.

Using the Local Integration Repository Data File



- Click **No** to query the Integration Repository data file from the live database you are connecting to display the Integration Repository tree.

  Note: It is highly recommended that you create a local copy of the Integration Repository data file so that Adapter will query the data next time from the local copy in your workspace to enhance the performance.

  Click **Next** in the Operation page to open the Oracle Applications Module Browser.

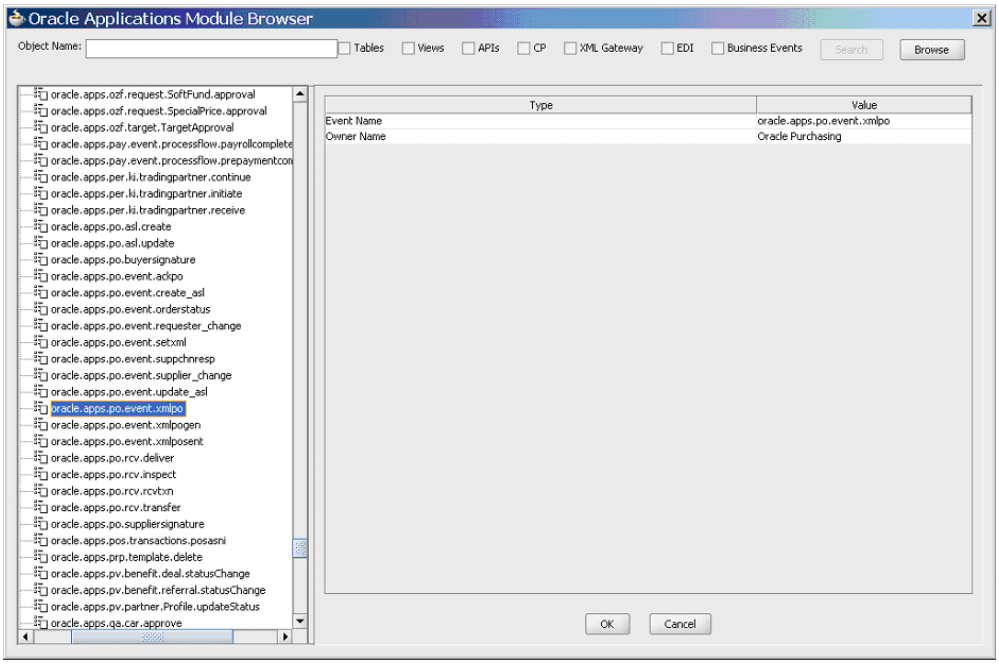
For Oracle E-Business Suite pre-Release 11.5.10:

If you are connecting to a pre-11.5.10 Oracle Applications instance, you must select the interface type in the Adapter Configuration Wizard. Select **Workflow Business Event System** to proceed.

Click **Get Object** in the Application Interface dialog to open the Oracle Applications Module Browser.

5. The Oracle Applications Module Browser combines interface data from Oracle Integration Repository with information about the additional interfaces supported by Oracle Application Adapter, organized in a tree hierarchy.

Oracle Applications Module Browser

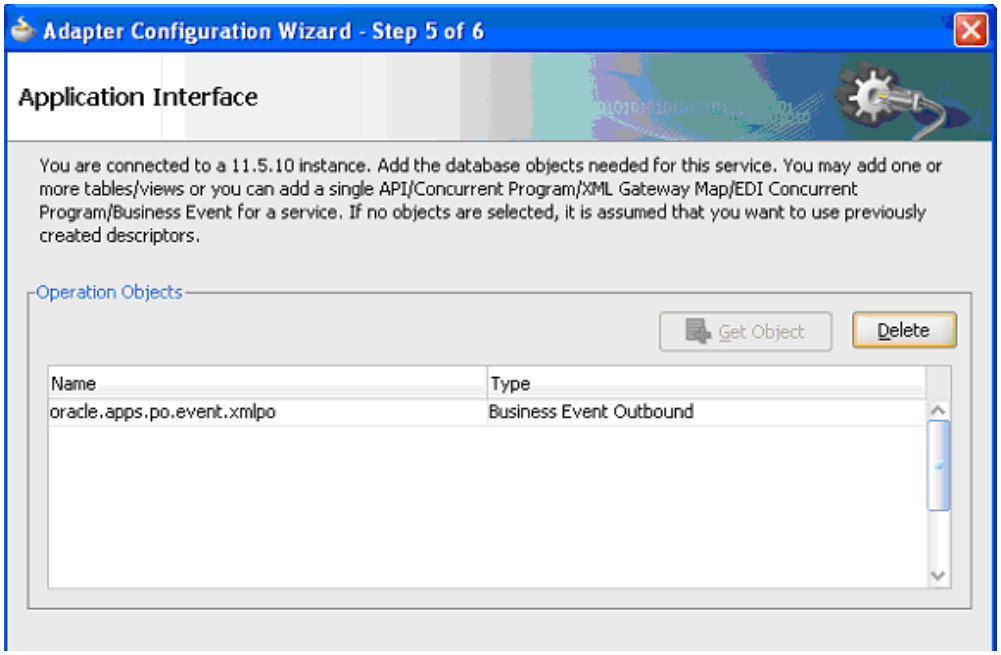


The Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. Each product family contains the individual products. Each product contains the business entities associated with the product. Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide.

> **Note:** Business Events can be found in the Other Interfaces node, which is at the product family level. For more information, see Using the Oracle Applications Module Browser, page 3-35.

6. Expand the navigation tree to **Product Families > Other Interfaces > Business Events > Outbound.** The direction outbound is from the Oracle E-Business Suite perspective, in this case listening to business events from Oracle Applications. Select the appropriate business event, for example, **oracle.apps.po.event.xmlpo**, and click **OK**. The Application Interface page is displayed with selected business event.

Adapter Configuration Wizard - Application Interface Page



7. Click **Next** in the Application Interface page. The WF Event Schema Definition page for business event payload appears.

8. You must specify one of the following options to be used for the business event payload:

> **Note:** When you select either the 'No Schema' or 'Any Schema' option, there is no need to further specify the schema information for your business event, and you will proceed to the next step.

- **No Schema**

  If you select the No Schema option, then the payload data would be available in the form of string. This option also allows you to receive non-XML event payload.

- **Any Schema**

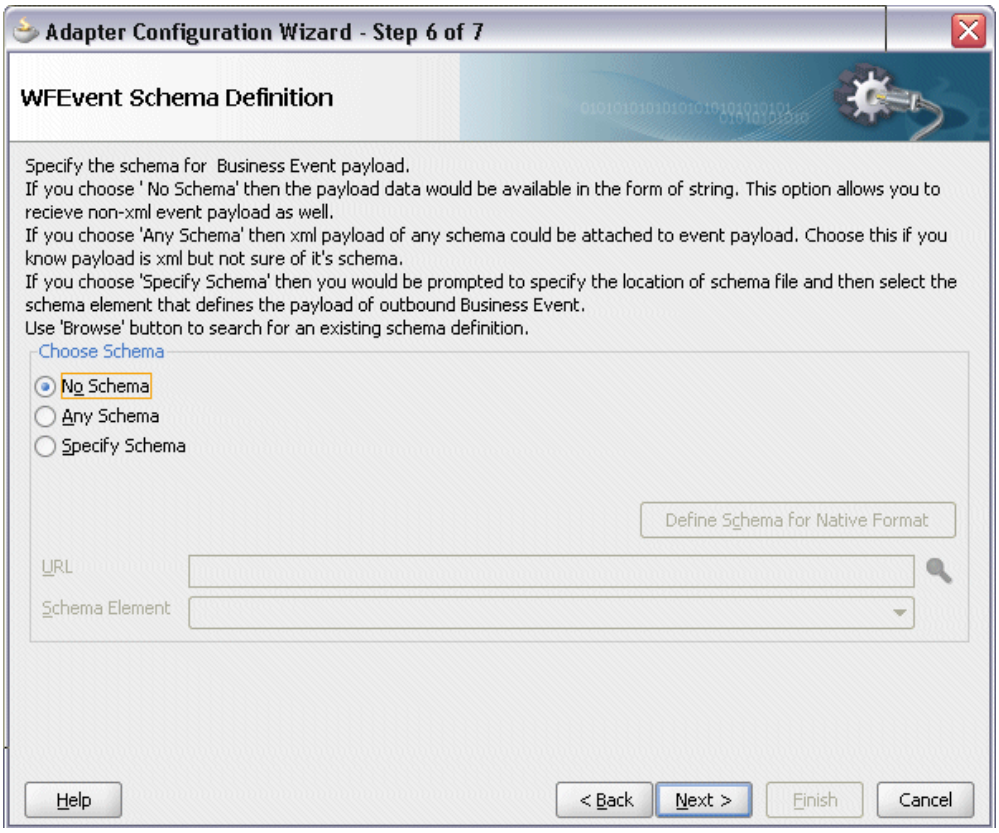
  If you select the Any Schema option, then XML payload of any schema could be attached to event payload. You should select this option if you know the payload is XML, but not sure of its schema.

- **Specify Schema**

  If you select the Specify Schema option, then the Schema Location and Schema

Element fields become visible. You must specify the location of schema file and then select the schema element that defines the payload of outbound business event.

Selecting Business Event Payload Schema



Select 'No Schema' and click **Next**.

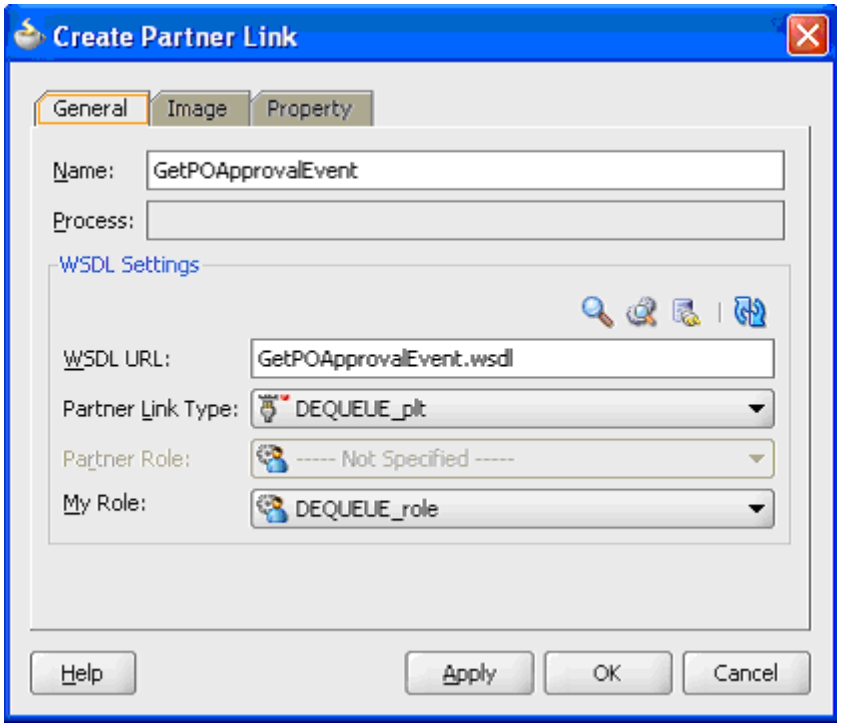
9. The Finish page appears indicating that you have finished defining the business event service.

   The wizard generates the GetPOApprovalEvent WSDL file corresponding to the **oracle.apps.po.event.xmlpo** business event service.

   The main Create Partner Link dialog appears with the new WSDL file.

   Click **Apply** and then **OK** to complete the partner link configuration.

Partner Link Information



The partner link is created with the required WSDL settings, and is represented in the BPEL project by a new icon in the border area of the process diagram.

Configuring the Receive Activity

The next task is to configure a Receive activity to receive event details from the partner link that you just configured for the Oracle Application adapter service as an input to the Assign activity.

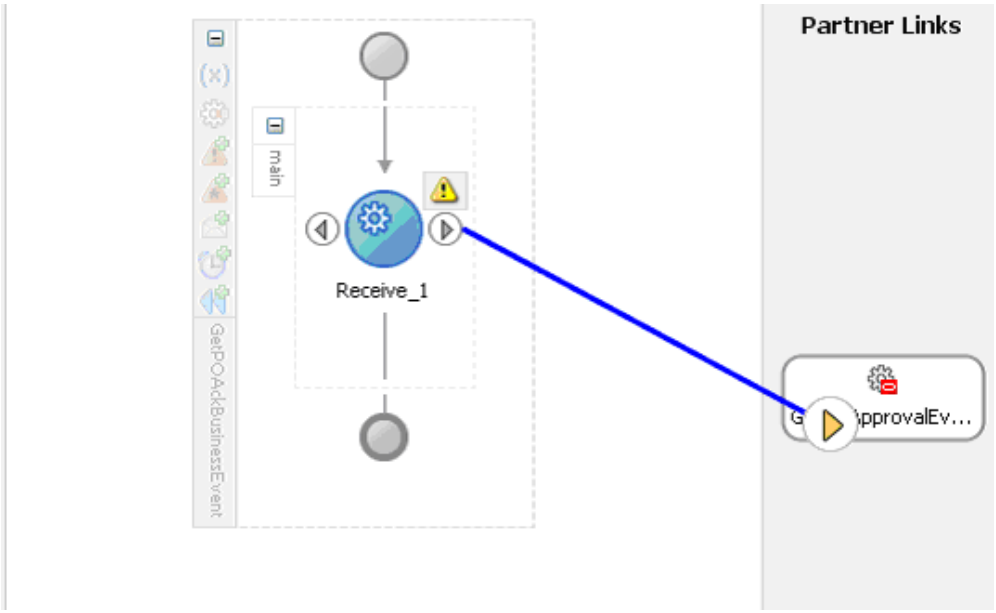
To configure the Receive activity:

1. In JDeveloper BPEL Designer, click **BPEL Services and Components** in the Component palette.

   Drag and drop **Receive** from the BPEL activity list into the center swim lane of the process diagram.

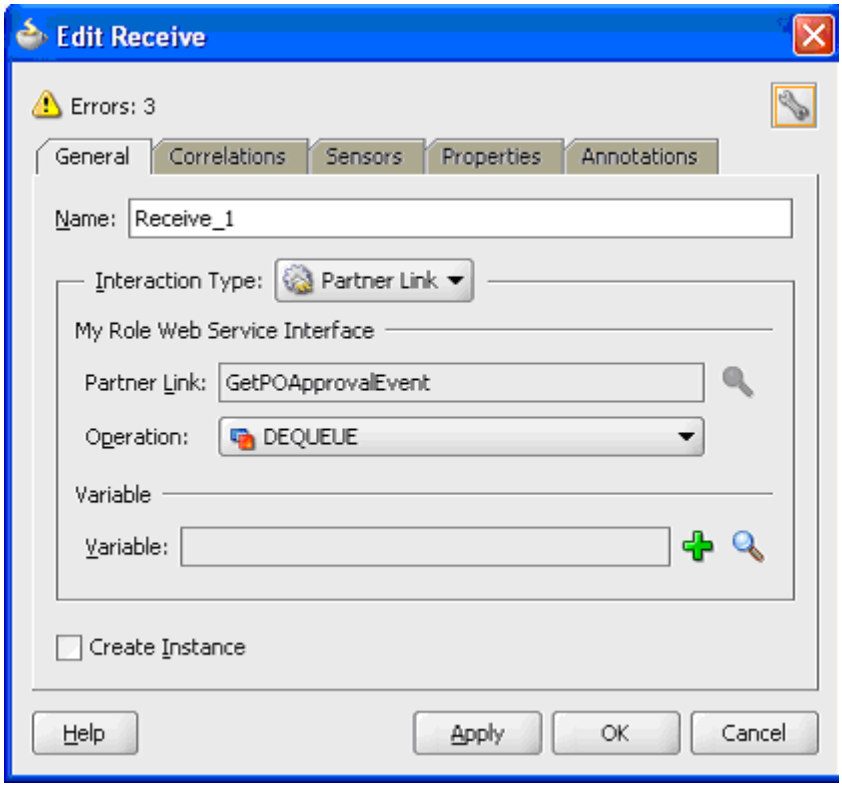
2. Link the **Receive** activity to the partner link `GetPOApprovalEvent` that you just created earlier.

Associating the Receive Activity with the Partner Link



The Receive dialog appears.

Configuring the Receive Activity



3. Enter an appropriate name for the Receive activity.

   The Dequeue **Operation** is automatically selected since the partner link has been configured with an outbound business event.

4. Specify a **Variable** to receive the message data from the partner link by clicking the **Create** icon to the right of the Variable field. The **Create Variable** dialog box appears.

5. Click **OK** to accept the default name.

6. Click **Apply** in the Receive dialog, then click **OK**.

Adding a Partner Link for the File Adapter

If you are configuring an outbound business event, you need to add another partner link for the file adapter. This allows the outbound business event to write the data to the XML file.

To add a partner link for the file adapter:

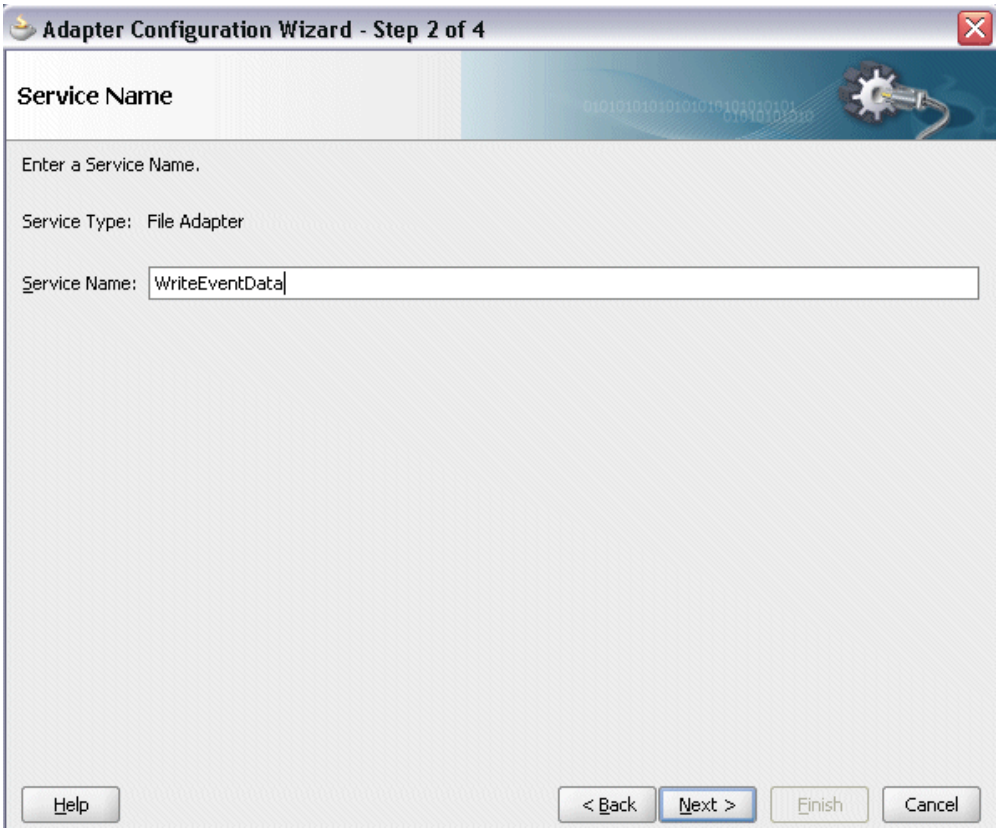
1. In JDeveloper BPEL Designer, click **BPEL Services** in the Component palette.

   Drag and drop **File Adapter** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears.

   Click **Next**.

2. In the Service Name page, enter a name for the file adapter service. For example, enter `WriteEventData`.

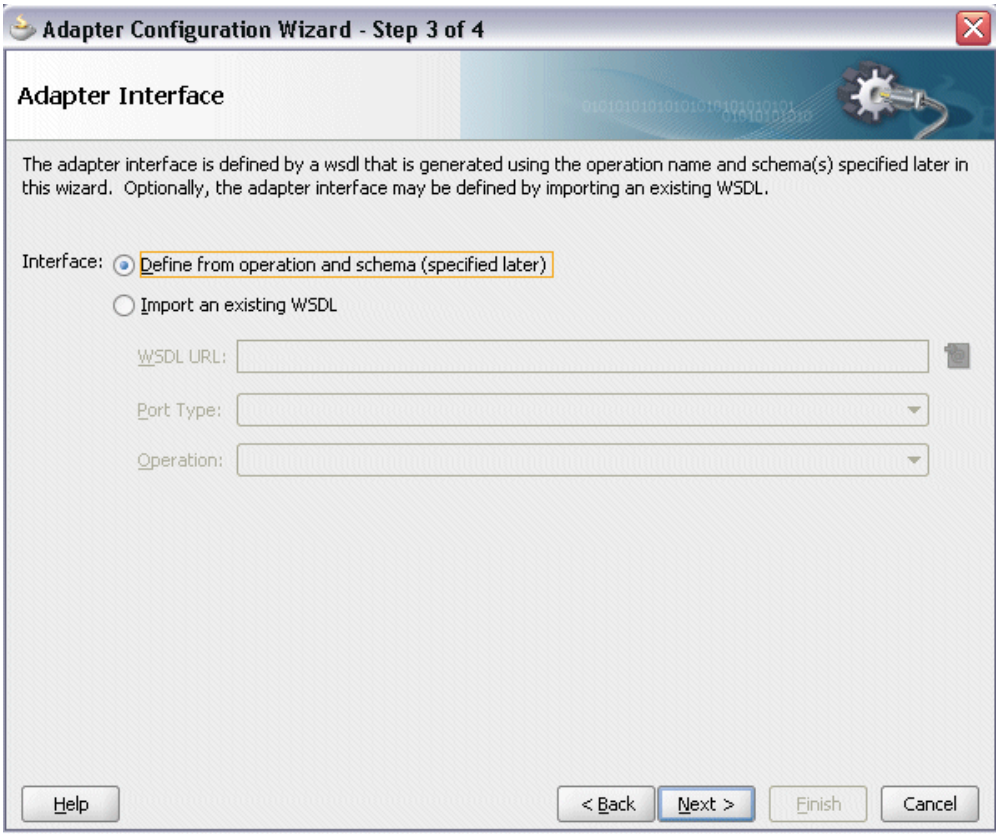
   Specifying the Service Name



3. Enter a name for the file adapter service, for example, `WriteEventData`.

4. Click **Next**. The Adapter Interface page appears.

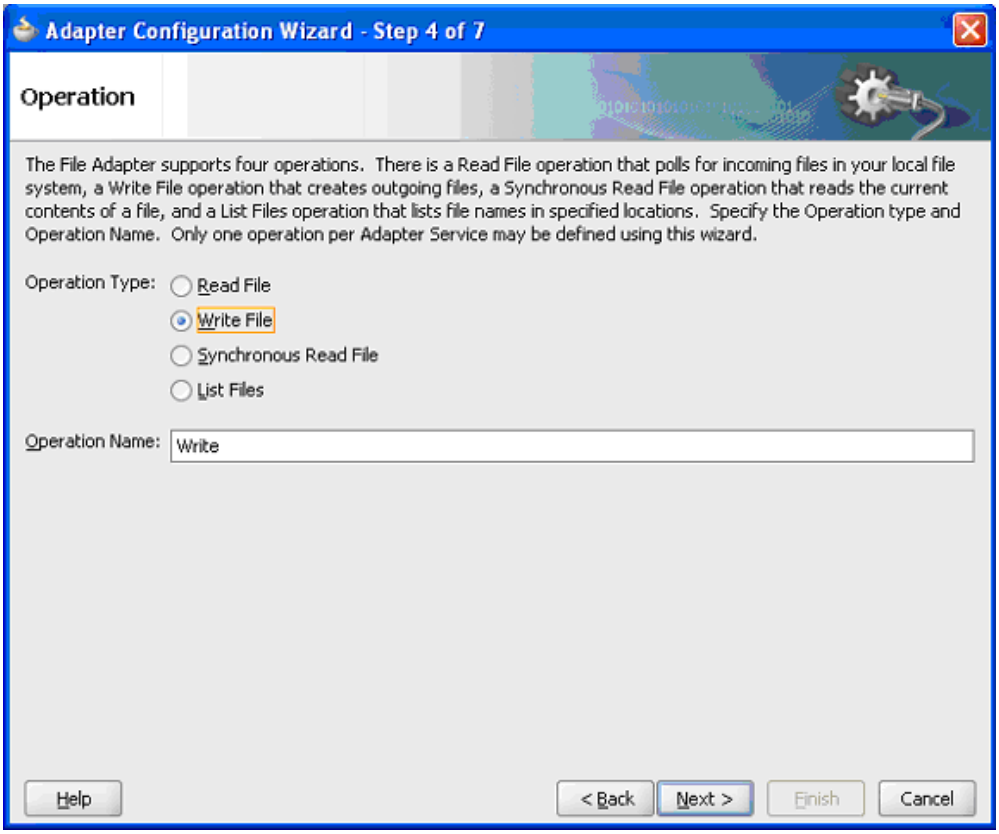
Specifying the Adapter Interface



Select the **Define from operation and schema (specified later)** radio button and click **Next**.

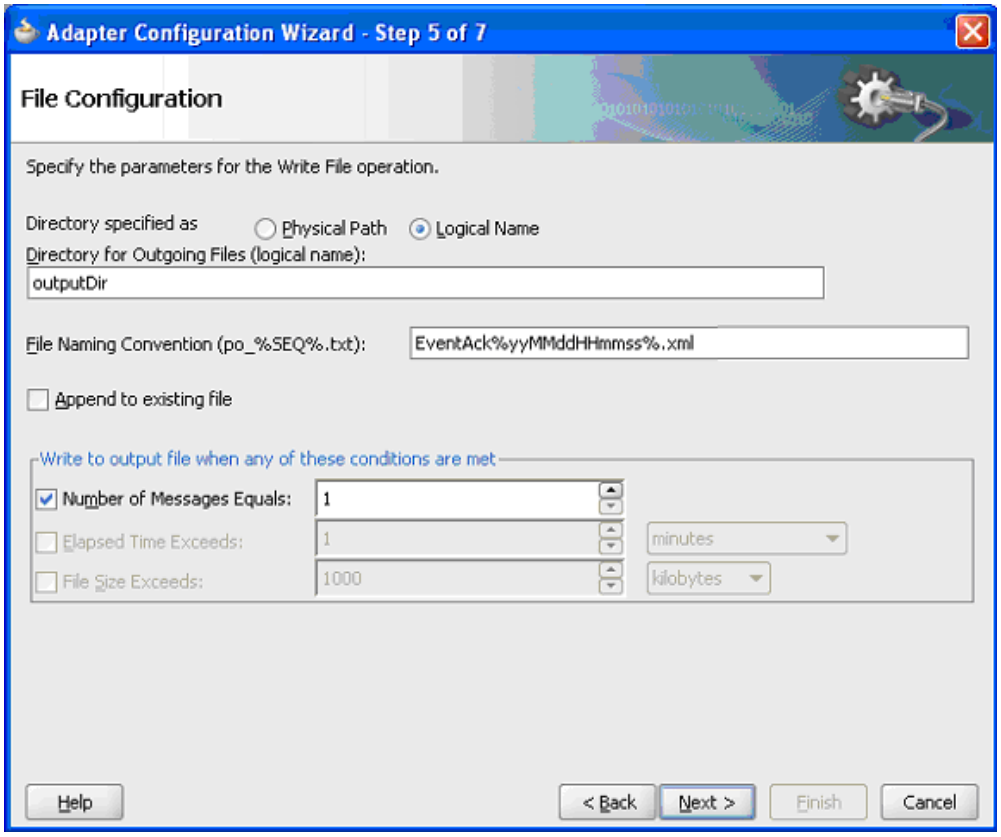
5. In the Operation page, specify the operation type. For example, select the **Write File** radio button. This automatically populates the **Operation Name** field.

Specifying the Operation



6. Click **Next** to access the File Configuration page.

Configuring the Output File



7. For the **Directory specified as** field, select the **Logical Name** radio button. Enter `outputDir` as the **Directory for Outgoing Files (logical name)** and specify a naming convention for the output file, such as `EventAck%yyMMddHHmmss%.xml`.

   > **Tip:** When you type a percent sign (`%`), you can choose from a list of date variables or a sequence number variable (`SEQ`) as part of the filename.

   Confirm the default write condition: **Number of Messages Equals 1**.

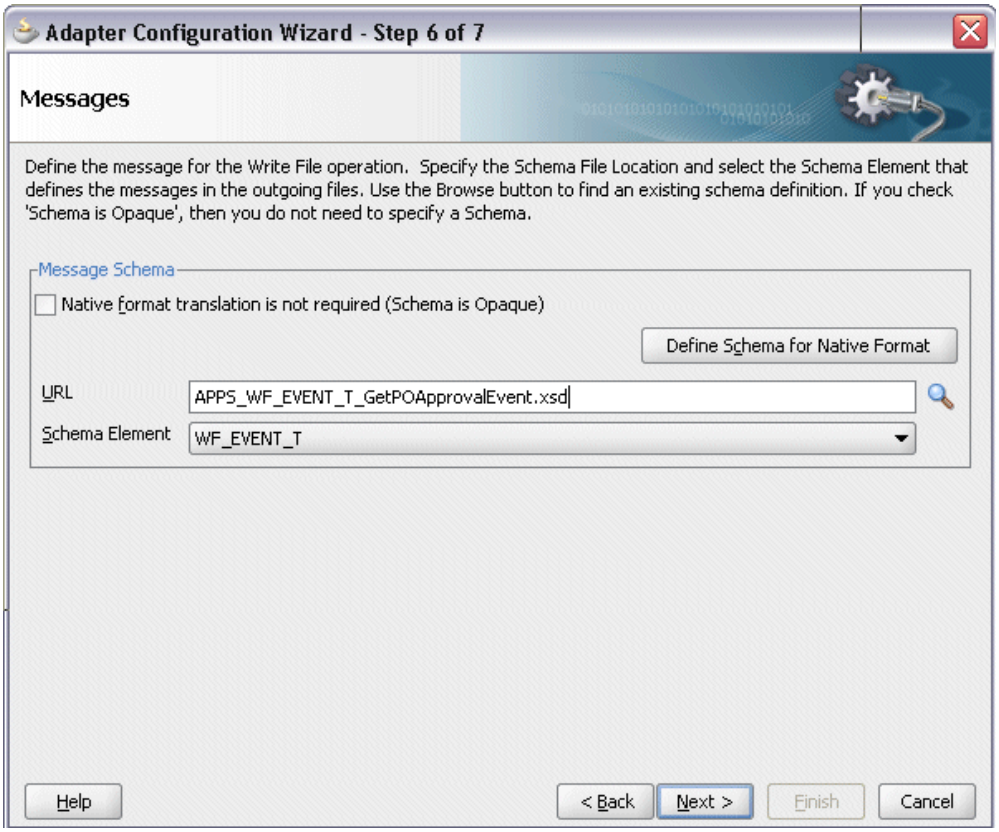
8. Click **Next**, and the Messages page appears. For the output file to be written, you must provide a schema.

9. Click **Browse** to access the Type Chooser.

10. Expand the node by clicking **Project Schema Files > APPS_WF_EVENT_T_GetPOApprovalEvent.xsd**. Select **WF_EVENT_T** as the element and click **OK**.

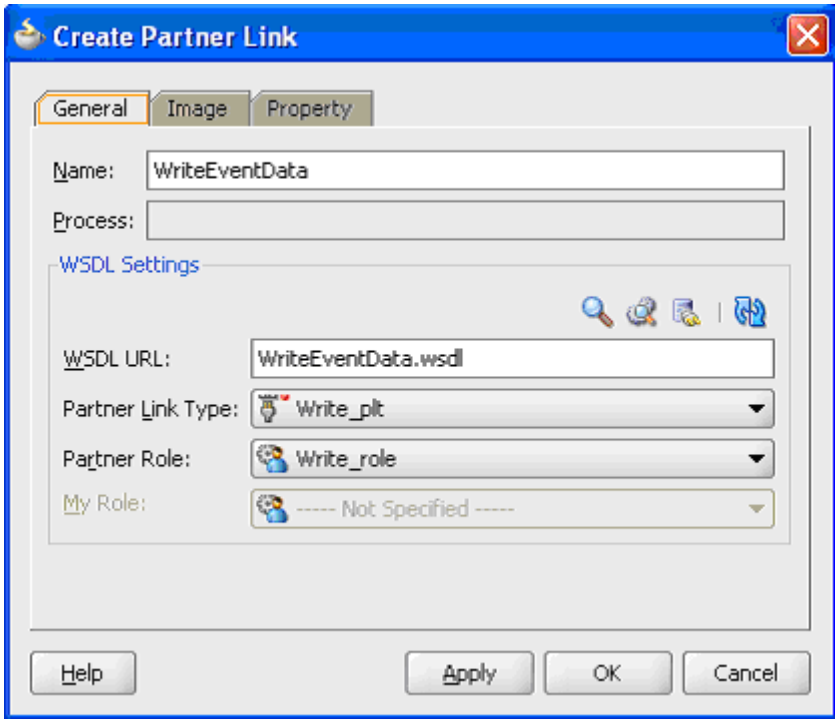
The selected schema information will be automatically populated in the URL and Schema Element fields.

Populating the Selected Message Schema



11. Click **Next** and then **Finish**. The wizard generates the WSDL file corresponding to the partner link. The main Create Partner Link dialog box appears, specifying the new WSDL file.

Completing the Partner Link Configuration



12. Click **Apply** and then **OK** to complete the configuration and create the partner link with the required WSDL settings for the File Adapter service.

Configuring the Invoke Activity

After adding the File Adapter partner link, you need to configure an Invoke activity to associate it with the File Adapter link.

Through the Invoke activity, the business event information can be written to the XML file you specified as the output directory.

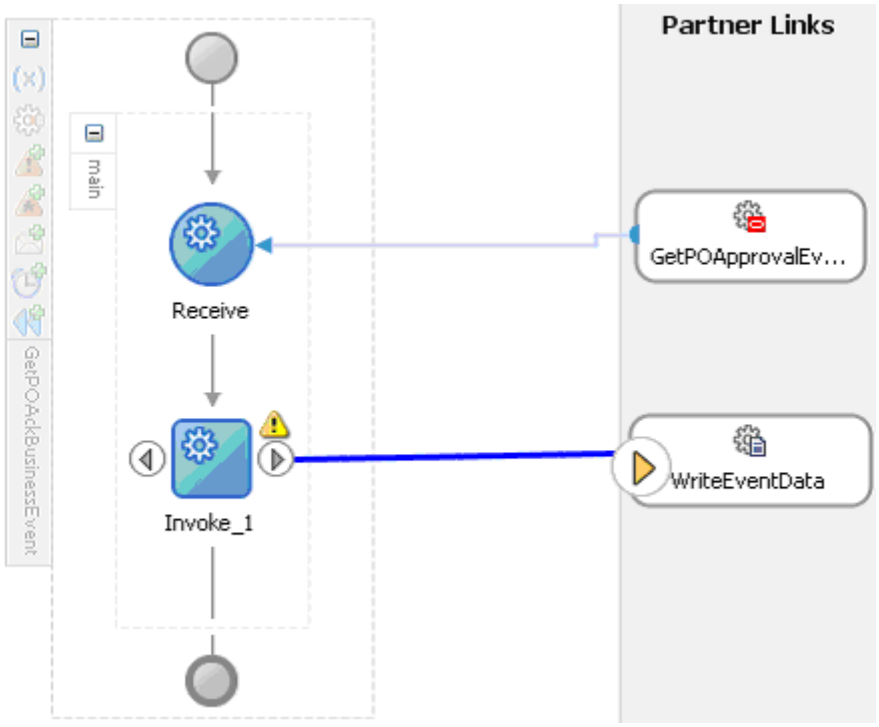
To configure the Invoke activity:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette.

   Drag and drop an **Invoke** activity into the center swim lane of the process diagram after the **Receive** activity.

2. Link the Invoke activity to the `WriteEventData` File Adapter service.

Associating Invoke with the File Adapter Link

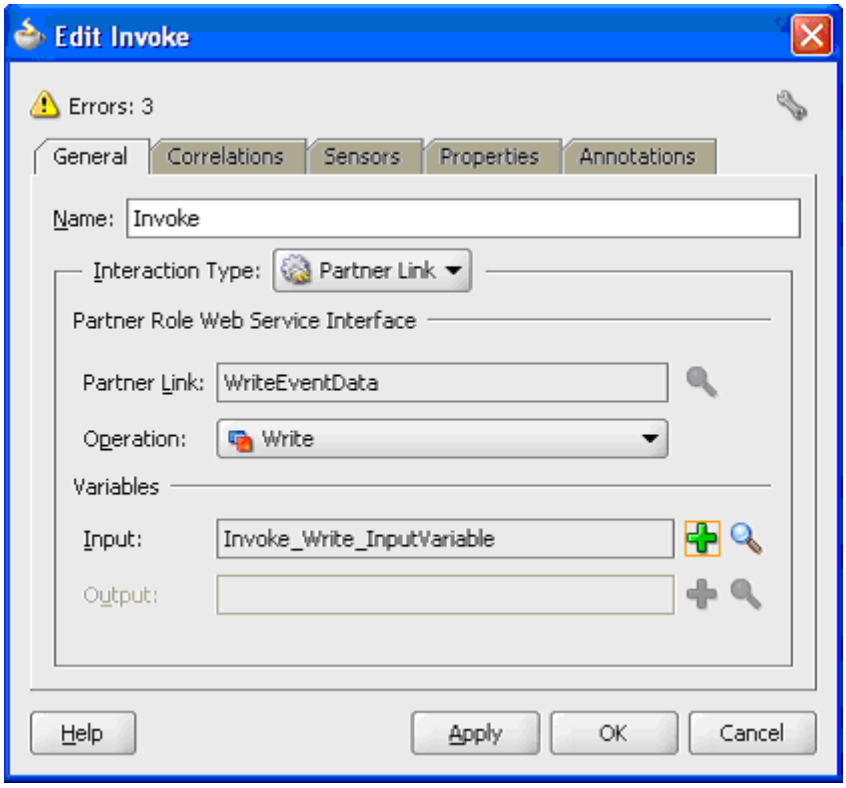


The Invoke activity will send event data to the partner link. The Edit Invoke dialog appears.

3. Enter a name for the Invoke activity, then click the **Create** icon next to the **Input Variable** field to create a new variable. The Create Variable dialog appears.

4. Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK** to return to the Edit Invoke dialog.

Editing the Invoke Activity



Click **Apply** and then **OK** to finish configuring the Invoke activity.

Configuring the Assign Activity

Use the Assign activity to take the output from the Receive activity and to provide input to the invoke activity.

To configure the Assign activity:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette.

   Drag and drop the **Assign** activity into the center swim lane of the process diagram, between the **Receive** activity and the **Invoke** activity.

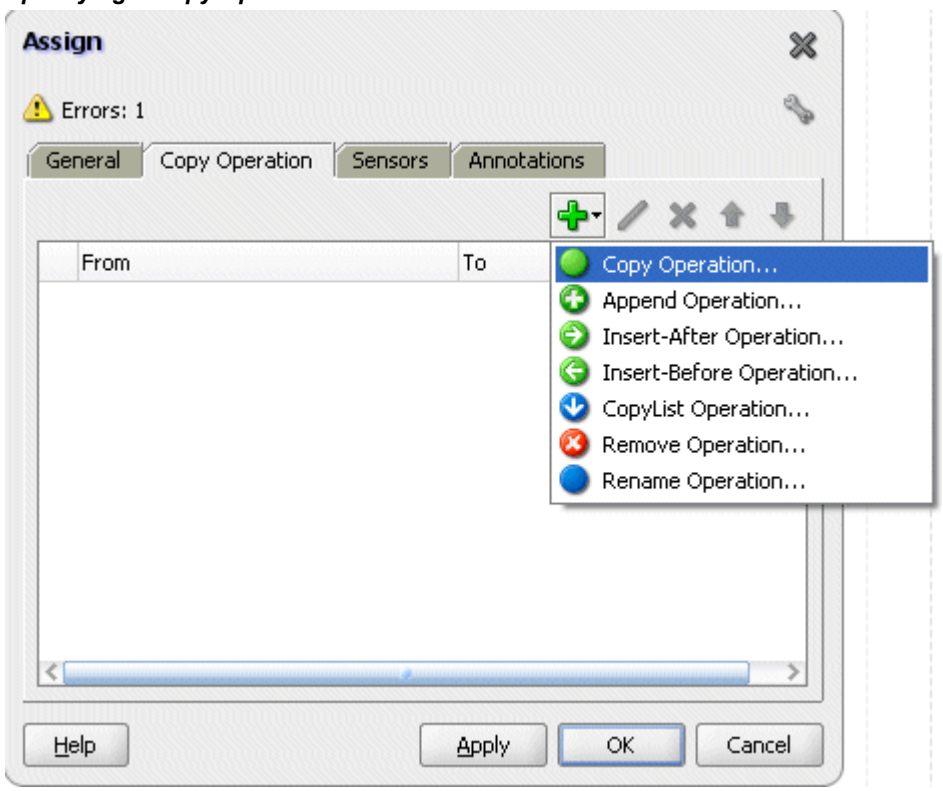
2. Double-click the **Assign** activity to access the Edit Assign dialog.

   Click the General tab to enter a name for the Assign activity. For example, `setEventData`.

3. Select the Copy Operation tab, click the 'Plus' sign icon and select **Copy Operation**

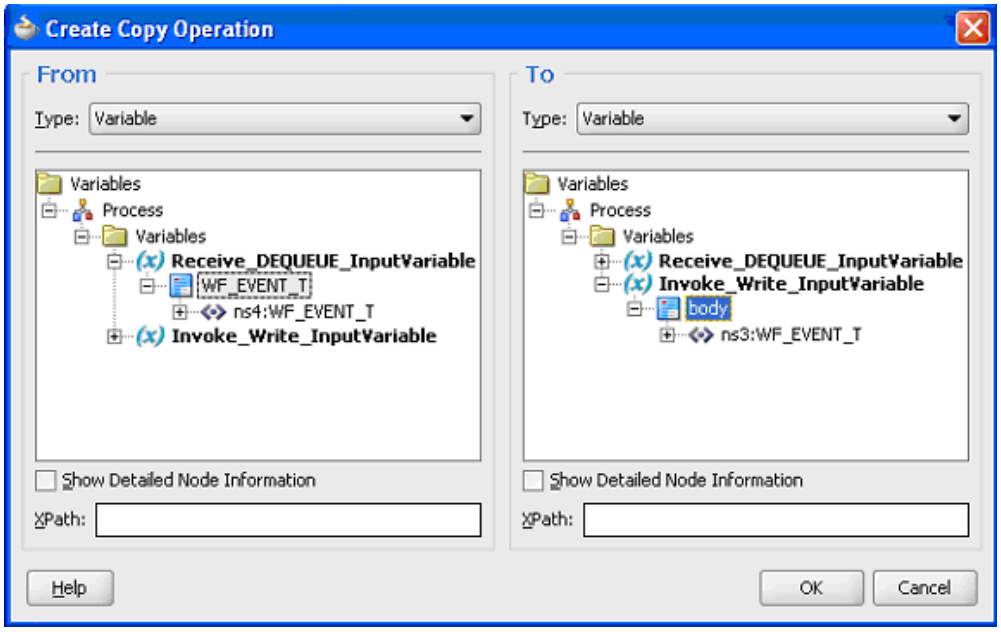
from the menu. The Create Copy Operation window appears.

Specifying a Copy Operation Action



4. In the From navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Receive_DEQUEUE_InputVariable > WF_EVENT_T** and select **ns4:WF_EVENT_T**. The XPath field should contain your selected entry.

Defining the Copy Operation

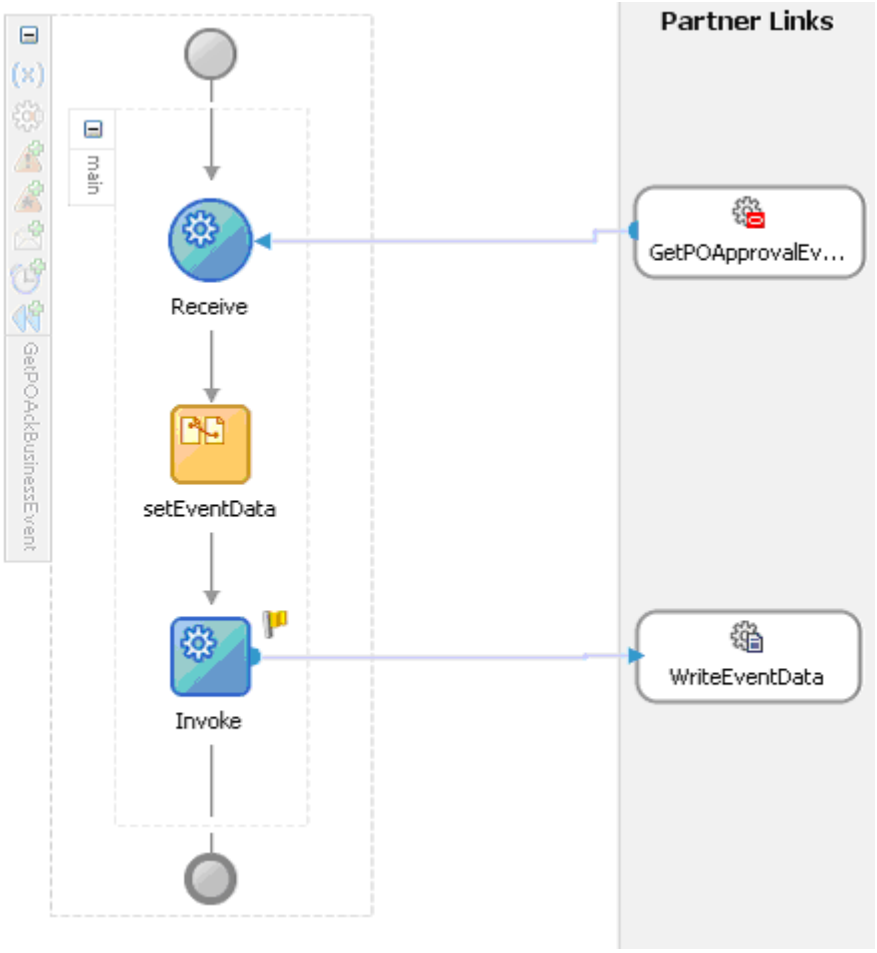


5. In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_Write_InputVariable > WF_EVENT_T** and select **ns3:WF_EVENT_T**. The XPath field should contain your selected entry.

6. Click **OK**.

   Click **Apply** and then **OK** in the Edit Assign dialog box to complete the configuration of the Assign activity.

Completed Outbound Business Event BPEL Process Project



Click the `composite.xml` to display the Oracle JDeveloper composite diagram:

> **Note:** Click the Source tab of `composite.xml` to enter a value for the physical directory `outputDir` for the reference WriteEventData (such as `/usr/tmp`).
>
> ```
> <property name="outputDir" type="xs:string"
> many="false" override="may">/usr/tmp</property>
> ```

Specifying the Physical Directory for the Property

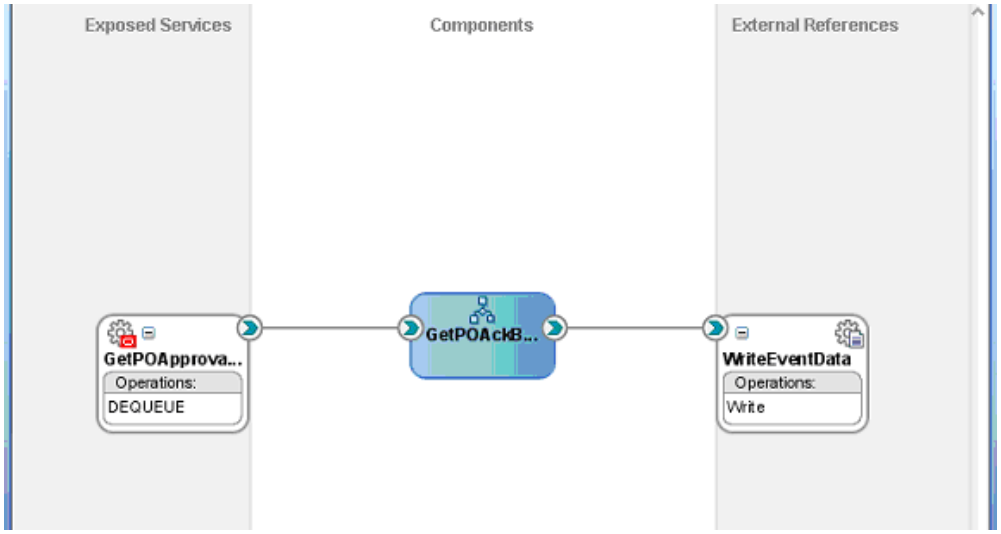
```
  xmlns:ui="http://xmlns.oracle.com/soa/designer/">
 namespace="http://xmlns.oracle.com/pcbpel/adapter/apps/GetPOAckBusinessEvent-App/ada
 location="GetPOApprovalEvent.wsdl" importType="wsdl"/>
 namespace="http://xmlns.oracle.com/pcbpel/adapter/file/GetPOAckBusinessEvent-App/ada
 location="WriteEventData.wsdl" importType="wsdl"/>
e name="GetPOApprovalEvent" ui:wsdlLocation="GetPOApprovalEvent.wsdl">
rface.wsdl interface="http://xmlns.oracle.com/pcbpel/adapter/apps/GetPOAckBusinessEve
ing.jca config="GetPOApprovalEvent_apps.jca"/>
ce>
ent name="GetPOAckBusinessEvent">
mentation.bpel src="GetPOAckBusinessEvent.bpel"/>
erty name="activationAgent.GetPOApprovalEvent.className"
     type="xs:string" many="false">oracle.tip.adapter.fw.agent.jca.JCAActivationAgent
erty name="activationAgent.GetPOApprovalEvent.portType"
     type="xs:string" many="false">DEQUEUE_ptt</property>
nent>
nce name="WriteEventData" ui:wsdlLocation="WriteEventData.wsdl">
rface.wsdl interface="http://xmlns.oracle.com/pcbpel/adapter/file/GetPOAckBusinessEve
ing.jca config="WriteEventData_file.jca"/>
erty name="outputDir" type="xs:string" many="false" override="may">/usr/tmp</property
ence>

ce.uri>GetPOApprovalEvent</source.uri>
et.uri>GetPOAckBusinessEvent/GetPOApprovalEvent</target.uri>


ce.uri>GetPOAckBusinessEvent/WriteEventData</source.uri>
et.uri>WriteEventData</target.uri>

te>
```

Oracle JDeveloper Composite Diagram



Run-Time Tasks for Outbound Business Events

After designing the BPEL process, you can compile, deploy and test it.

1. Deploy the BPEL process, page 5-34.

2. Test the BPEL process, page 5-36.

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 application server (Oracle WebLogic Server) that you have established the connection.

Prerequisites

Before deploying the BPEL process using Oracle JDeveloper, you must ensure the following:

- You must have established the connectivity between the deign-time environment and an application server.

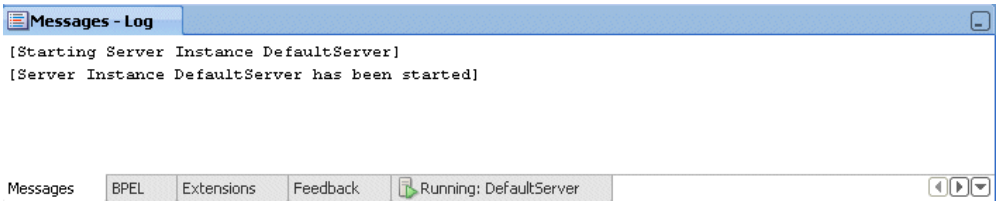
  For more information, see Configuring the Data Source in Oracle WebLogic Server, page A-3 and Creating an Application Server Connection, page A-8.

- Oracle WebLogic Server has been started.

  Before deploying the BPEL process, you need to start the Oracle WebLogic Server that you have established the connection.

  If a local instance of the WebLogic Server is used, start the WebLogic Server by selecting **Run > Start Server Instance** from Oracle JDeveloper.

  Once the WebLogic Admin Server "DefaultServer" instance is successfully started, you should find that <Server started in Running mode> and DefaultServer started message in the Running:DefaultServer and Messages logs.



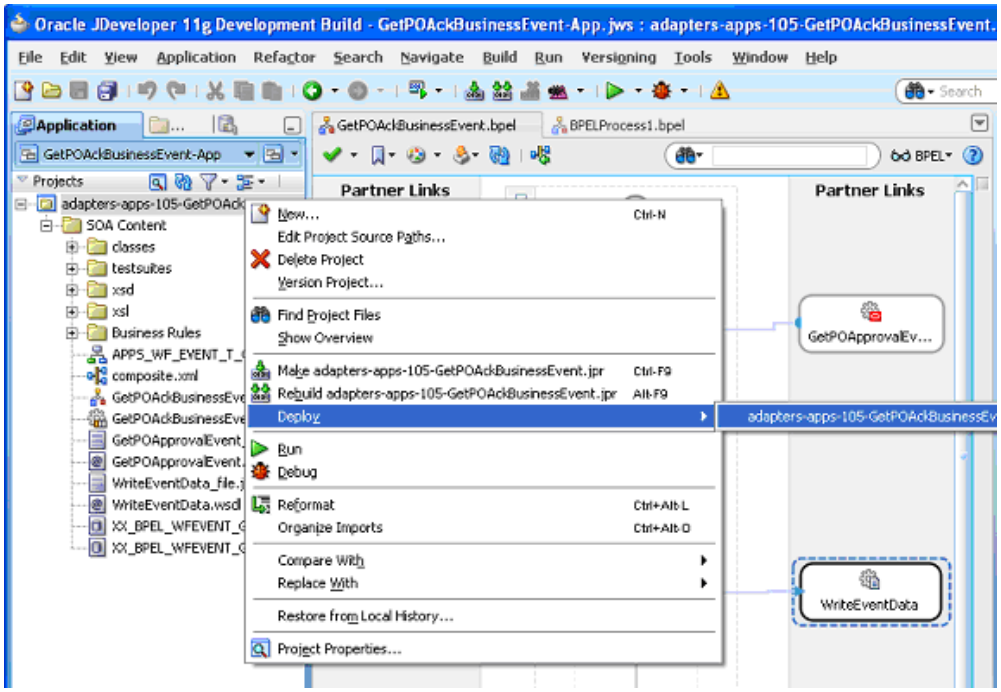
To deploy the BPEL process:

1. Select your BPEL project in the Applications Navigator.

   In the Applications Navigator of JDeveloper BPEL Designer, select your BPEL project (such as **GetPOAckBusinessEvent**) project.

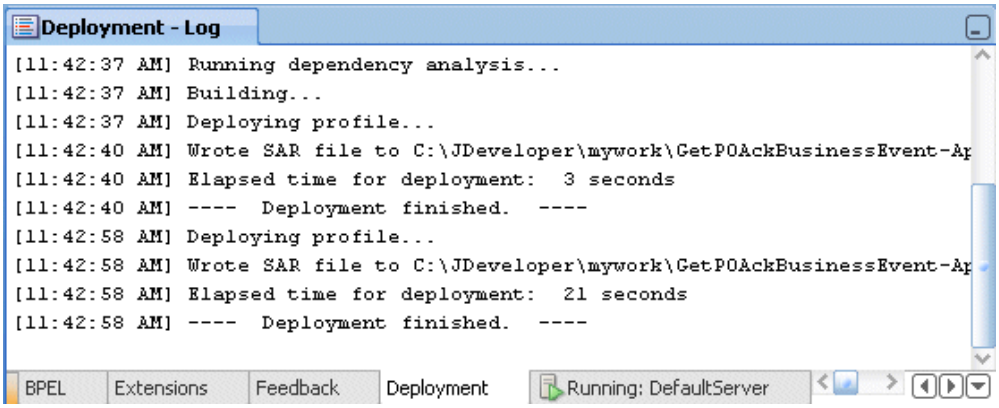
2. Right-click the project name, and then select **Deploy > [project name] > [serverConnection]** from the menu that appears.

   Deploying the BPEL Process



3. The BPEL project is compiled and successfully deployed.

Compilation and Deployment Message Logs

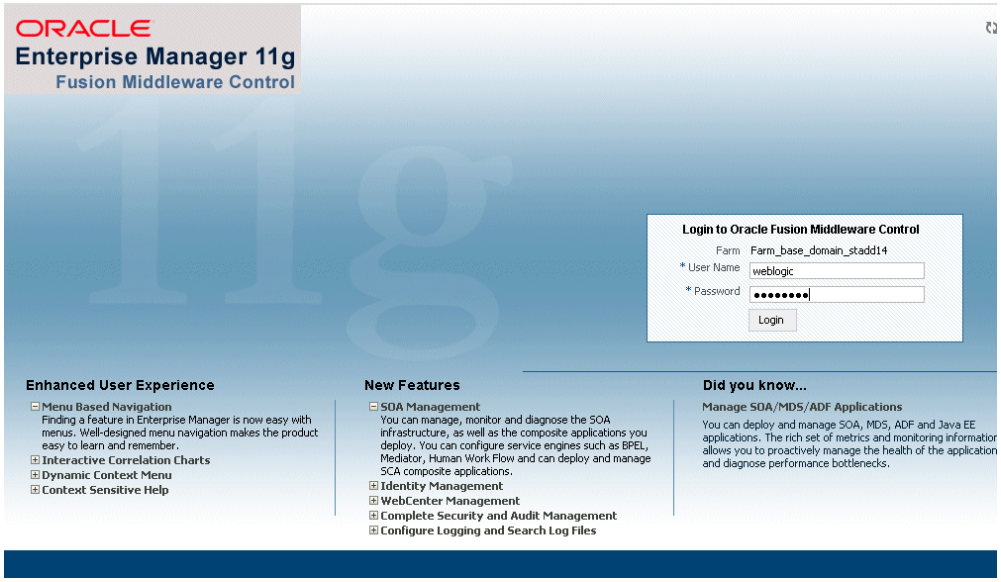


Testing the BPEL Process

Once the BPEL process is deployed, you can manage and monitor the process from the Oracle Enterprise Manager Fusion Middleware Control Console. You can also test the process and the integration interface by manually initiating the process.

To test the BPEL process:

1. Navigate to Oracle Enterprise Manager Fusion Middleware Control Console (`http://<servername>:<portnumber>/em`). The composite you deployed is displayed in the Applications Navigation tree.



2. Enter username (such as `weblogic`) and password and click **Login** to log in to a farm.

   You may need to select an appropriate target instance farm if there are multiple target Oracle Enterprise Manager Fusion Middleware Control Console farms.

3. From the Farm base domain, expand the **SOA** >**soa-infra** to navigate through the SOA Infrastructure home page and menu to access your deployed SOA composite applications running in the SOA Infrastructure for that managed server.

   > **Note:** The Farm menu always displays at the top of the navigator. When you expand the SOA folder in the navigator and click the links displayed beneath it, the SOA Infrastructure menu becomes available at the top of the page.

4. Log on to Oracle Applications with the XML Gateway responsibility.

   This is to ensure that the XML Gateway trading partner is set up correctly so that a purchase order can have a valid supplier that has been defined.

5. Select Define Trading Partner from the navigation menu to access the Trading Partner Setup window.

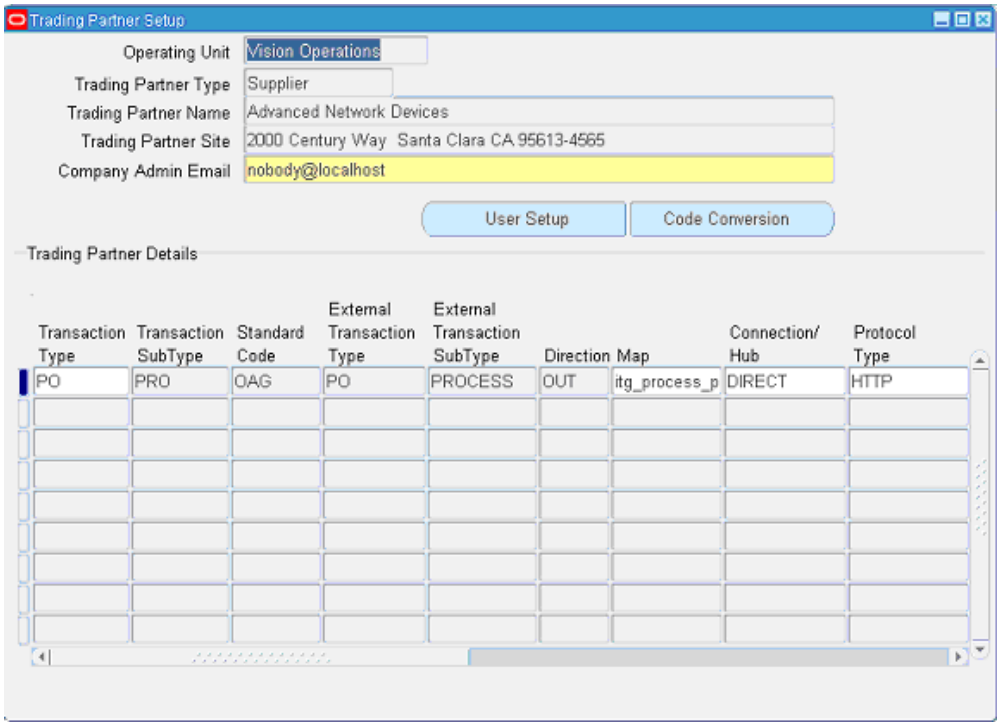
6. Enter the header values on the Trading Partner Setup form as follows:

   - Trading Partner Type: Supplier
   - Trading Partner Name: For example, Advanced Network Devices

- Trading Partner Site: Enter a trading partner site information. For example, 2000 Century Way, Santa Clara, CA 95613-4565
- Company Admin Email: Enter a valid email address.

7. Enter the following trading partner details:
   - Transaction Type: PO
   - Transaction SubType: PRO
   - Standard Code: OAG
   - External Transaction Type: PO
   - External Transaction SubType: Process
   - Direction: Out
   - Map: itg_process_po_007_out
   - Connection / Hub: DIRECT
   - Protocol Type: HTTP
   - Username: 'operation'
   - Password: 'welcome'. Password has to be entered twice before moving to next field.
   - Protocol Address: 'http://appsadapter.sample.com'
   - Source Trading partner location code: STPLC

Trading Partner Setup Form



Save the trading partner details.

8. Switch responsibility by selecting the Purchasing, Vision Operations (USA) and select Purchase Order from the navigation menu.

   The Oracle Applications Forms open up with the Purchase Order forms.

9. Create a purchase order with the header values reflecting the trading partner you previously defined in the Purchase Order window:

   - Supplier: Enter a supplier information, such as 'Advanced Network Devices'.
   - Site: Select a site information, such as 'SANTA CLARA-ERS'.

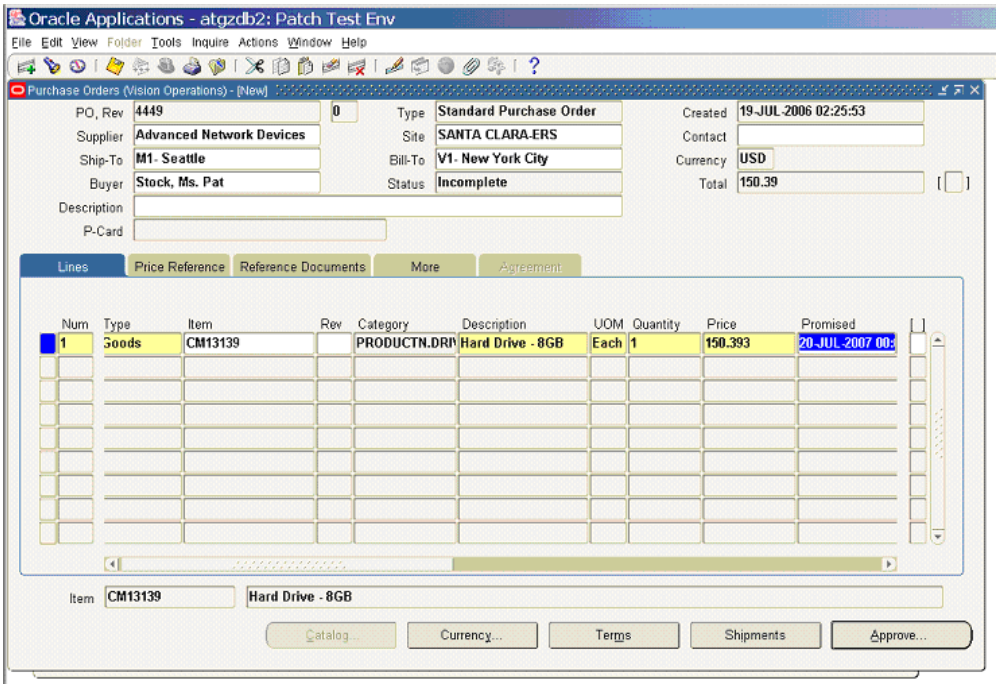
10. On the Lines tab, enter a data row with the following values:

    - Type: Goods
    - Item: CM13139
    - Quantity: 1
    - Description: Hard Drive - 8GB

- Promised: Enter any future date in the format of dd-mmm-yyyy (such as 23-JUN-2009)

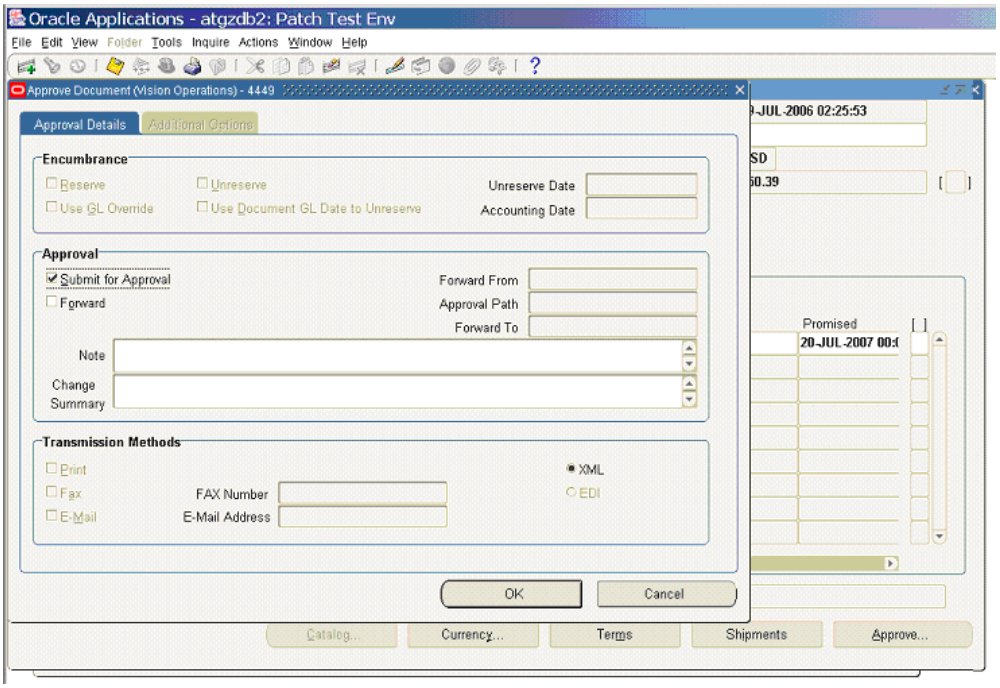
11. Save your purchase order. The status of the purchase order is 'Incomplete'.

Setting Up a Purchase Order



12. Click **Approve**. The Approve Document form appears.

Approving the Purchase Order



Click **OK** to confirm the approval.

> **Note:** Because the trading partner is set up and valid, the transmission method is automatically set to **XML**.

The status of the purchase order is now changed to 'Approved'. For future reference, record the value of the PO, Rev field (for example, the PO number 4449 in this case).

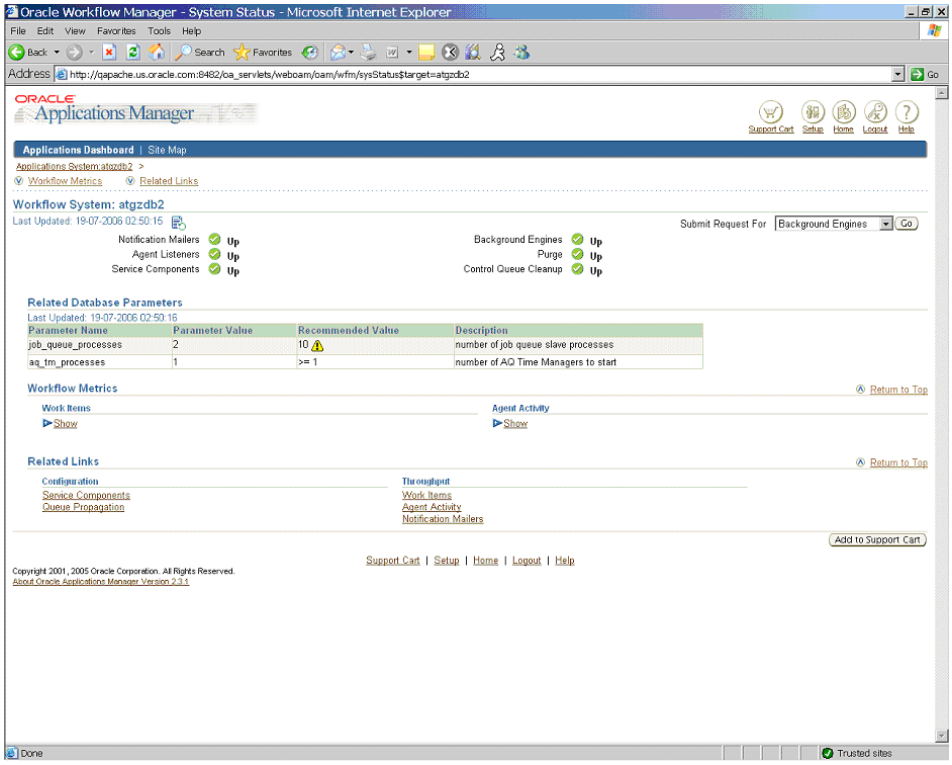
Once the purchase order is approved, the business event `oracle.apps.po.event.xmlpo` is raised.

13. Use the following steps to ensure that the **WF_Deferred Agent Listener** is running on the target database.

    1. Log in to Oracle Applications with the System Administrator responsibility.

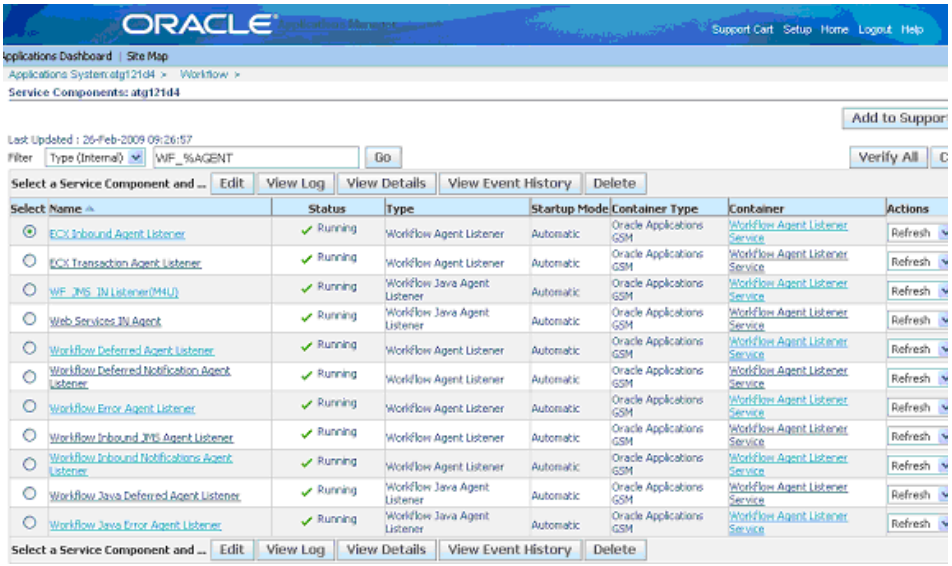
    2. On the Oracle Applications home page, select the Workflow Administrator Web Applications responsibility. Select **Oracle Applications Manager > Workflow Manager** from the menu.

Managing Oracle Applications



3. On the Applications Manager page, click the **Agent Listeners** icon. The Service Components page appears, containing a list of the installed agent listeners.

Reviewing Agent Listener status

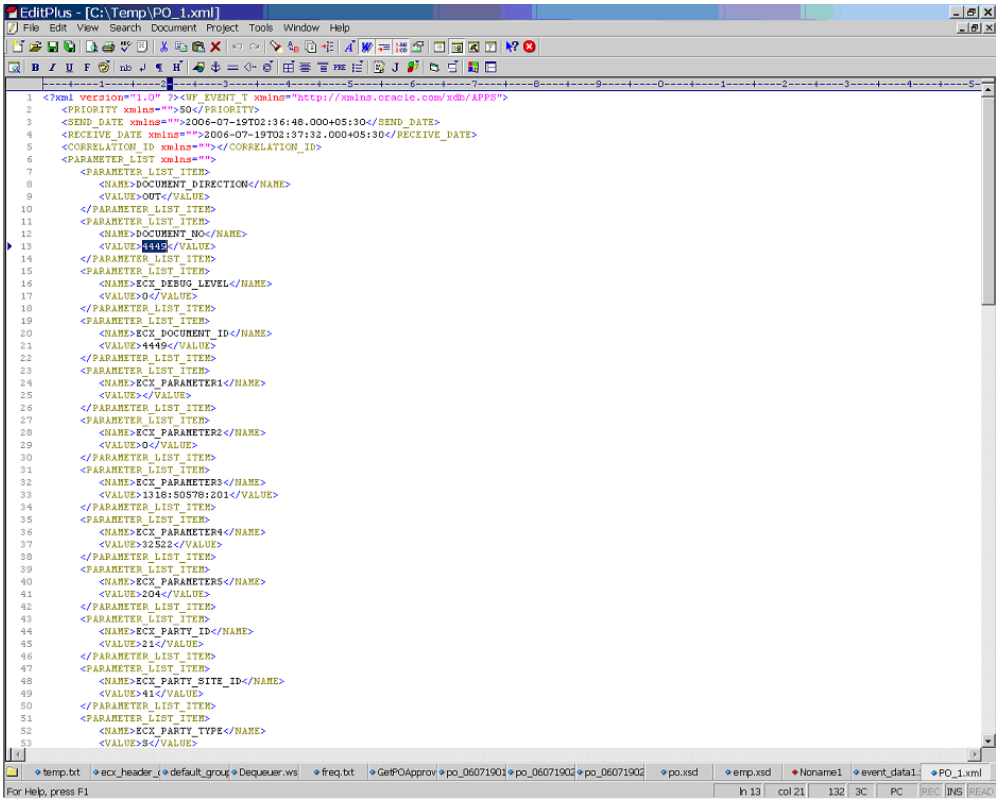


4. Confirm that the **Workflow Deferred Agent Listener** is in `Running` status.

14. At this time, your deployed your BPEL project is listening for `oracle.apps.po.event.xmlpo` business event. Please allow 2-3 minutes for the BPEL process to activate after the event is raised.

15. Go to the directory, for example *outputDir* (typically under `c:\temp`) you specified for the write operation. Open the output file (such as `EventAck%yyMMddHHmmss.xml`), and confirm that the order number is same as that of the approved purchase order.

Confirming the Output Order Number



Troubleshooting and Debugging

If you experience problems with your Business Event System integration, you can take the following troubleshooting steps:

- Confirm that WF_Listener is up and running.
- Ensure that business events are raised only after the BPEL process is deployed.
- If the BPEL process is created on one instance of Oracle Applications and deployed on another instance, ensure the following:
  - WF_BPEL_Q, WF_BPEL_QTab, and WF_BPEL_QAgent should be present on the target database.
  - A custom subscription for the raised business event should be present on the target database.

If you still experience problems with your integration, you can enable debugging.

Enabling Debugging

You can enable debugging for business events using the BPEL Process Manager.

To enable debugging:

1. Log into your BPEL Process Manager domain.
2. Select *yourdomain*.collaxa.cube.ws
3. Select **Debug.**

Debugging information is output to the log file for your domain. To examine the log file in the BPEL Process Manager, navigate to **Home > BPEL Domains >*yourdomain* > Logs.** The log file is ***yourdomain*.log**.

6

Using Concurrent Programs

This chapter covers the following topics:

- Overview of Concurrent Programs
- Design-Time Tasks for Concurrent Programs
- Creating a New BPEL Project
- Adding Partner Links
- Adding Partner Links for File Adapter
- Configuring the Invoke Activities
- Configuring Assign Activities
- Run-Time Tasks for Concurrent Programs
- Deploying the BPEL Process
- Testing the BPEL Process
- Verifying Records in Oracle Applications
- Troubleshooting and Debugging

Overview of Concurrent Programs

A concurrent program is an instance of an execution file. Concurrent programs use a 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. Concurrent manager runs in the background waiting for a concurrent program to be submitted. As soon as a concurrent program is submitted, it is put into an execution queue by concurrent manager. Concurrent manager also manages the concurrent execution of concurrent programs.

Concurrent programs associated with the Open Interface Table move the data from interface table to base tables. While other concurrent programs execute the business logic and application-level processing for Oracle E-Business Suite.

Design-Time Tasks for Concurrent Programs

This section describes how to configure the Adapter for Oracle Applications to use concurrent programs. It describes the tasks required to configure Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper.

BPEL Process Scenario

Take Open Interface: Order Import concurrent program (`OEOIMP`) as an example. This concurrent program allows you to run the Order Import process to retrieve order details from the Order Management open interface tables to create sales orders.

When a request of creating an order is received, order details will be passed and inserted into Open Interface tables (`OE_HEADER_IFACE_ALL` and `OE_LINES_IFACE_ALL`). Once the insertion is completed, the concurrent program is invoked to import order data from Open Interface tables to Order Management base tables.

If the BPEL process is successfully executed after deployment, you should find the order is created in Oracle Applications. The purchase order ID should be the same as the payload input value.

Prerequisites to Configuring Concurrent Programs

Populating Application Context Header Variables

You need to populate certain variables in the BPEL PM in order to provide context information for Oracle Applications. The context information is required for a transaction in order for an Oracle Applications user that has sufficient privileges to run the program.

The context is set taking into account the values passed for the header properties including *Username*, *Responsibility*, *Responsibility Application*, *Security Group*, and *NLS Language*. If the values for the new header properties *Responsibility Application*, *Security Group*, and *NLS Language* are not passed, context information will be determined based on *Username* and *Responsibility*.

The default Username is `SYSADMIN`, the default Responsibility is `SYSTEM ADMINISTRATOR`, the default Security Group Key is `Standard`, and the default NLS Language is `US`.

You can change the default values specified in the generated WSDL. 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 values. The context information can be specified by configuring an Invoke activity.

For more information about applications context, see Supporting for Normalized Message Properties, page 3-8.

BPEL Process Creation Flow

Based on the process scenario, the following design-time tasks are explained in this chapter:

1. Create a new BPEL project., page 6-3
2. Add partner links., page 6-7
3. Add a partner link for File Adapter., page 6-30
4. Configure Invoke activities., page 6-41
5. Configure Assign activities., page 6-48

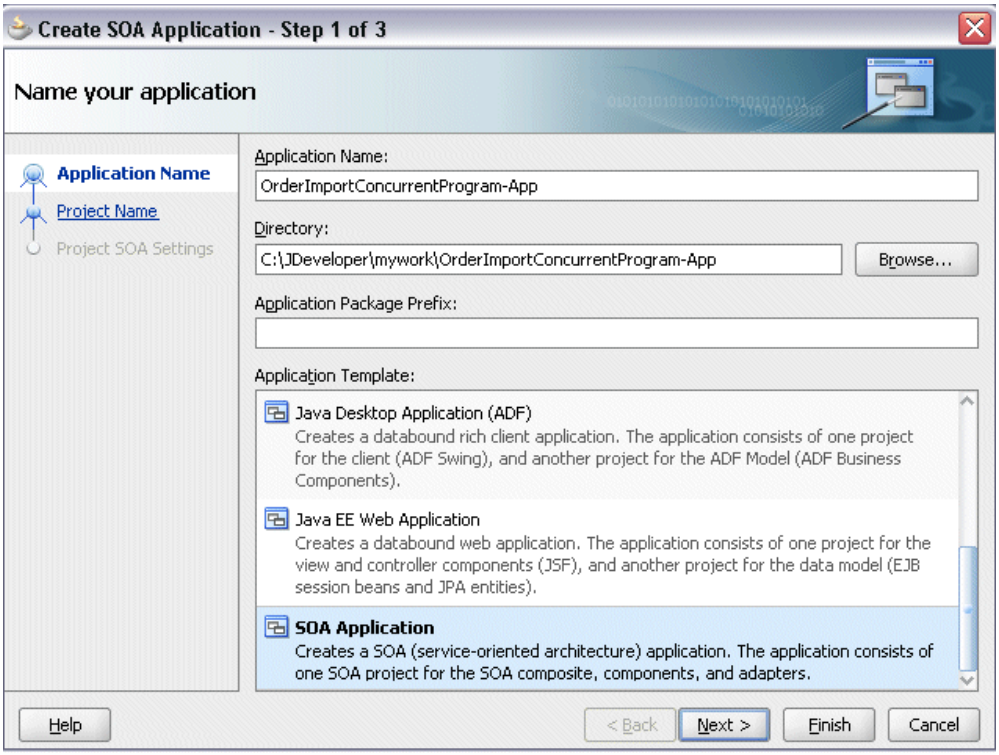
Creating a New BPEL Project

To create a new BPEL project:

1. Open JDeveloper BPEL Designer. Click **New Application** in the Application Navigator.

   The Create SOA Application - Name your application page is displayed.

The Create SOA Application - Name your application Page

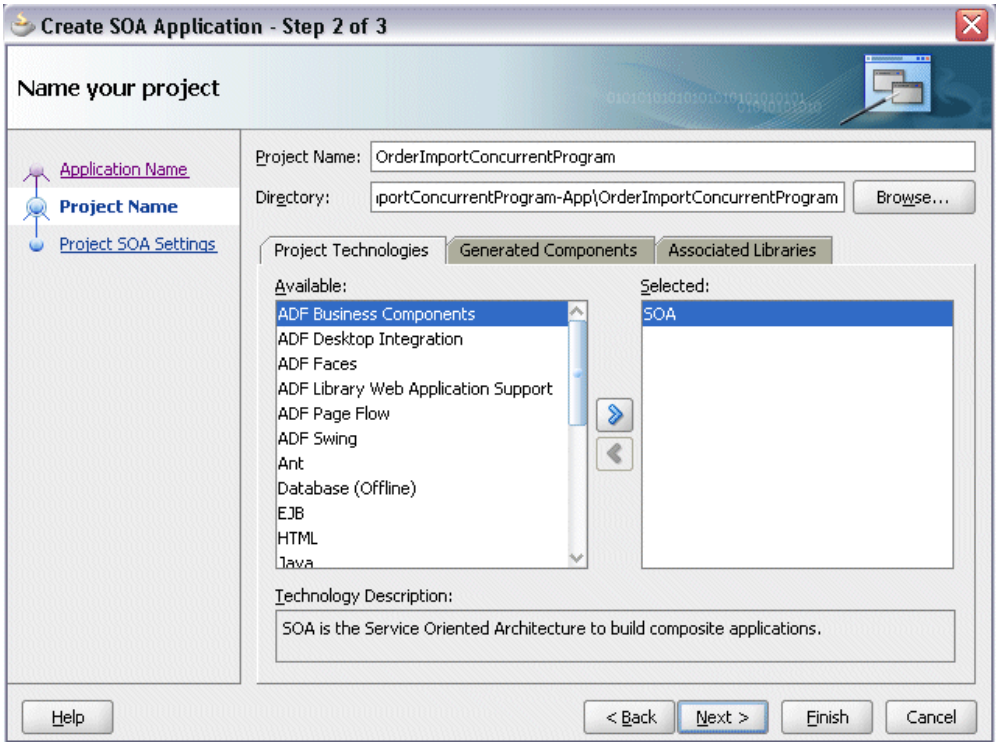


2. Enter an appropriate name for the application in the **Application Name** field and select **SOA Application** from the Application Template list.

   Click **Next**. The Create SOA Application - Name your project page is displayed.

3. Enter an appropriate name for the project in the **Project Name** field. For example, `OrderImportConcurrentProgram`.

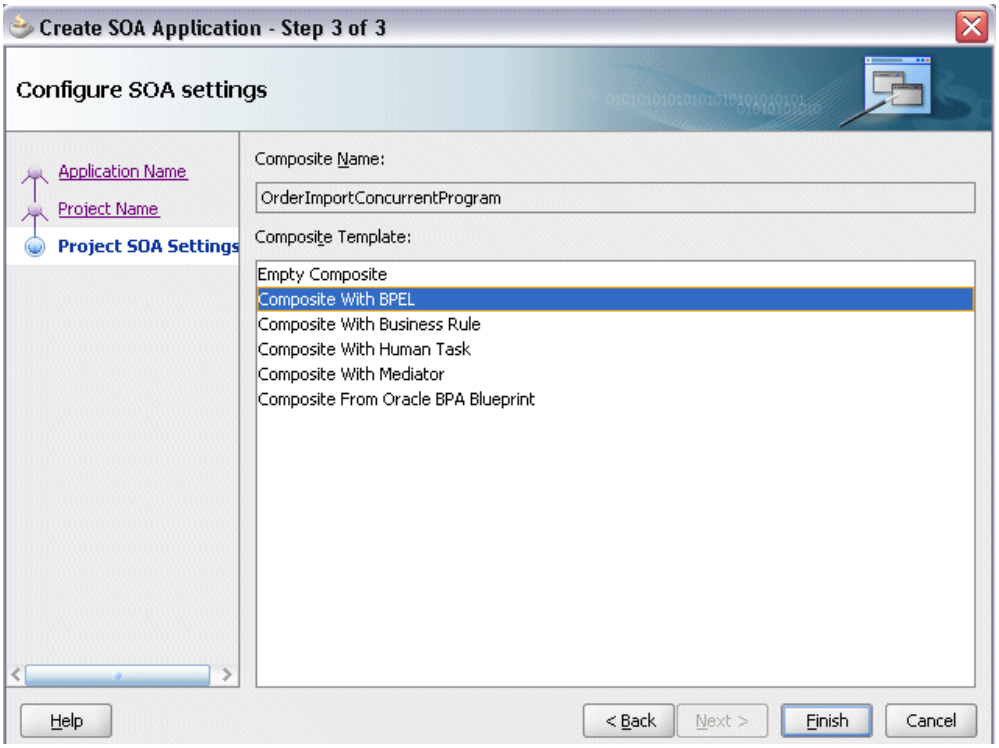
The Create SOA Application - Name your project Page



4. In the Project Technologies tab, ensure that **SOA** is selected from the Available technology list to the Selected technology list.

   Click **Next**. The Create SOA Application - Configure SOA settings page is displayed.

The Create SOA Application - Configure SOA settings Page

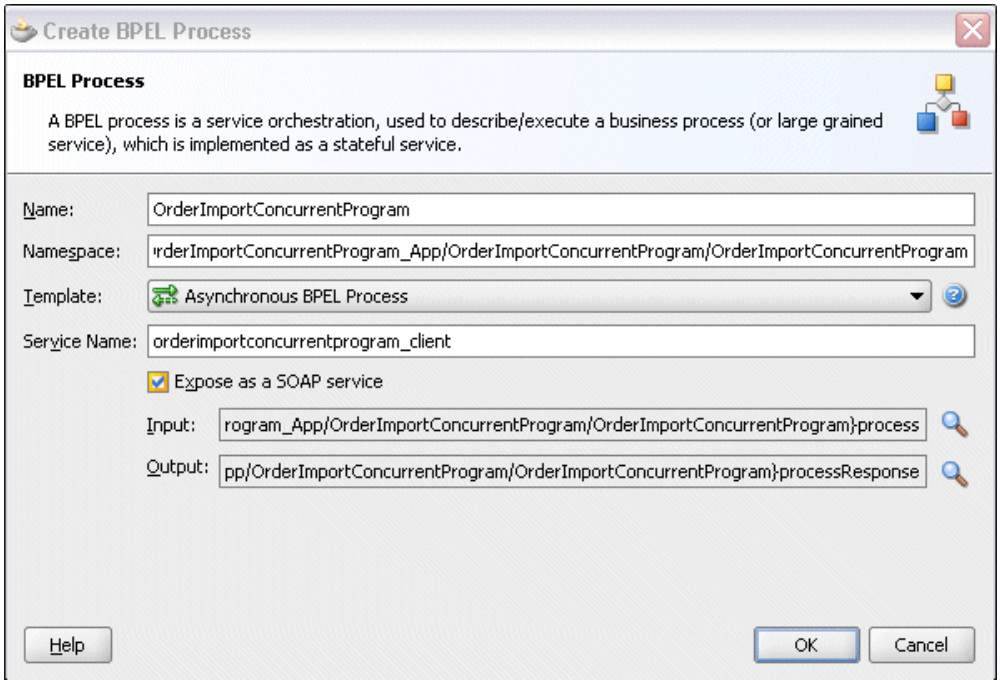


5. In the **BPEL Process Name** field, enter a descriptive name. For example, `OrderImportConcurrentProgram`.

6. Select **Composite With BPEL** from the Composite Template list, and then click **Finish**. You have created a new application, and an SOA project. This automatically creates an SOA composite.

   The Create BPEL Process page is displayed.

The Create BPEL Process Page



7. Enter an appropriate name for the BEPL process in the **Name** field. For example, `OrderImportConcurrentProgram`.

   Select **Asynchronous BPEL Process** in the **Template** field. Click **OK**.

   An asynchronous BPEL process is created with the Receive and Reply activities. The required source files including bpel and wsdl, using the name you specified (for example, `OrderImportConcurrentProgram.bpel` and `OrderImportConcurrentProgram.wsdl`) and `composite.xml` are also generated.

Adding Partner Links

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.

Based on the BPEL process scenario discussed earlier, the following two partner links need to be configured:

- Add the first partner link `InsertOrder` to insert order details into selected Open Interface tables

- Add the second partner link `ImportOrderCP` to import purchase order data from the Open Interface tables to Order Management base tables

To add the first partner link:

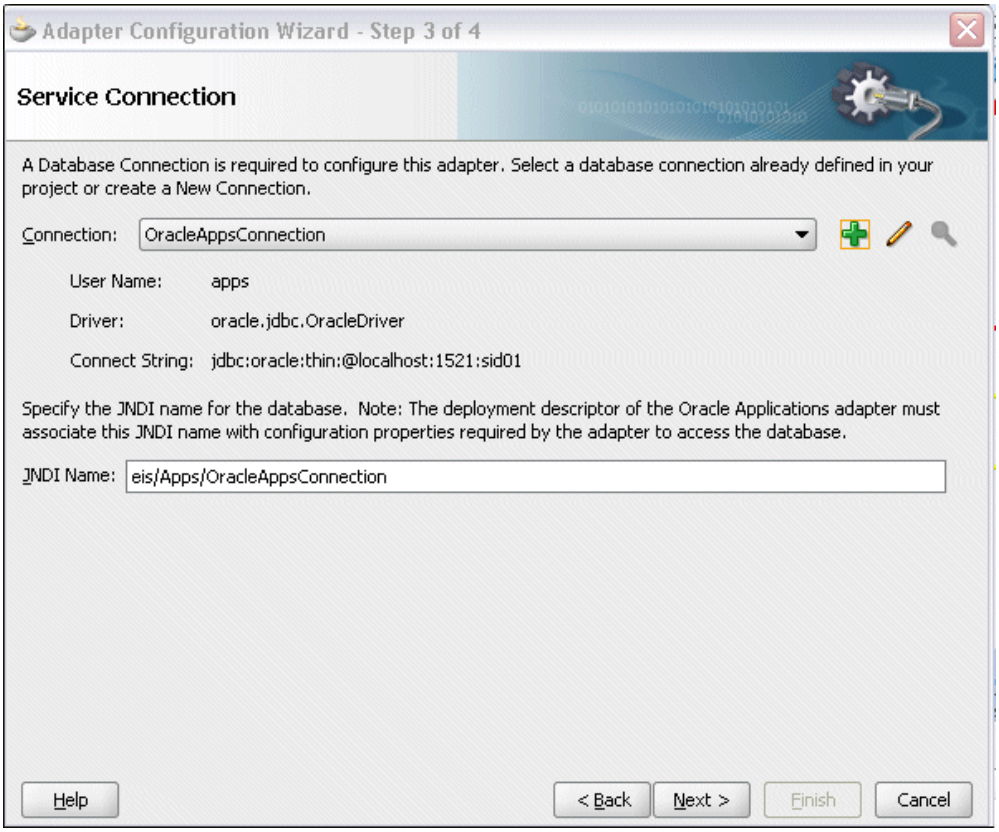
1. Click **BPEL Services** in the Component palette.

   Drag and drop **Oracle Applications** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears. Click **Next**.

2. Enter a service name in the **Service Name** field. For example, `InsertOrder`.

   Click **Next**. The Service Connection dialog appears.

Specifying a Database Service Connection



3. You can perform either one of the following options for your database connection:

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

   - You can create a new database connection by clicking the **Create a New Database Connection** icon.

How to define a new database connection, see Create a New Database Connection, page 4-13.

- You can select an existing database connection that you have configured earlier from the **Connection** drop-down list.

  The Service Connection page will be displayed with the selected connection information. The JNDI (Java Naming and Directory Interface) name corresponding to the database connection appears automatically in the **Database Server JNDI Name** field. Alternatively, you can specify a JNDI name.

  > **Note:** When you specify a JNDI name, the deployment descriptor of the Oracle Applications adapter must associate this JNDI name with configuration properties required by the adapter to access the database.

  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.

  > **Note:** For more information about JNDI concepts, refer to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

4. Click **Next** in the Service Connection dialog box. You can add a concurrent program by browsing through the list of concurrent programs available in Oracle Applications.

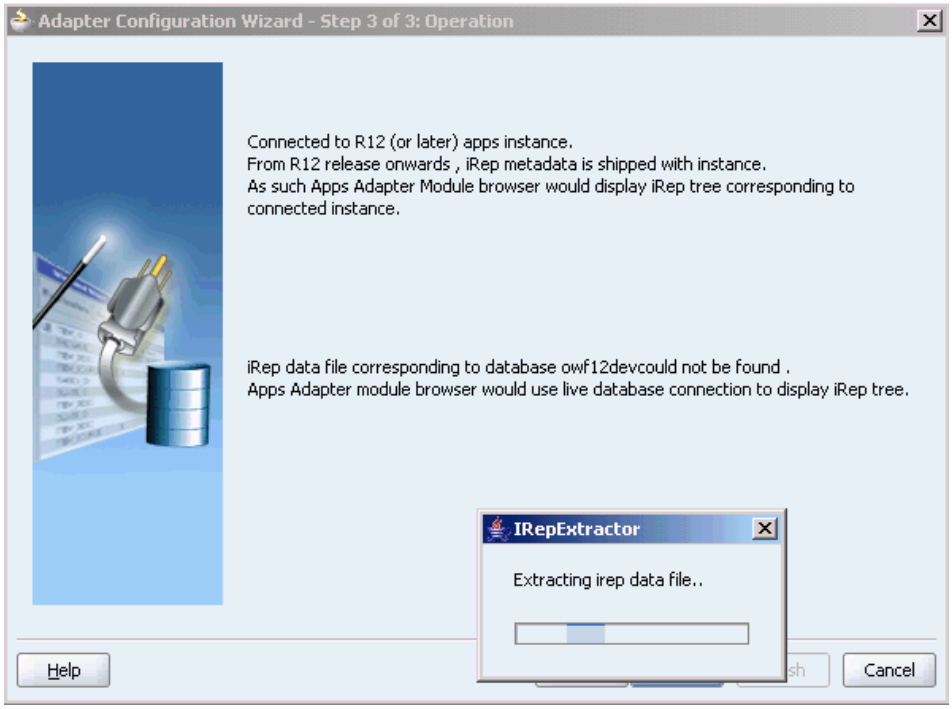
   For Oracle E-Business Suite Release 12:

   If you are connecting to Oracle E-Business Suite Release 12, then the **IREP File not present** dialog box appears indicating that Adapter could not find the Oracle Integration Repository data file corresponding to the database you are connecting in your workspace. Absence of the data file would make browsing or searching of Integration Repository tree considerably slow. You can choose to extract the data file and create a local copy of the Integration Repository data file. Once it is created successfully, Adapter will pick it up automatically next time and retrieve data from your local Integration Repository.

   You can select one of the following options:

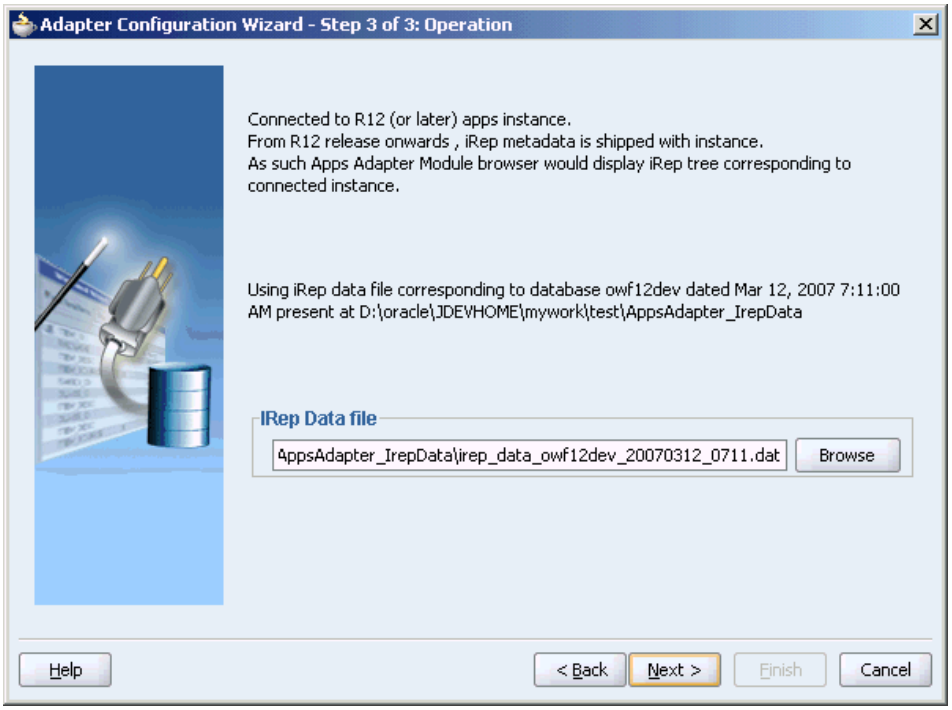
   - Click **Yes** to extract the Integration Repository data file.

Extracting Integration Repository Data File



After the system successfully creates a local copy of the Integration Repository data file, next time when you connect to the database, you will find the **IRep Data File** field appears in the Operation dialog box indicating where your local copy exists with the creation date and time as part of the file name.

Using the Local Integration Repository Data File



- Click **No** to query the Integration Repository data file from the live database you are connecting to display the Integration Repository tree.

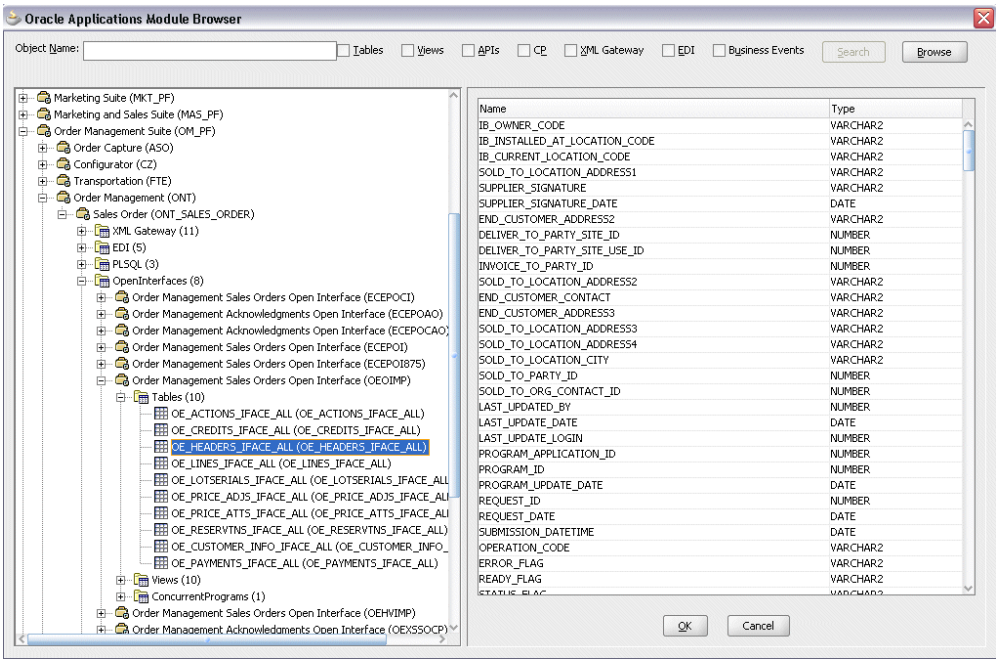
  Note: It is highly recommended that you create a local copy of the Integration Repository data file so that Adapter will query the data next time from the local copy in your workspace to enhance the performance.

For Oracle E-Business Suite pre-Release 11.5.10:

If you are connecting to a pre-11.5.10 Oracle Applications instance, you must select the interface type in the Adapter Configuration Wizard. Select **Tables/Views/APIs/Concurrent Programs** to proceed.

5. Click **Get Object** to open the Oracle Applications Module Browser.

Selecting a Concurrent Program from the Module Browser



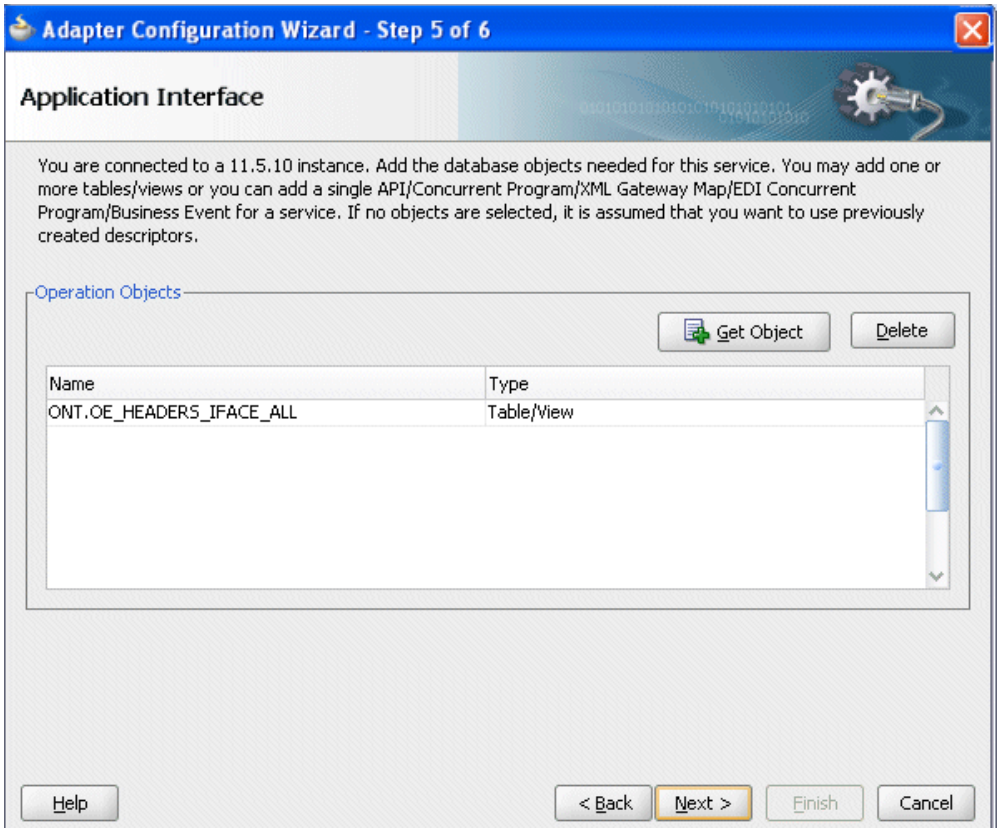
Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. For example, Applications Technology or Order Management Suite are product families in Oracle Applications. The product families contain the individual products. For example, Order Management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

6. Navigate to *Order Management Suite (OM_PF) > Order Management (ONT) > Sales Order (ONT_SALES_ORDER) > OpenInterfaces >Order Management Sales Orders Open Interface (OEOIMP) > Tables* to select `OE_HEADERS_IFACE_ALL`.

   Click **OK**. The Application Interface page appears with the selected open table.

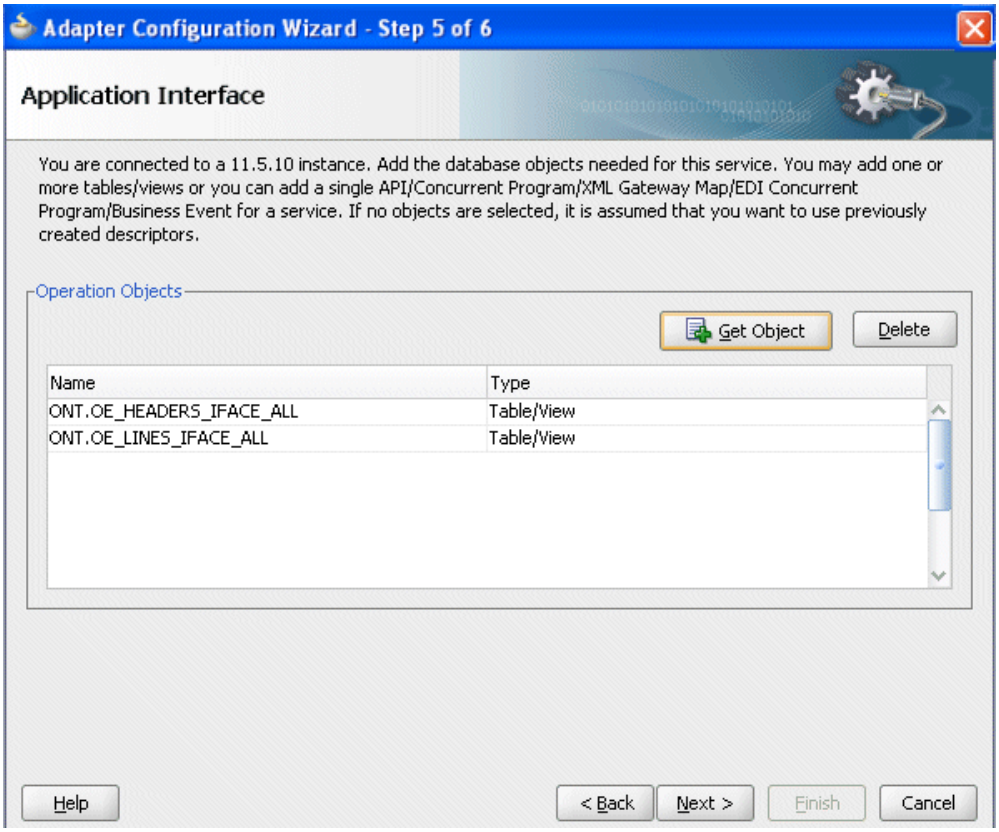
Adapter Configuration Wizard - Application Interface Page



7. Click **Get Object** to open the Oracle Applications Module Browser again to select another open interface table `OE_LINES_IFACE_ALL` using the same navigation path *Order Management Suite (OM_PF) > Order Management (ONT) > Sales Order (ONT_SALES_ORDER) > OpenInterfaces >Order Management Sales Orders Open Interface (OEOIMP) > Tables*.

   Click **OK**. The Application Interface page appears with the two selected tables.

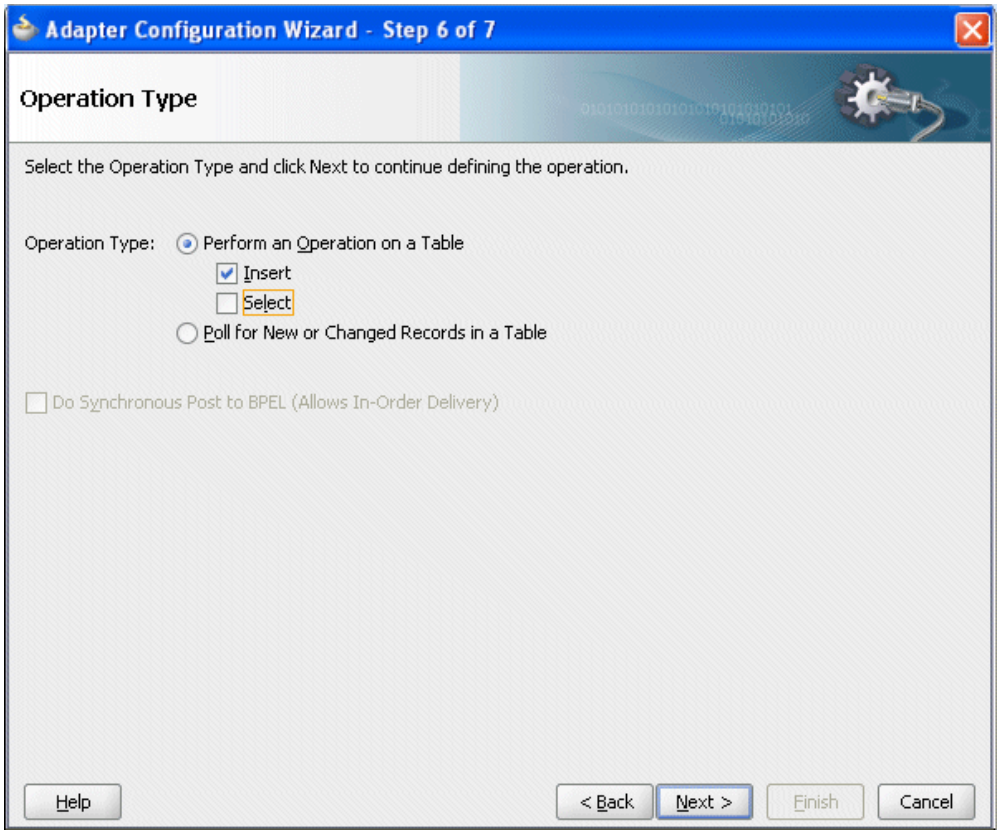
Adapter Configuration Wizard - Application Interface Page



Click **OK**.

8. Click **Next**. The Operation Type page is displayed.

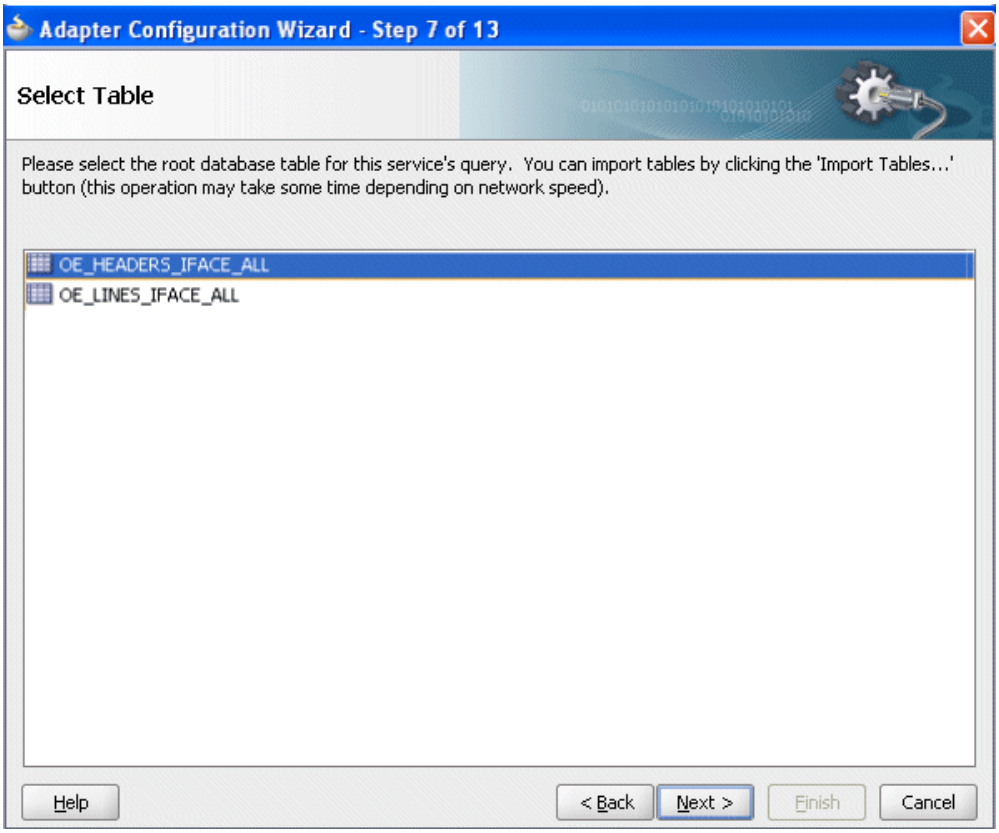
Adapter Configuration Wizard - Operation Type Page



Select the **Perform an Operation on a Table** radio button and then choose the **Insert** check box. Ensure that the **Select** check box is deselected.

9. Click **Next**. The Select Table page is displayed.

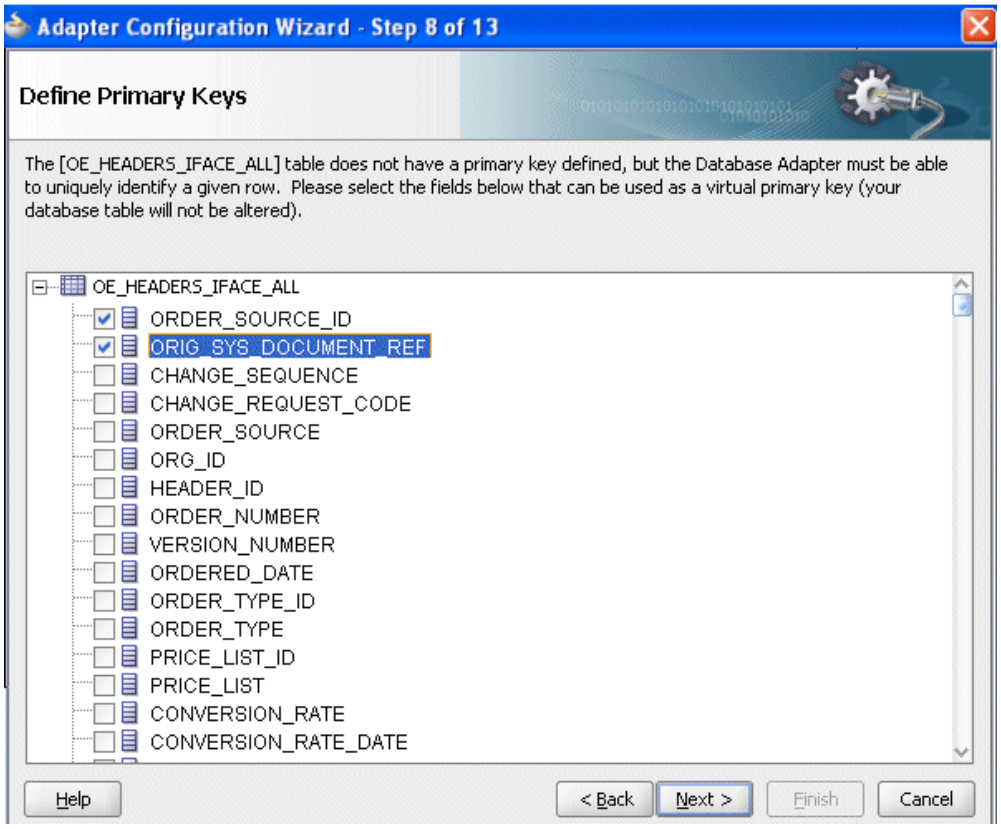
Adapter Configuration Wizard - Select Table Page



Select OE_HEADERS_IFACE_ALL as the root database table for this service's query.

10. Click **Next**. The Define Primary Keys page is displayed.

Adapter Configuration Wizard - Define Primary Keys Page for OE_HEADERS_IFACE_ALL



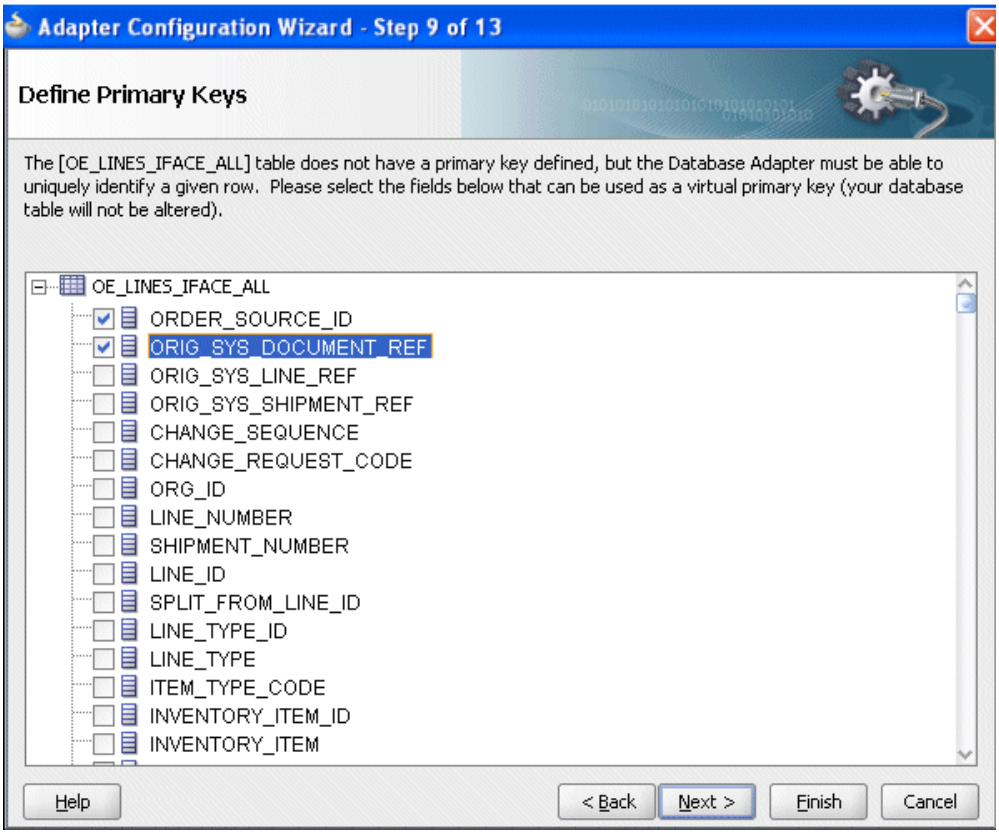
Select the following primary keys for the OE_HEADERS_IFACE_ALL table:

- ORDER_SOURCE_ID
- ORIG_SYS_DOCUMENT_REF

Click **Next**.

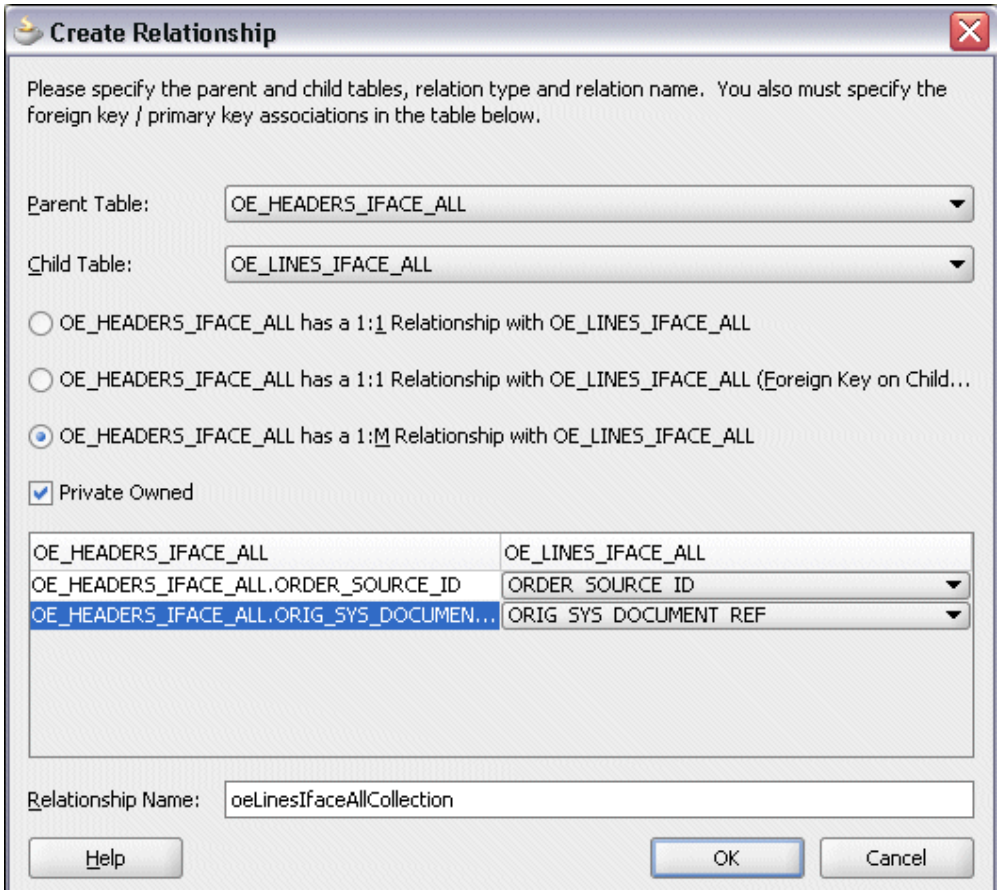
Select the same primary keys for the OE_LINES_IFACE_ALL table.

Adapter Configuration Wizard - Define Primary Keys Page for OE_LINES_IFACE_ALL



11. Click **Next**. The Relationships page appears. Click **Create** to open the Create Relationship dialog.

Defining Relationships



Enter the following information to define the relationship between the header and the detail table:

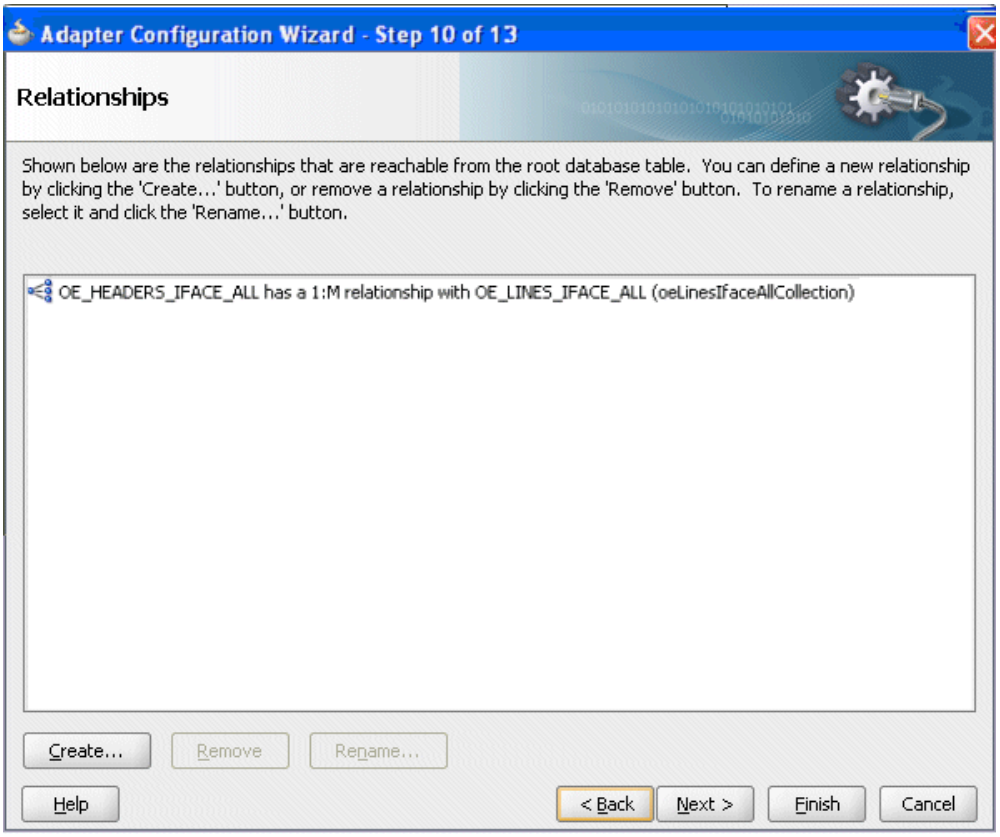
- Select the OE_HEADERS_IFACE_ALL as the parent table and OE_LINES_IFACE_ALL as the child table.
- Select the mapping type: OE_HEADERS_IFACE_ALL has a 1:M Relationship with OE_LINES_IFACE_ALL

> **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.

- Select the **Private Owned** check box.
- Associate the foreign key fields with the primary key fields:
  - OE_HEADERS_IFACE_ALL.ORDER_SOURCE_ID: ORDER_SOURCE_ID
  - OE_HEADERS_IFACE_ALL.ORIG_SYS_DOCUMENT_REF: ORIG_SYS_DOCUMENT_REF
- The Relationship Name field is populated automatically by default. You can optionally specify a new name for the relationship you are creating.

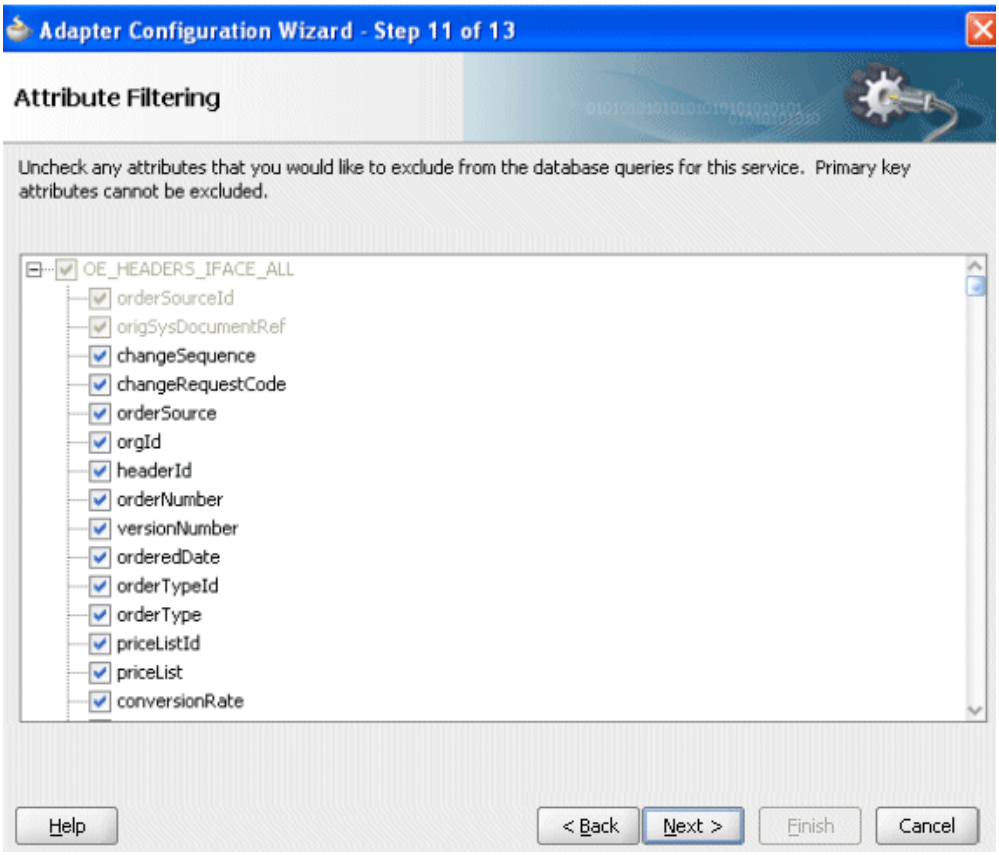
Click **OK**. The newly created relationship information appears in the Relationships page.

Adapter Configuration Wizard - Relationships Page



12. Click **Next**. The Attribute Filtering page appears. Leave default selections unchanged.

Adapter Configuration Wizard - Attribute Filtering Page

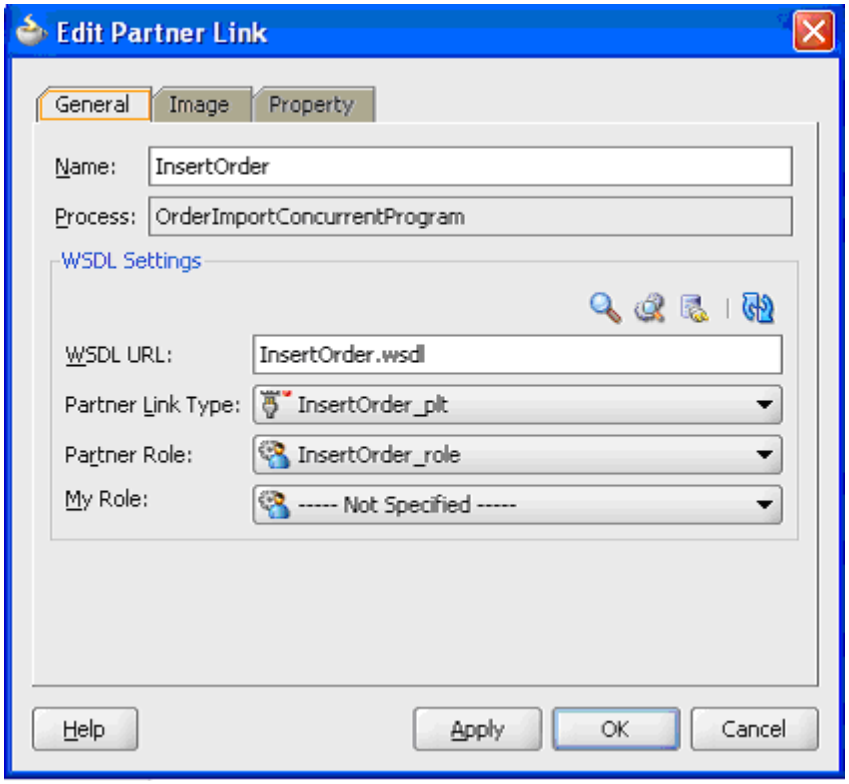


Click **Next**. The Advanced Options page appears.

Click **Next**.

13. Click **Finish**. The wizard generates the WSDL file corresponding to the selected interface. This WSDL file is now available for the partner link.

Completing the Partner Link Configuration



14. Click **Apply** and then **OK**. The partner link is created with the required WSDL settings.

To add the second partner link:

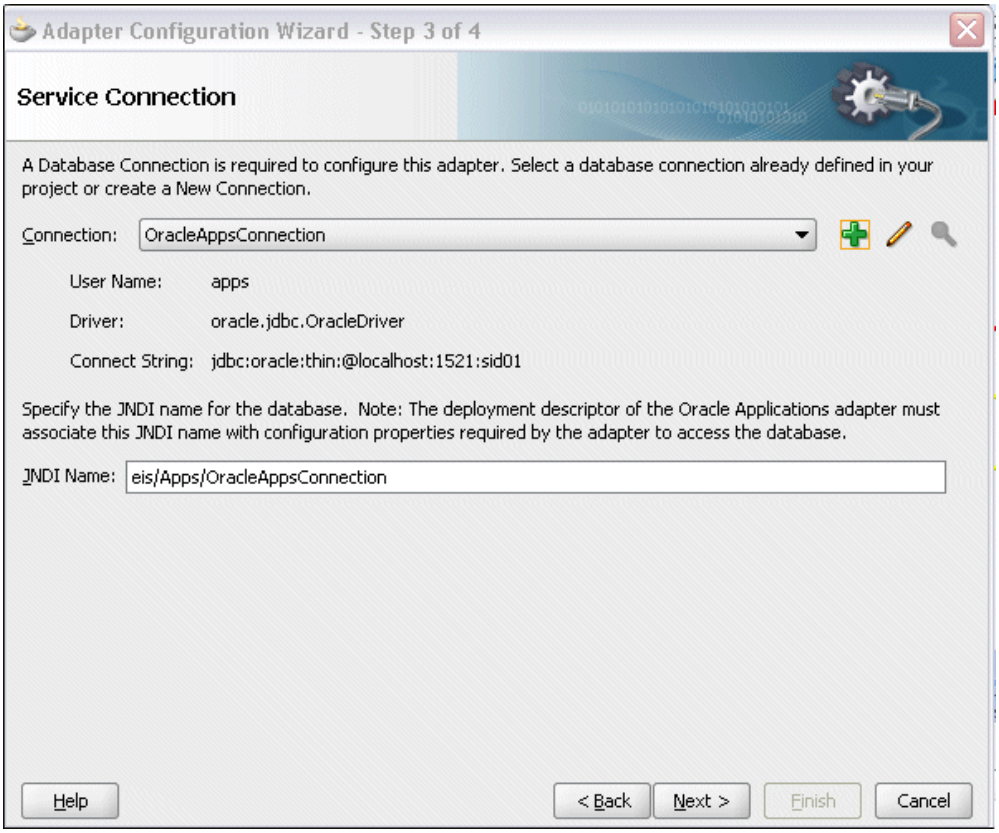
1. Click **BPEL Services** in the Component palette.

   Drag and drop **Oracle Applications** from the BPEL Services list into the right Partner Link swim lane of the process diagram after the first partner link. The Adapter Configuration Wizard Welcome page appears. Click **Next**.

2. Enter a service name in the **Service Name** field. For example, `ImportOrderCP`.

   Click **Next**. The Service Connection dialog appears.

Specifying a Database Service Connection



3. You can perform either one of the following options for your database connection:

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

   - You can create a new database connection by clicking the **Create a New Database Connection** icon.

     How to define a new database connection, see Create a New Database Connection, page 4-13.

   - You can select an existing database connection that you have configured earlier from the **Connection** drop-down list.

     The Service Connection page will be displayed with the selected connection information. The JNDI (Java Naming and Directory Interface) name corresponding to the database connection appears automatically in the **Database Server JNDI Name** field. Alternatively, you can specify a JNDI name.

Note: When you specify a JNDI name, the deployment descriptor of the Oracle Applications adapter must associate this JNDI name with configuration properties required by the adapter to access the database.

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.

Note: For more information about JNDI concepts, refer to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

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

   Click **Next**.

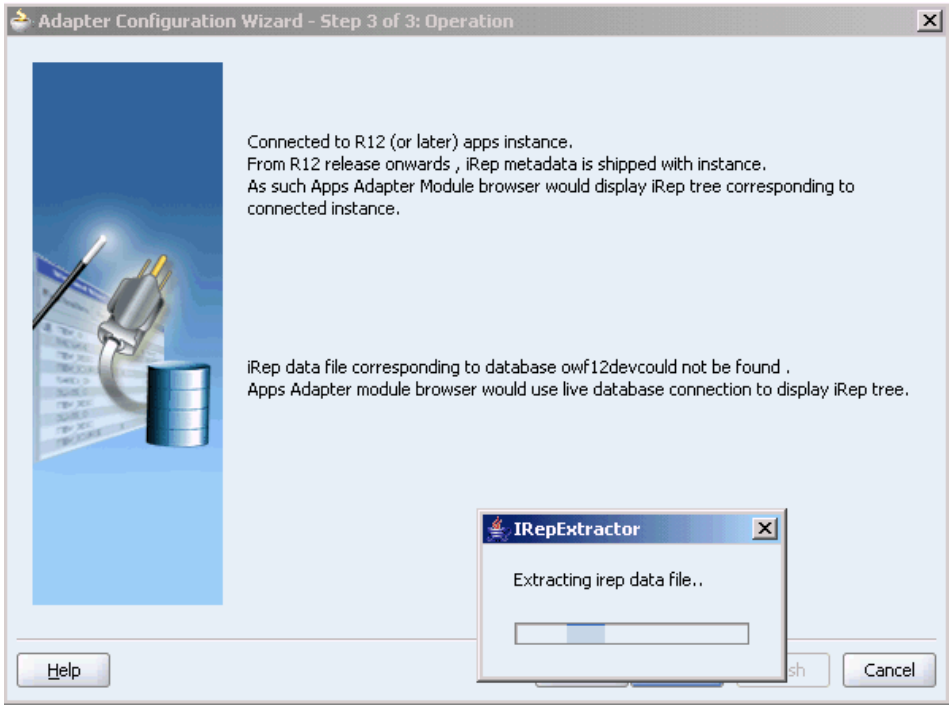
   For Oracle E-Business Suite Release 12:

   If you are connecting to Oracle E-Business Suite Release 12, then the **IREP File not present** dialog appears indicating that Adapter could not find the Oracle Integration Repository data file corresponding to the database you are connecting to Oracle Applications in your workspace. Absence of the data file would make browsing or searching of Integration Repository tree considerably slow. You can choose to extract the data file and create a local copy of the Integration Repository data file. Once it is created successfully, Adapter will pick it up automatically next time and retrieve data from your local Integration Repository.

   You can select one of the following options:

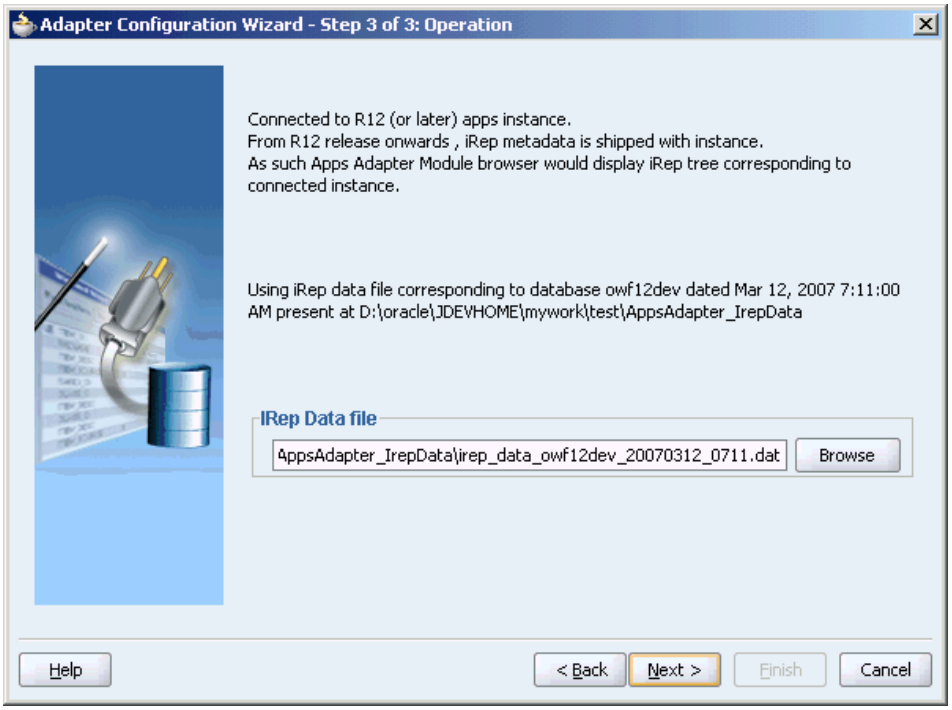
   - Click **Yes** to extract the Integration Repository data file.

Extracting Integration Repository Data File



After the system successfully creates a local copy of the Integration Repository data file, next time when you connect to the database, you will find the **IRep Data File** field appears in the Operation dialog indicating where your local copy exists with the creation date and time as part of the file name.

Using the Local Integration Repository Data File



- Click **No** to query the Integration Repository data file from the live database you are connecting to display the Integration Repository tree.

  Note: It is highly recommended that you create a local copy of the Integration Repository data file so that Adapter will query the data next time from the local copy in your workspace to enhance the performance.

  Click **Next** in the Operation page to open the Oracle Applications Module Browser.

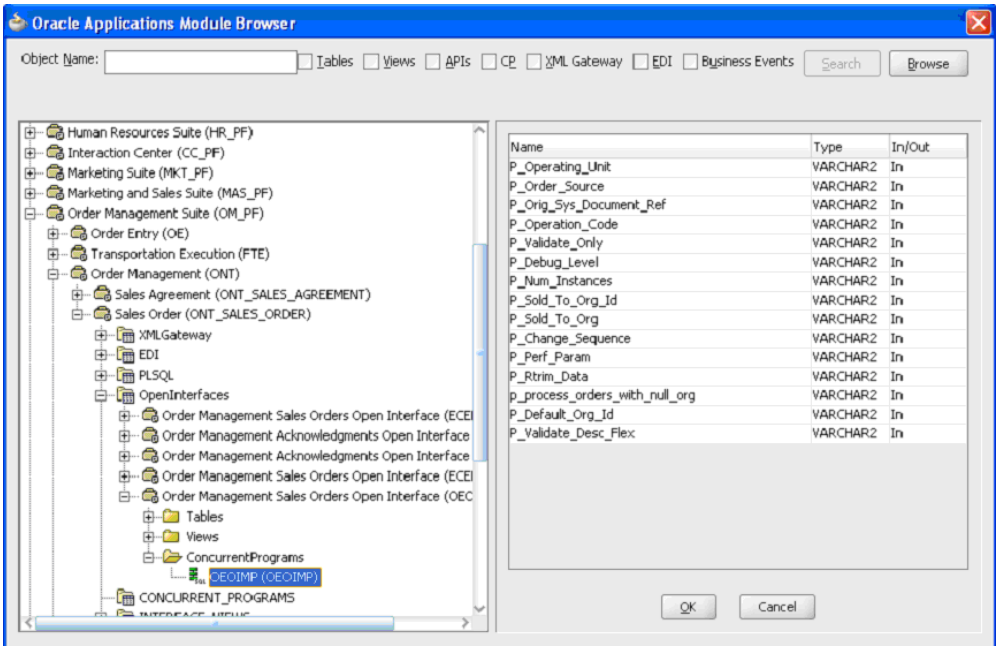
For Oracle E-Business Suite pre-Release 11.5.10:

If you are connecting to a pre-11.5.10 Oracle Applications instance, you must select the interface type in the Adapter Configuration Wizard. Select **Tables/Views/APIs/Concurrent Programs** to proceed.

Click **Get Object** in the Application Interface dialog to open the Oracle Applications Module Browser.

5. The Oracle Applications Module Browser combines interface data from Oracle Integration Repository with information about the additional interfaces supported by Oracle Application Adapter, organized in a tree hierarchy.

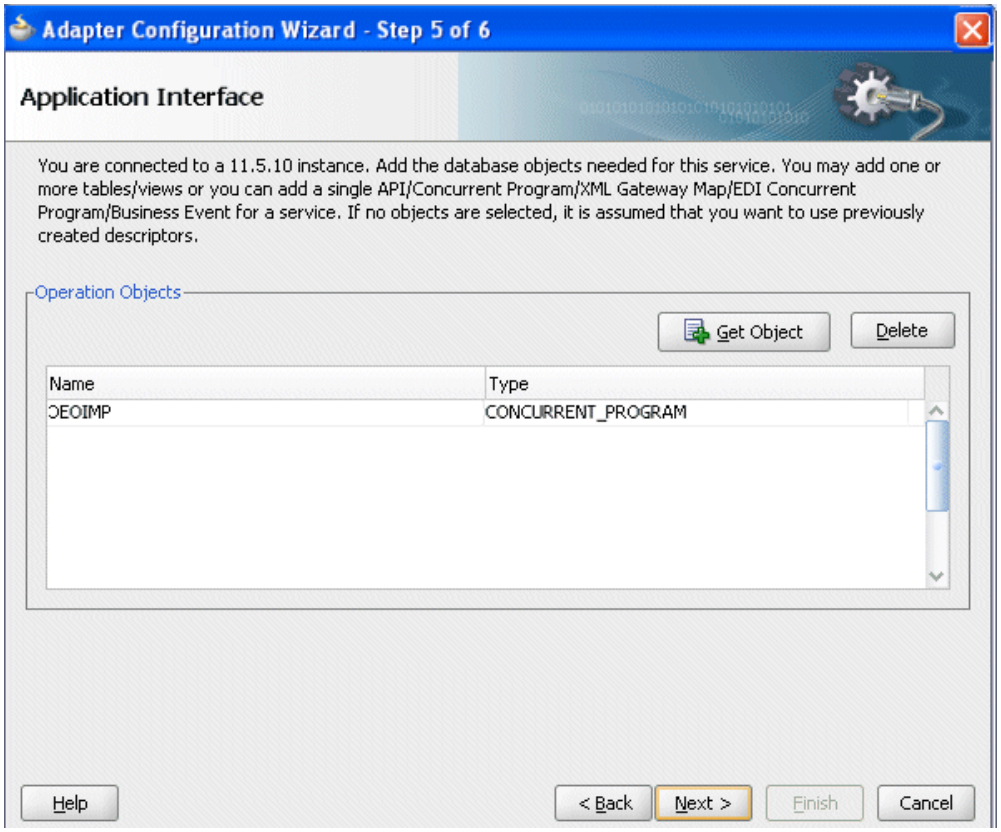
Specify a concurrent program from The Oracle Applications Module Browser



Navigate to *Order Management Suite (OM_PF) >Order Management (ONT) > Sales Order (ONT_SALES_ORDER) > OpenInterfaces > Order Management Sales Orders Open Interface > ConcurrentPrograms* to select a concurrent program `OEOIMP (OEOIMP)`.

6. Click **OK**. The Application Interface page appears.

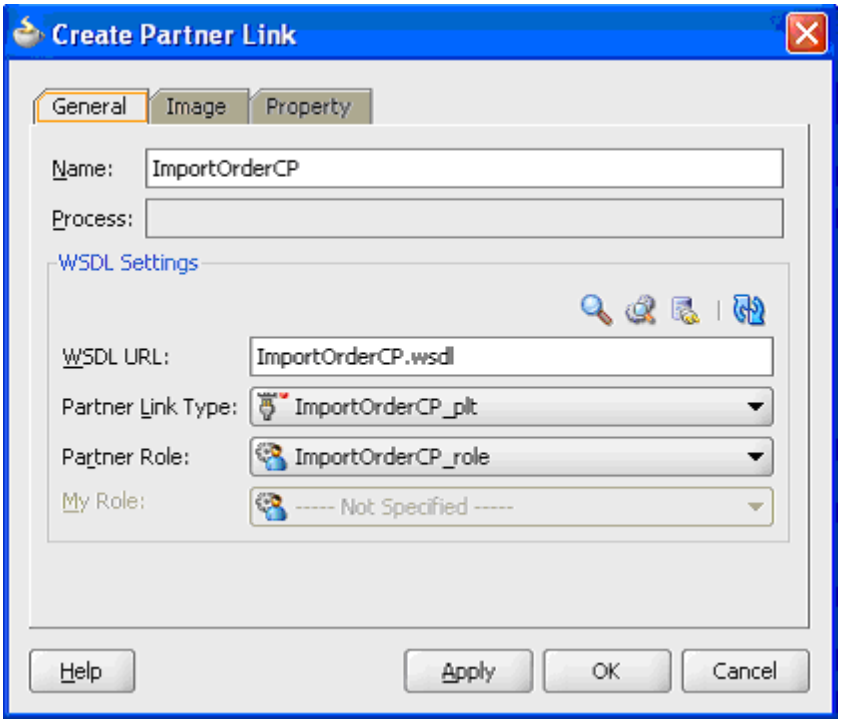
Adapter Configuration Wizard - Application Interface Page



7. Click **Next**, then click **Finish** to complete the process of configuring 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.

Create Partner Link



8. Click **Apply** and then **OK**. The partner link is created with the required WSDL settings.

Adding Partner Links for File Adapter

Use this step to configure a BPEL process by synchronously reading payload from an input file to obtain the order details and concurrent program details.

Configure the following two partner links:

1. To obtain order details from an input file through Synchronous Read operation.
2. To obtain concurrent program details from an input file through Synchronous Read operation.

To add the first Partner Link for File Adapter to read order details:

1. In JDeveloper BPEL Designer, click **BPEL Services** in the Component palette.

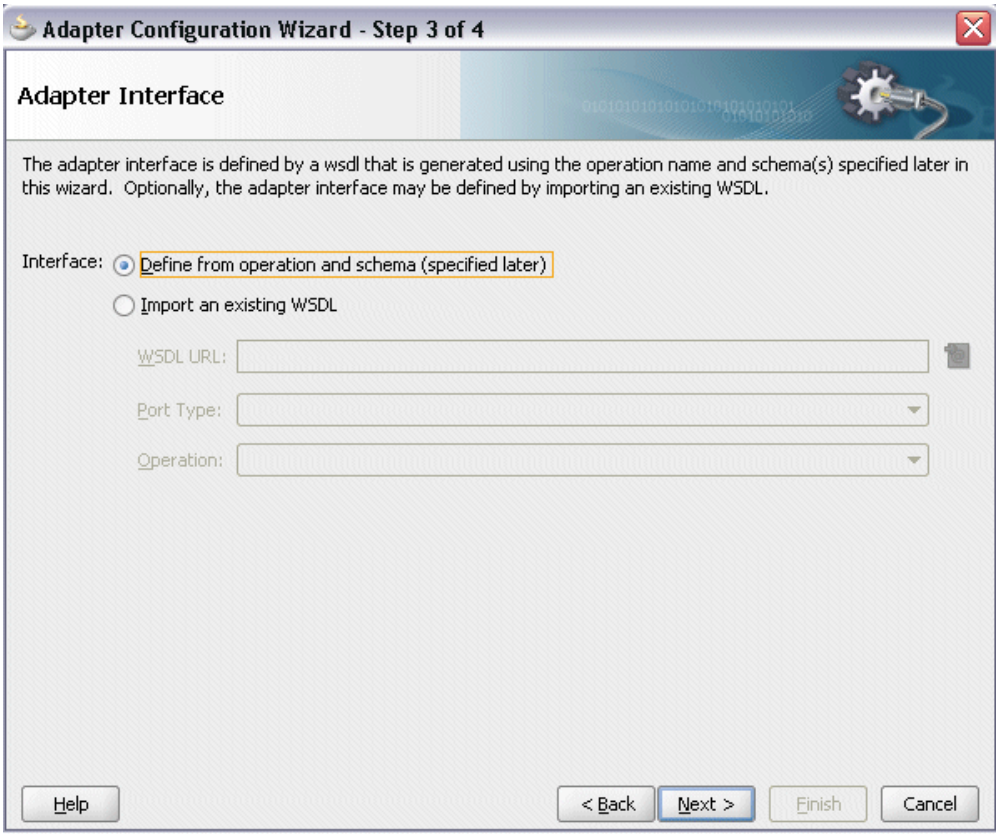
   Drag and drop **File Adapter** from the BPEL Services list into the right Partner Link swim lane before the first partner link `InsertOrder`. The Adapter Configuration Wizard Welcome page appears.

   Click **Next**.

2. In the Service Name dialog, enter a name for the File Adapter service, for example, `getOrderDetails`.

3. Click **Next**. The Adapter Interface page appears.

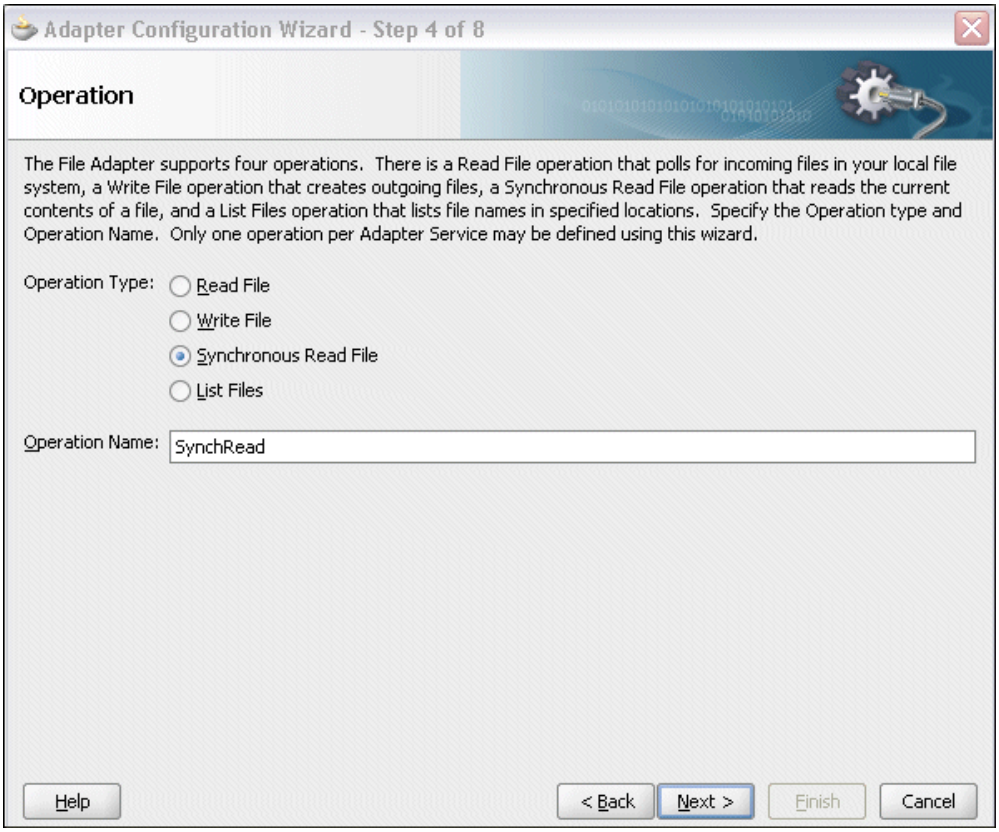
Specifying the Adapter Interface



Select the **Define from operation and schema (specified later)** radio button and click **Next**.

4. In the Operation page, specify the operation type. For example, select the **Synchronous Read File** radio button. This automatically populates the **Operation Name** field.

Specifying the Operation

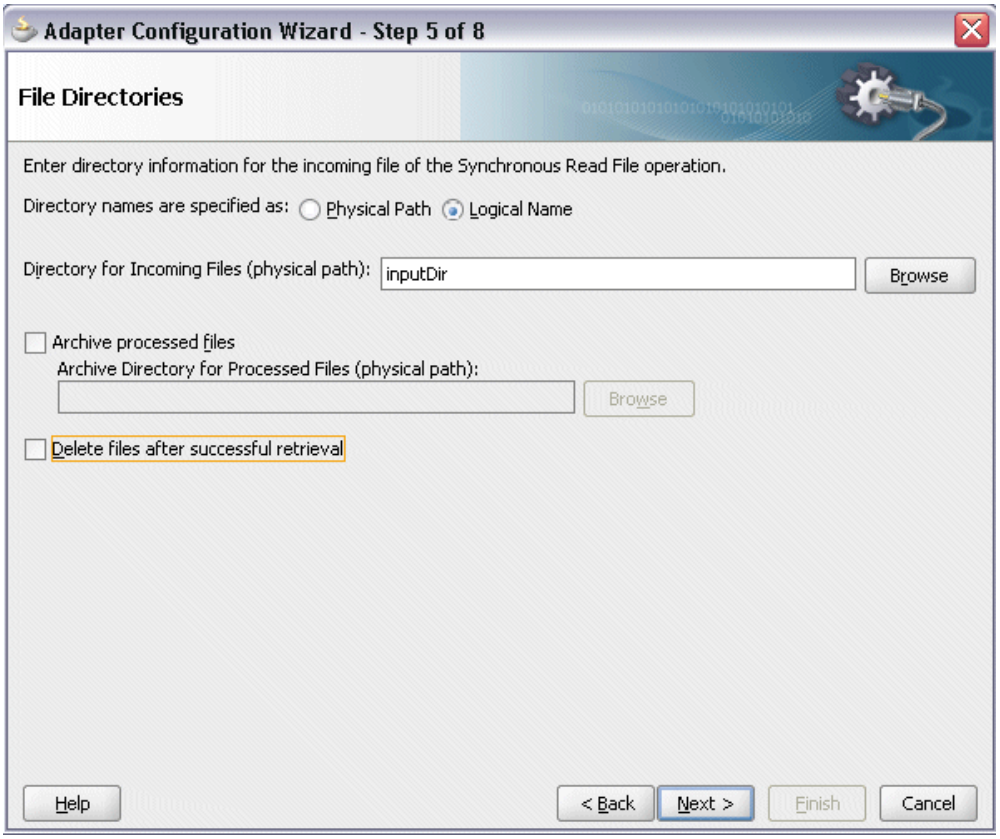


Click **Next** to access the File Directories page.

5. Select the **Logical Name** radio button and specify directory for incoming files, such as `inputDir`.

   Ensure the **Delete files after successful retrieval** check box is not selected.

Configuring the Input File



Click **Next** to open the File Name page.

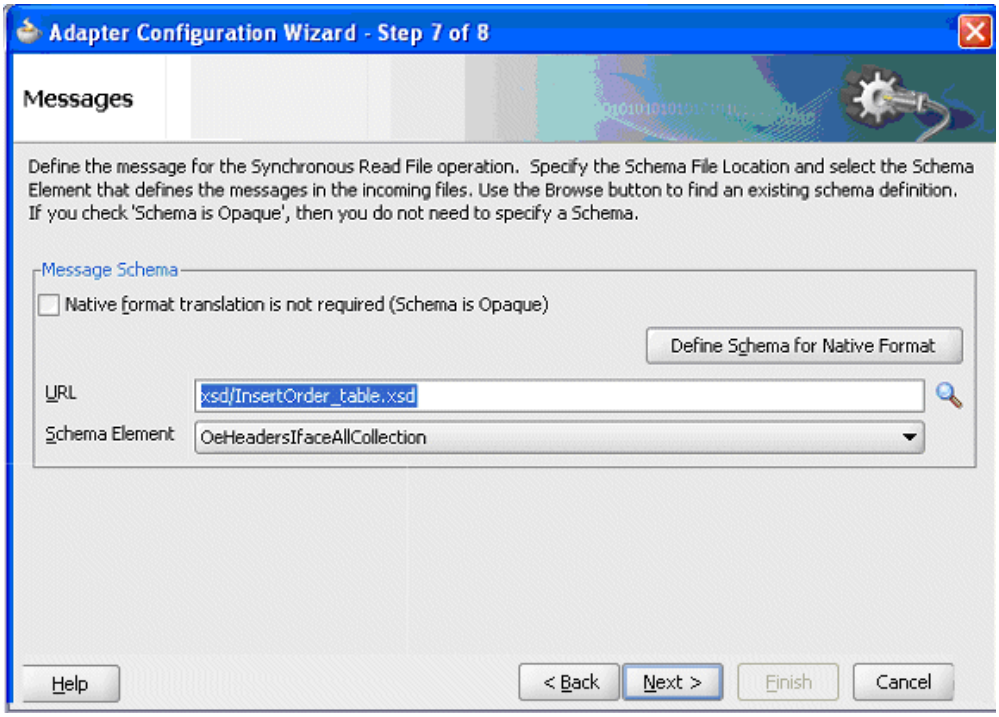
6. Enter the name of the file for the synchronous read file operation. For example, enter `order_data.xml`. Click **Next**. The Messages page appears.

7. Select the 'browse for schema file' icon next to the URL field to open the Type Chooser.

   Click Type Explorer and select *Project Schema Files* > *InsertOrder_table.xsd* > *OeHeadersIfaceAllCollection*. Click **OK**.

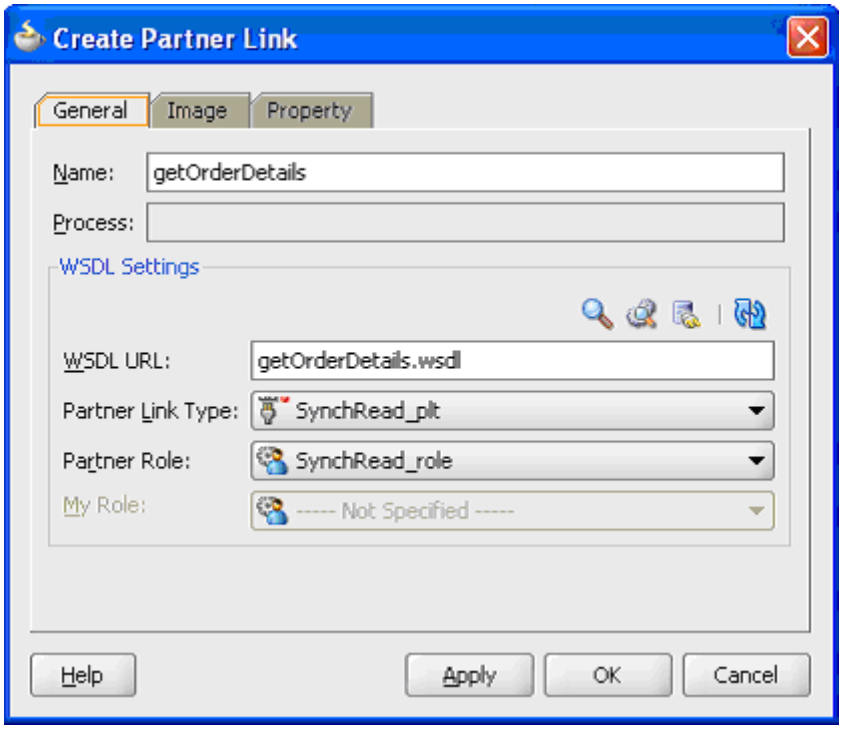
   The selected schema information will be automatically populated in the URL and Schema Element fields.

Specifying Message Schema



8. Click **Next** and then **Finish**. The wizard generates the WSDL file corresponding to the partner link. The main Create Partner Link dialog box appears, specifying the new WSDL file `getOrderDetails.wsdl`.

Completing the Partner Link Configuration



Click **Apply** and **OK** to complete the configuration and create the partner link with the required WSDL settings for the File Adapter service.

The `getOrderDetails` Partner Link appears in the BPEL process diagram.

To add the second Partner Link for File Adapter to read concurrent program details:

1. In JDeveloper BPEL Designer, click **BPEL Services** in the Component palette.

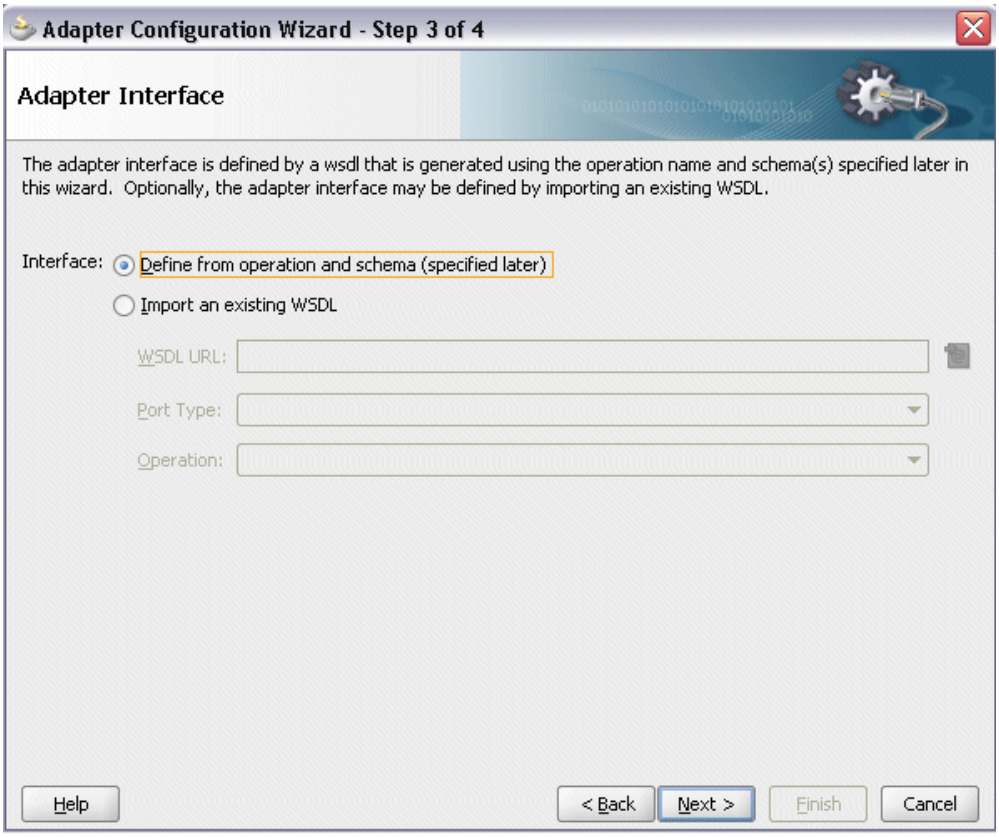
   Drag and drop **File Adapter** from the BPEL Services list into the right Partner Link swim lane after the first File Adapter partner link `getOrderDetails`. The Adapter Configuration Wizard Welcome page appears.

   Click **Next**.

2. In the Service Name page, enter a name for the File Adapter service, such as `getCPDetails`.

3. Click **Next**. The Adapter Interface page appears.

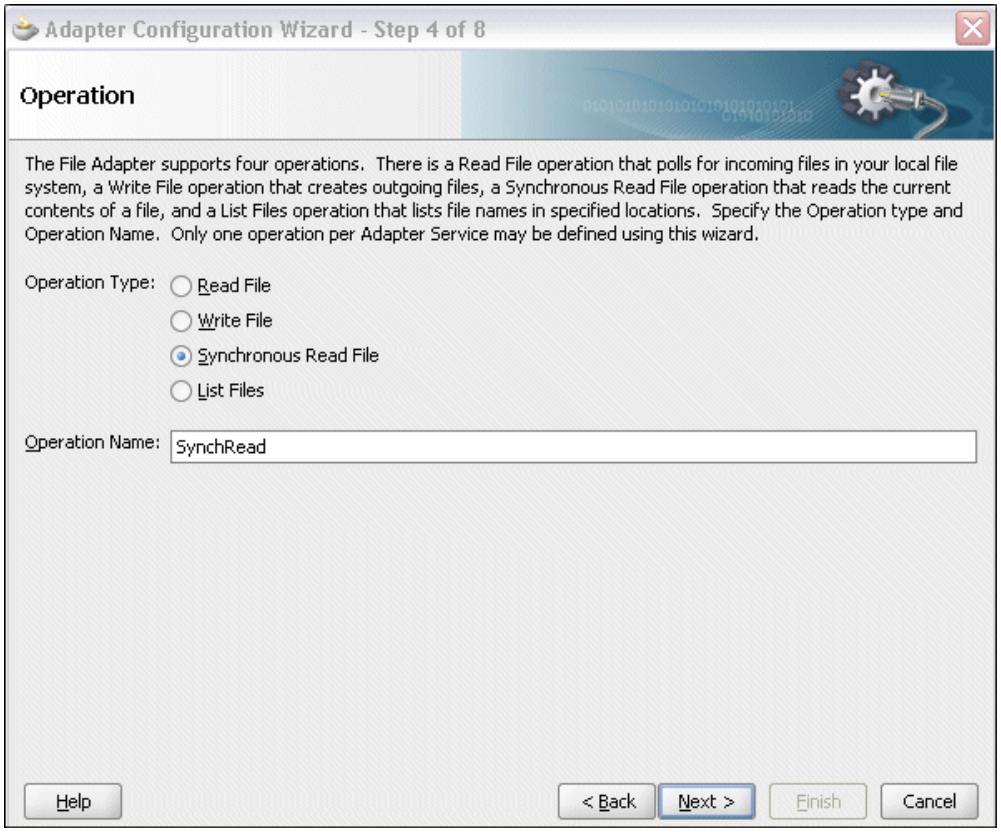
Specifying the Adapter Interface



Select the **Define from operation and schema (specified later)** radio button and click **Next**.

4. In the Operation page, specify the operation type. For example, select the **Synchronous Read File** radio button. This automatically populates the **Operation Name** field.

Specifying the Operation

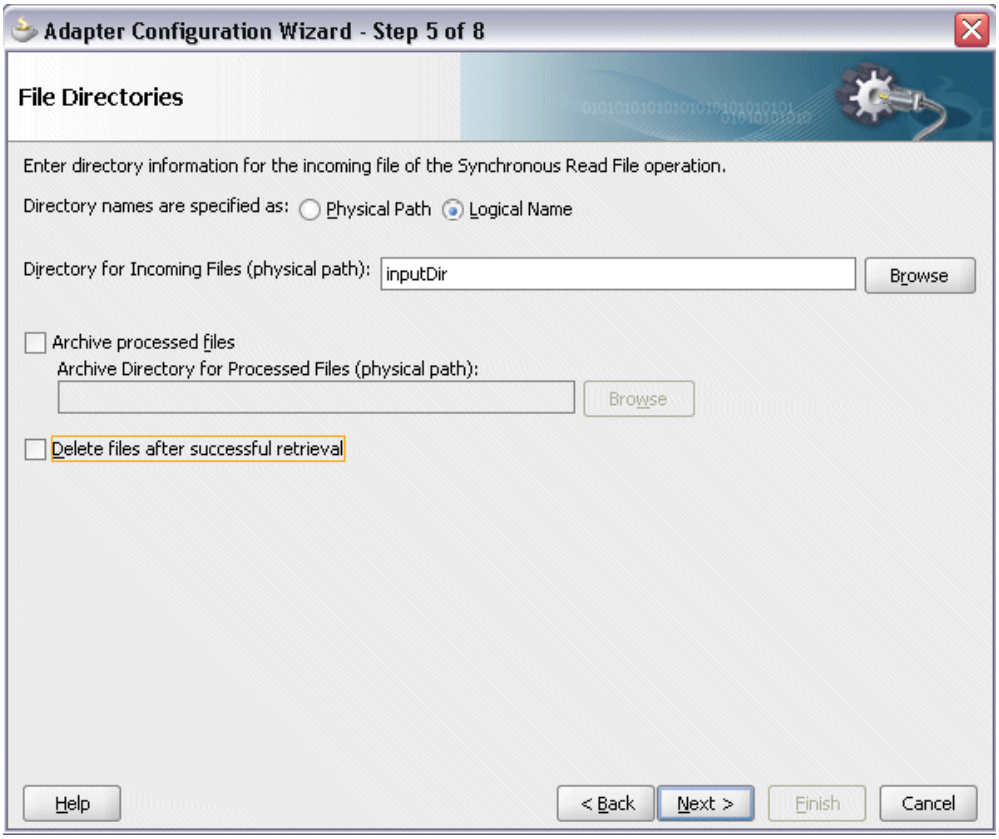


Click **Next** to access the File Directories page.

5. Select the **Logical Name** radio button and specify directory for incoming files, such as `inputDir`.

   Ensure the **Delete files after successful retrieval** check box is not selected.

Configuring the Input File

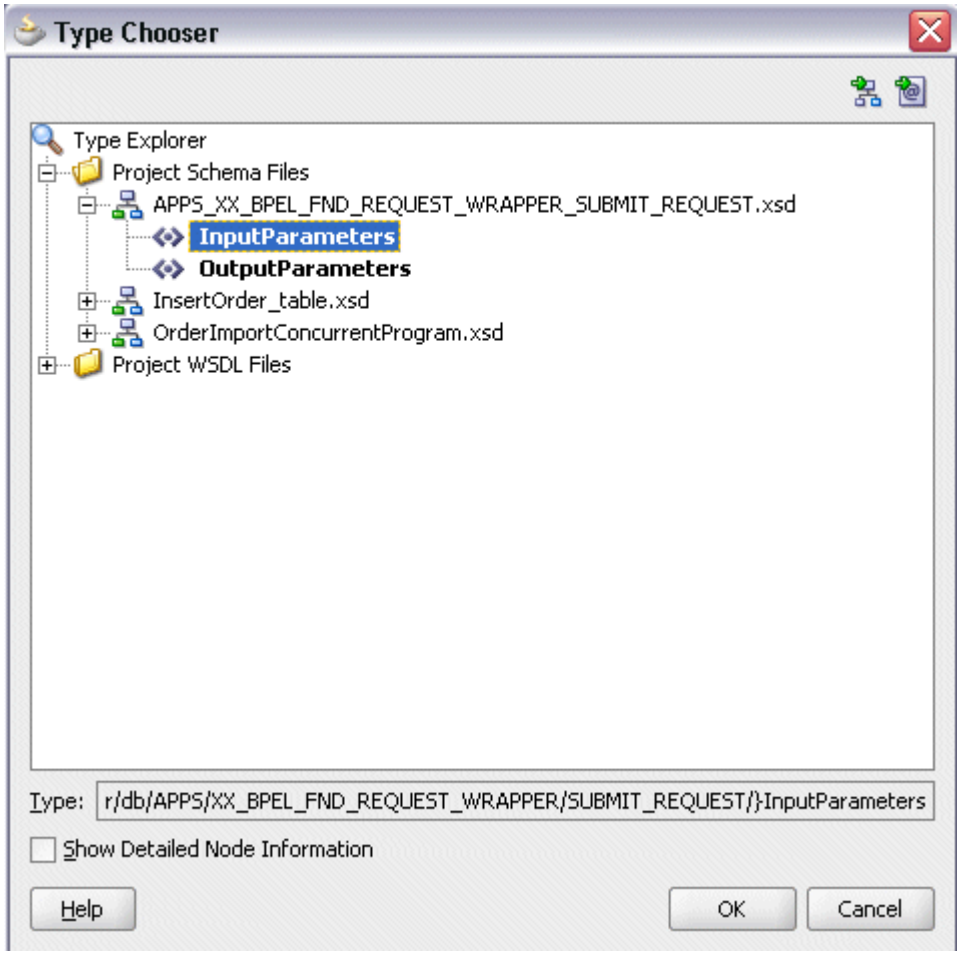


Click **Next** to open the File Name page.

6. Enter the name of the file for the synchronous read file operation. For example, enter `cp_data.xml`. Click **Next** to open the Messages page.

7. Select the 'browse for schema file' icon next to the URL field to open the Type Chooser.

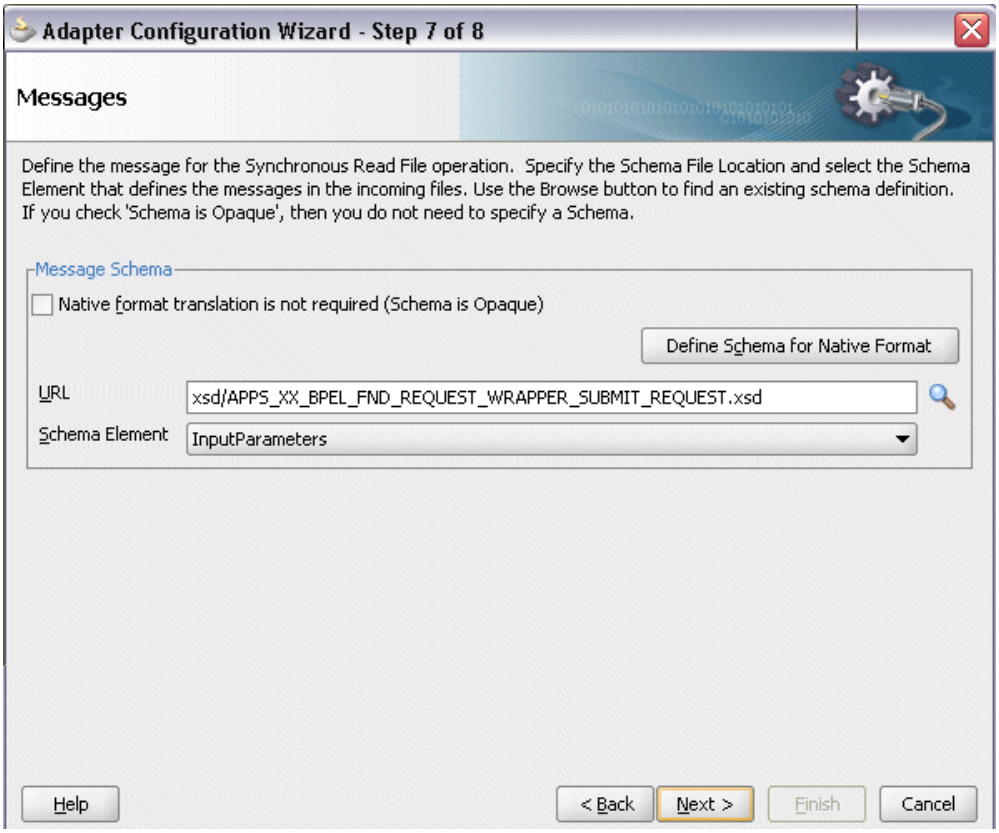
Selecting Schema URL and Element



Click Type Explorer and select *Project Schema Files* > *APPS_XX_BPEL_FND_REQUEST_WRAPPER_SUBMIT_REQUEST.xsd* > *InputParameters*. Click **OK**.

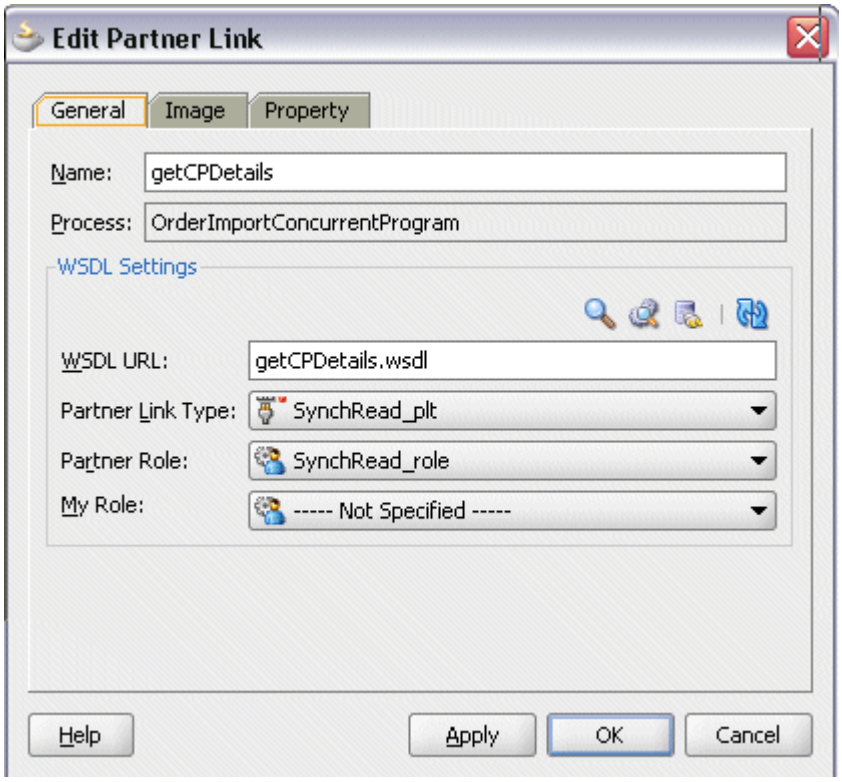
The selected schema information will be automatically populated in the URL and Schema Element fields.

Specifying Message Schema



8. Click **Next** and then **Finish**. The wizard generates the WSDL file corresponding to the partner link. The main Create Partner Link dialog box appears, specifying the new WSDL file `getCPDetails.wsdl`.

Completing the Partner Link Configuration



Click **Apply** and **OK** to complete the configuration and create the partner link with the required WSDL settings for the File Adapter service.

The `getCPDetails` Partner Link appears in the BPEL process diagram.

Configuring the Invoke Activities

After adding and configuring partner links, you need to configure the following four Invoke activities:

1. To get the order details by invoking the `getOrderDetails` partner link through Synchronous Read operation to read the order from an input file.

2. To get the concurrent program details by invoking the `getCPDetails` partner link through Synchronous Read operation to read the concurrent program information from an input file.

3. To insert order data into Open Interface tables by invoking `InsertOrder` partner link.

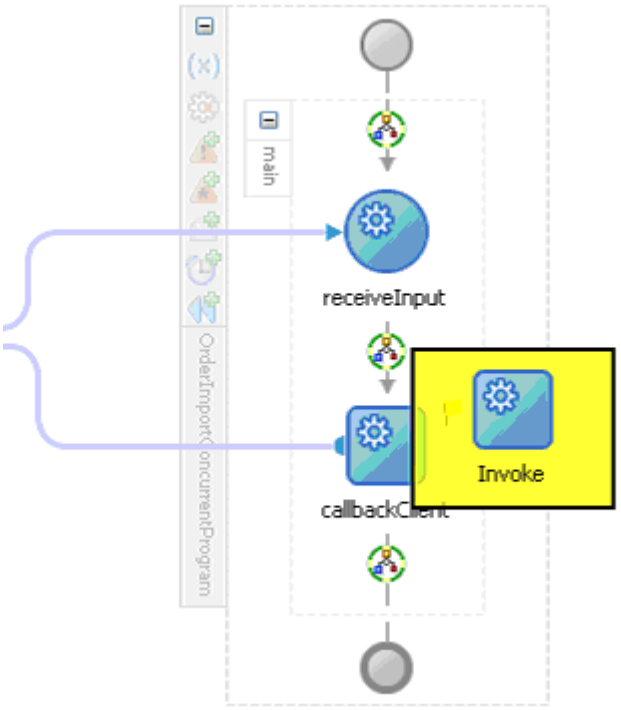
4. To import order data from Open Interface tables to Oracle Applications by invoking

`ImportOrderCP` concurrent program partner link.

To add the first Invoke activity for a partner link to get order details:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette. Drag and drop the first **Invoke** activity into the center swim lane of the process diagram, between the **receiveInput** and **callbackClient** activities.

Adding an Invoke Activity



2. Link the Invoke activity to the `getOrderDetails` service. The Edit Invoke dialog appears.

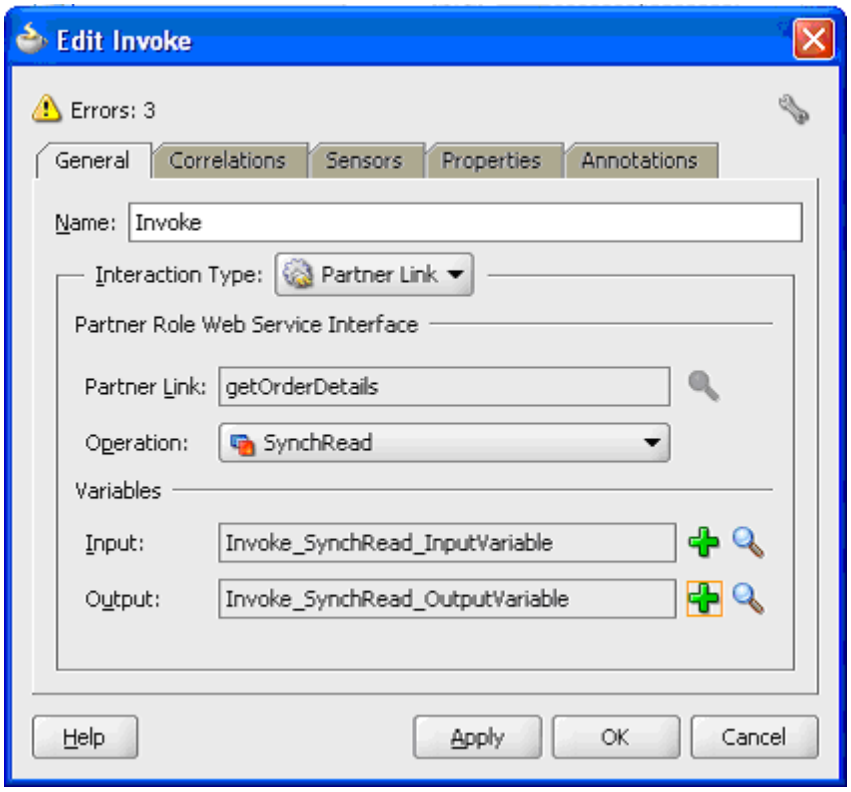
3. Enter a name for the Invoke activity, then click the **Create** icon next to the **Input Variable** field to create a new variable. The Create Variable dialog box appears.

4. Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK**.

5. Click the **Create** icon next to the **Output Variable** field to create a new variable. The Create Variable dialog box appears.

6. Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK** to return to the Edit Invoke dialog box.

Click **Apply** and then **OK** to finish configuring the Invoke activity.



The Invoke activity appears in the process diagram.

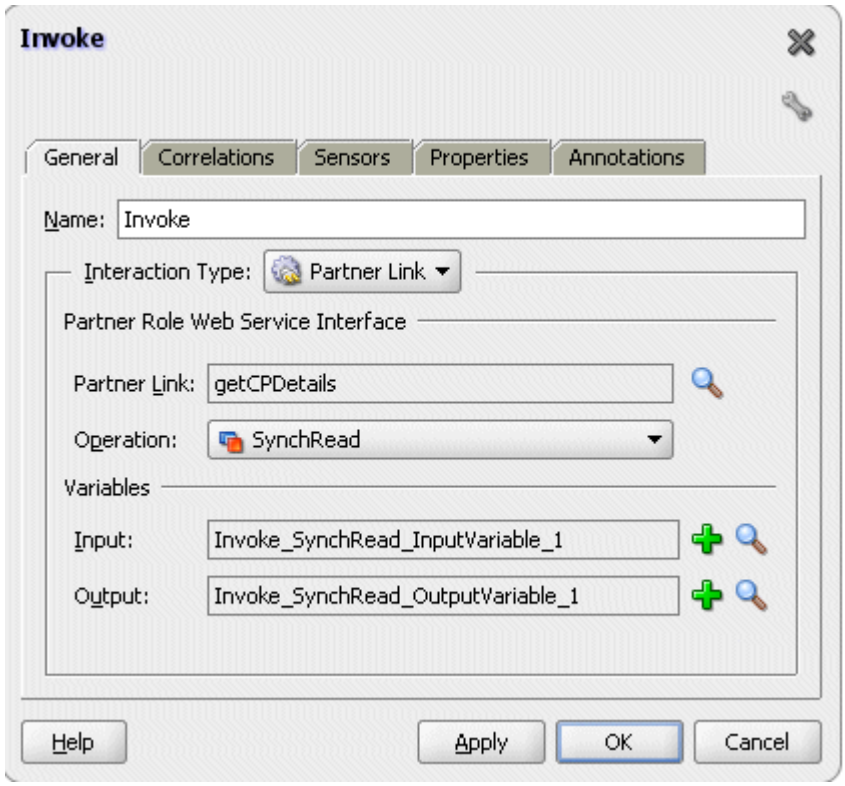
To add the second Invoke activity for a partner link to get concurrent program details:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette. Drag and drop the second **Invoke** activity into the center swim lane of the process diagram right after the first **Invoke** activity.

2. Link the Invoke activity to the `getCPDetails` service. The Edit Invoke dialog appears.

3. Repeat Step 3 to Step 6 described in the first Invoke activity procedure.

   The input and output variables should be populated in the Edit Invoke dialog box.

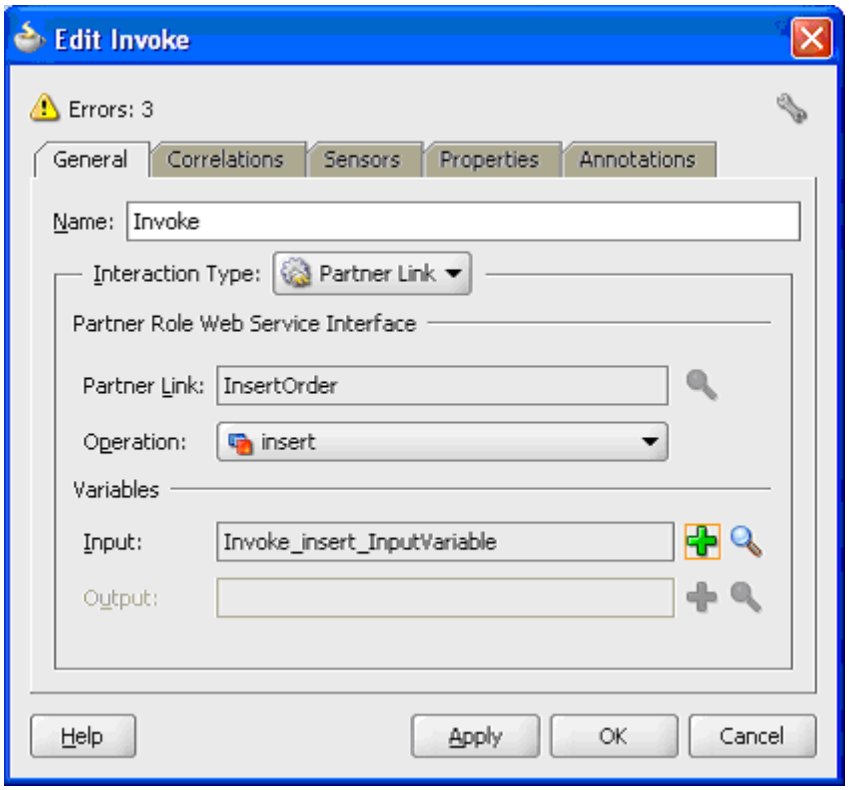


To add the third Invoke activity for a partner link to insert order data into Open Interface tables:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette. Drag and drop the third **Invoke** activity into the center swim lane of the process diagram, between the second **Invoke** and **callbackClient** activities.

2. Link the Invoke activity to the `InsertOrder` service. The Edit Invoke dialog box appears.

3. Repeat Step 3 to Step 4 described in the first Invoke activity procedure. Click **OK** in the Edit Invoke dialog box to finish configuring the second Invoke activity.



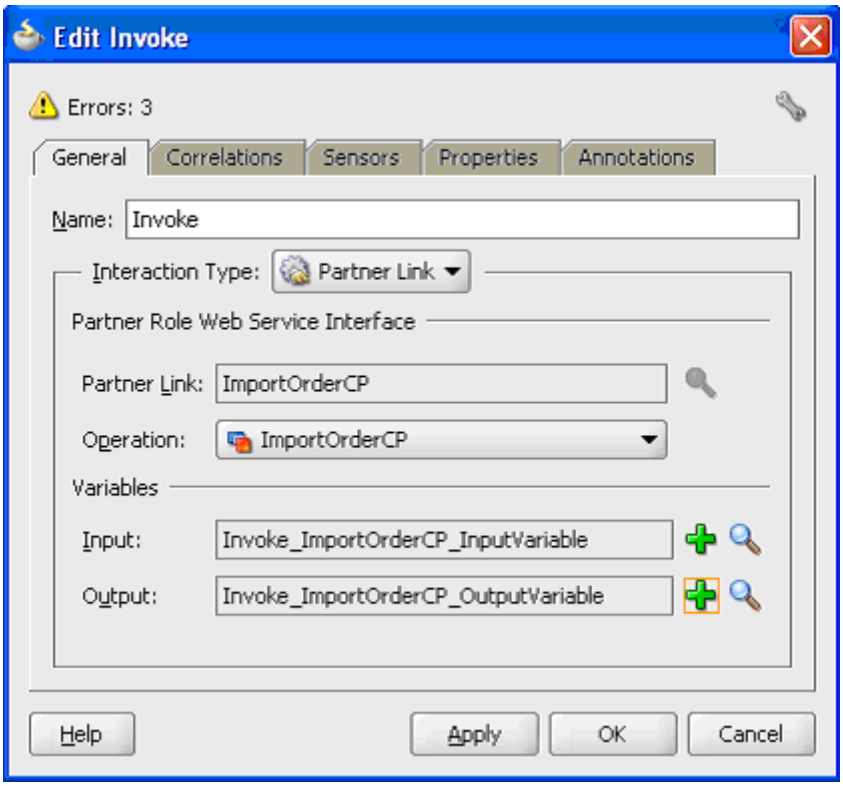
To add the fourth Invoke activity for a partner link to import the order information to Oracle Applications:

1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the fourth **Invoke** activity into the center swim lane of the process diagram, between the third **Invoke** and **callbackClient** activities.

2. Link the Invoke activity to the `ImportOrderCP` service. The Edit Invoke dialog box appears.

3. Repeat Step 3 to Step 6 described in the first Invoke activity procedure.

   The input and output variables should be populated in the Edit Invoke dialog box.



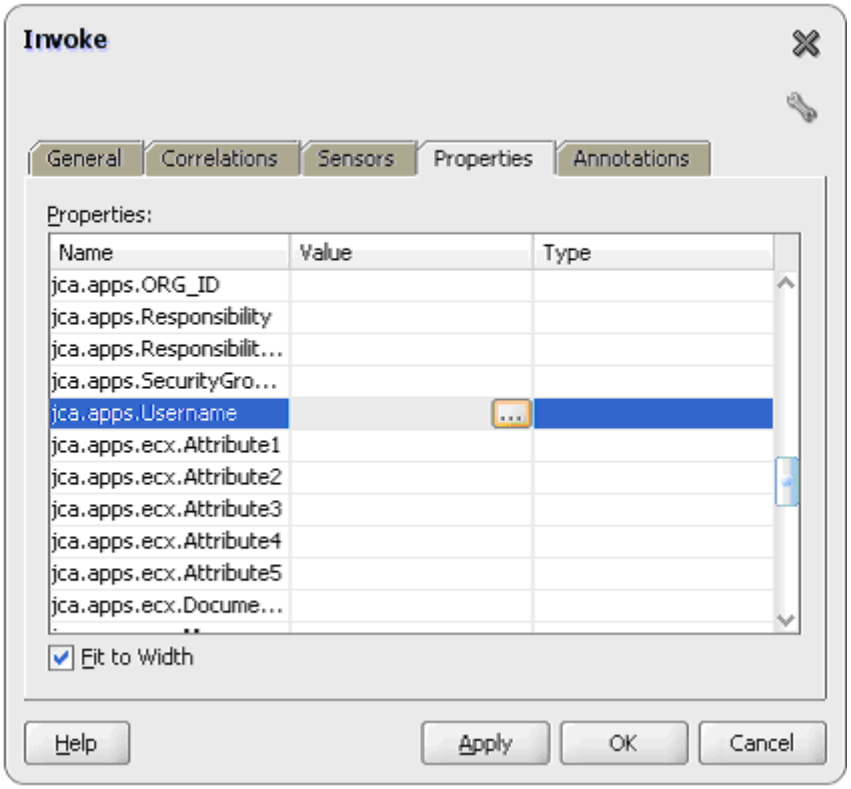
4. **Setting Header Properties for Application Context**

   Use the following steps to set the header message properties required to pass application context required to complete the BPEL process:

   1. Select the Properties tab in the Edit Invoke dialog box.

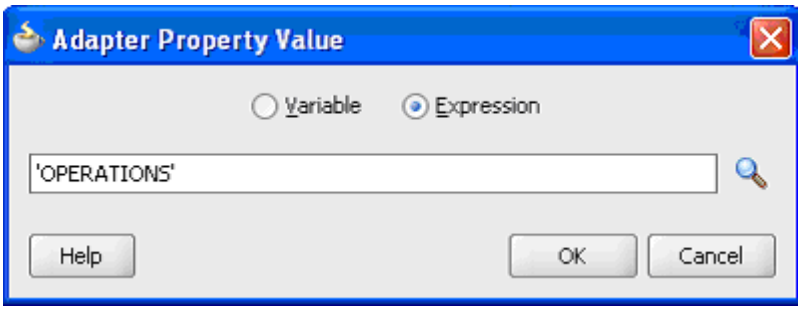
   2. Scroll down to locate the `jca.apps.Username` property from the list and double click the associated value field to enable the **Adapter Property Value** icon.

Setting Header Message Properties



3. Click the icon to open the Adapter Property Value dialog for the selected `jca.apps.Username` property.

Entering the Header Message Property Value

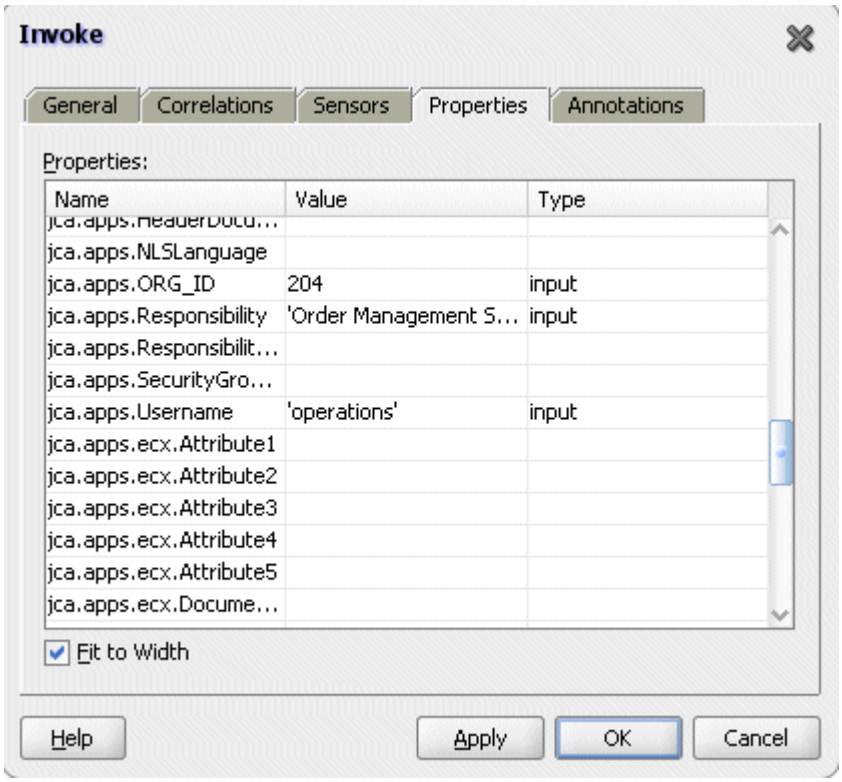


4. Select the **Expression** radio button and enter `'OPERATIONS'` as the property value.

   Click **OK**.

5. Repeat Step 2 to Step 4 to assign the following properties:

   - `'Order Management Super User, Vision Operations (USA)'` for `jca.apps.Responsibility`.
   - `204` for `jca.apps.ORG_ID`



5. Click **Apply** and then **OK** to complete the Invoke activity.

Configuring Assign Activities

Based on the BPEL process scenario, you need to configure the following two Assign activities:

1. To pass the output of File Adapter's Synchronous Read (`getOrderDetails`) service as an input to the Open Interface`InsertOrder` service.
2. To set the payload of the Concurrent Program (`ImportOrderCP`) service.

To add the first Assign activity:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette.

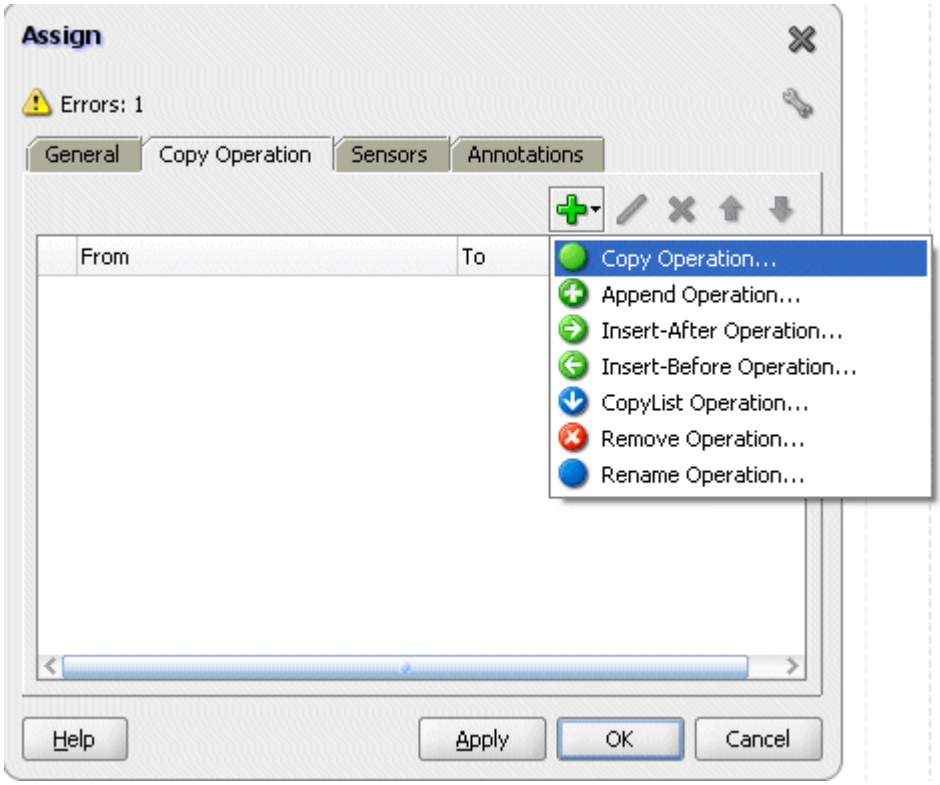
Drag and drop the first **Assign** activity into the center swim lane of the process diagram, between the first and second **Invoke** activities that you just created earlier.

2. Double-click the **Assign** activity to access the Edit Assign dialog.

   Click the General tab to enter a name for the Assign activity. For example, `SetOrderDetails`.

3. Select the Copy Operation tab, click the 'Plus' sign icon and select **Copy Operation** from the menu. The Create Copy Operation window appears.

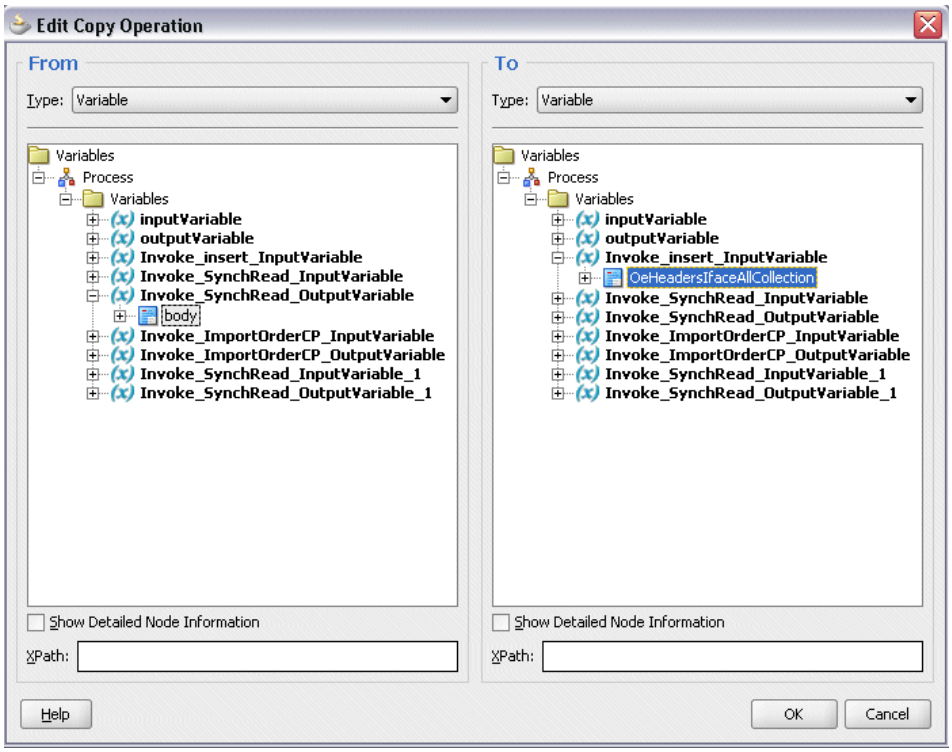
Specifying a Copy Operation Action



4. Enter the following parameter information:

   - In the From navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_SynchRead_OutputVariable** and select **body**.

   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_insert_InputVariable** and select **OeHeadersIfaceAllCollection**.



- Click **OK**. The Edit Assign dialog box appears.

5. Click **Apply** and then **OK** to complete the configuration of the Assign activity.

To add the second Assign activity:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette.

   Drag and drop the second **Assign** activity into the center swim lane of the process diagram, between the third and fourth **Invoke** activities that you just created earlier.

2. Double-click the **Assign** activity to access the Edit Assign dialog.

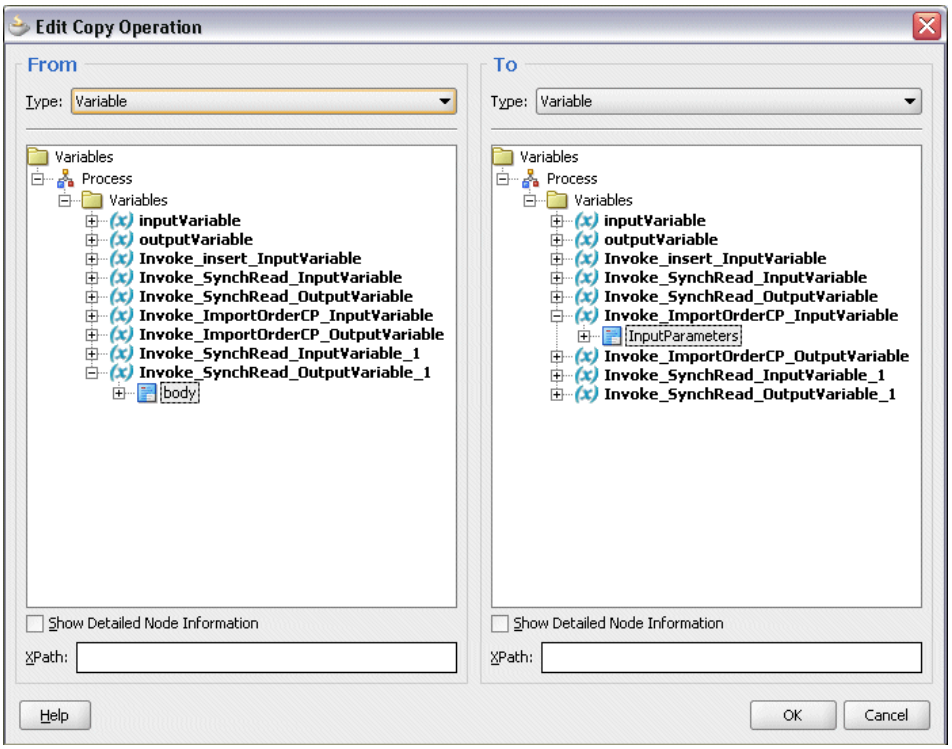
   Click the General tab to enter a name for the Assign activity. For example, `SetCPDetails`.

3. Select the Copy Operation tab, click the 'Plus' sign icon and select **Copy Operation** from the menu. The Create Copy Operation window appears.

4. Enter the following parameter information:

   - In the From navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_SynchRead_OutputVariable_1** and select **body**.

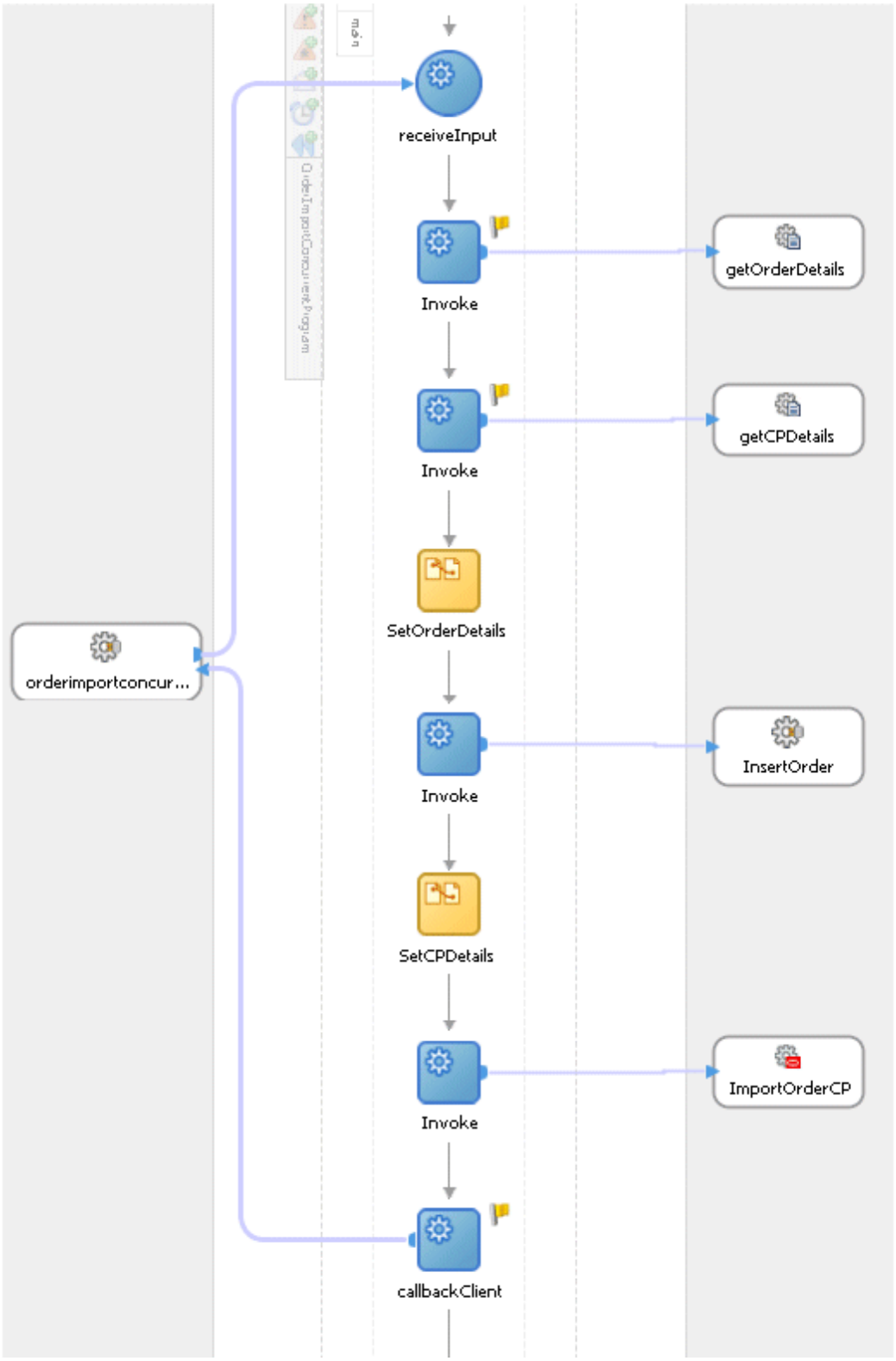
- In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_ImportOrderCP_InputVariable** and select **InputParameters**.



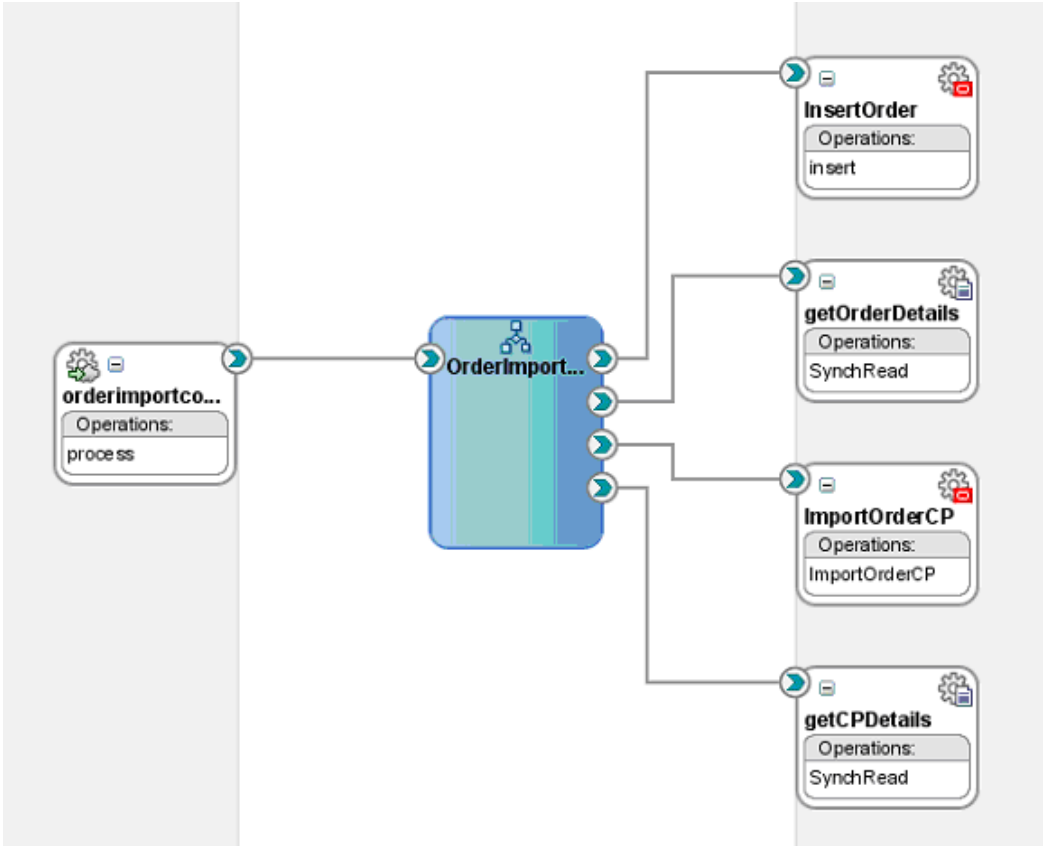
- Click **OK**. The Edit Assign dialog box appears.

5. Click **Apply** and then **OK** to complete the configuration of the Assign activity.

Completed Concurrent Program BPEL Process Project



Click the `composite.xml` to display the Oracle JDeveloper composite diagram:



Note: Click the Source tab of `composite.xml` to enter a value for the physical directory `inputDir` for the reference `getOrderDetails` and `getCPDetails` (such as `/usr/tmp`).

```
<property name="inputDir" type="xs:string"
many="false" override="may">/usr/tmp</property>
```

Specifying the Physical Directory for the Property

```
</reference>
<reference name="getOrderDetails" ui:wsdlLocation="getOrderDetails.wsdl">
  <interface.wsdl interface="http://xmlns.oracle.com/pcbpel/adapter/file/OrderImportC
  <binding.jca config="getOrderDetails_file.jca"/>
  <property name="inputDir" type="xs:string" many="false" override="may">/usr/tmp</pr
</reference>
<reference name="ImportOrderCP" ui:wsdlLocation="ImportOrderCP.wsdl">
  <interface.wsdl interface="http://xmlns.oracle.com/pcbpel/adapter/apps/OrderImportC
  <binding.jca config="ImportOrderCP_apps.jca"/>
</reference>
<reference name="getCPDetails" ui:wsdlLocation="getCPDetails.wsdl">
  <interface.wsdl interface="http://xmlns.oracle.com/pcbpel/adapter/file/OrderImportC
  <binding.jca config="getCPDetails_file.jca"/>
  <property name="inputDir" type="xs:string" many="false" override="may">/usr/tmp</pr
</reference>
<wire>
  <source.uri>orderimportconcurrentprogram_client_ep</source.uri>
  <target.uri>OrderImportConcurrentProgram/orderimportconcurrentprogram_client</targe
</wire>
<wire>
  <source.uri>OrderImportConcurrentProgram/InsertOrder</source.uri>
  <target.uri>InsertOrder</target.uri>
</wire>
<wire>
  <source.uri>OrderImportConcurrentProgram/getOrderDetails</source.uri>
  <target.uri>getOrderDetails</target.uri>
</wire>
```

Run-Time Tasks for Concurrent Programs

After designing the BPEL process, the next step is to deploy, run and monitor it.

1. Deploy the BPEL process., page 6-54
2. Test the BPEL process., page 6-57
3. Verify records in Oracle Applications., page 6-60

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 application server (Oracle WebLogic Server) that you have established the connection.

Prerequisites

Before deploying the BPEL process using Oracle JDeveloper, you must ensure the following:

- You must have established the connectivity between the deign-time environment and an application server.

  For more information, see Configuring the Data Source in Oracle WebLogic Server,

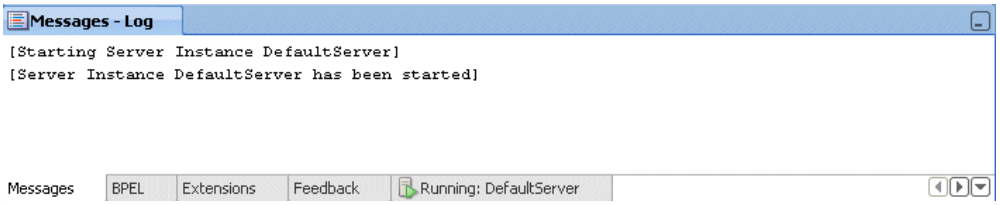
page A-3 and Creating an Application Server Connection, page A-8.

- Oracle WebLogic Server has been started.

  Before deploying the BPEL process, you need to start the Oracle WebLogic Server that you have established the connection.

  If a local instance of the WebLogic Server is used, start the WebLogic Server by selecting **Run > Start Server Instance** from Oracle JDeveloper.

  Once the WebLogic Admin Server "DefaultServer" instance is successfully started, you should find that <Server started in Running mode> and DefaultServer started message in the Running:DefaultServer and Messages logs.

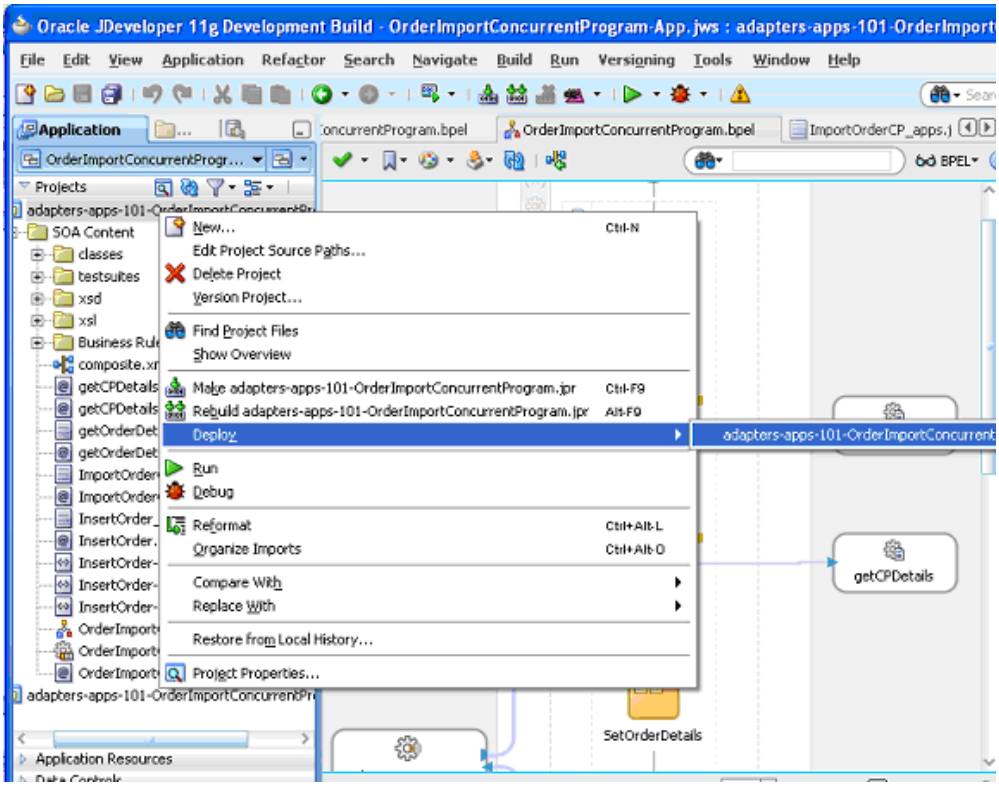


To deploy the BPEL process:

1. Select the BPEL project in the Applications Navigator.

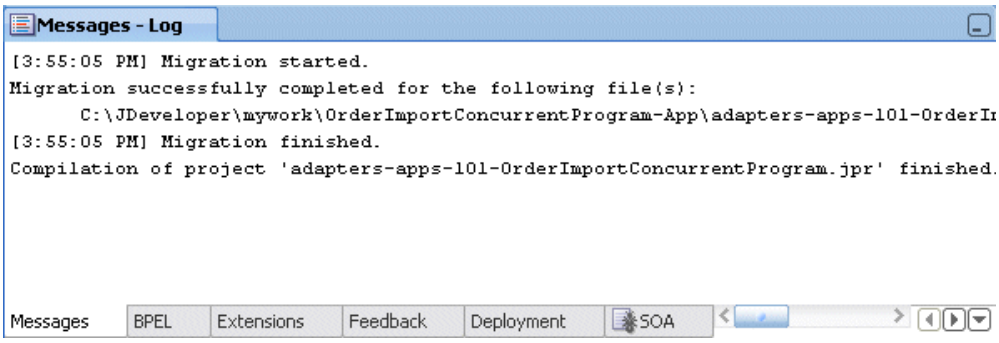
2. Right-click the project name, then select **Deploy > [project name] > [serverConnection]** from the menu that appears.

Deploying the BPEL Process

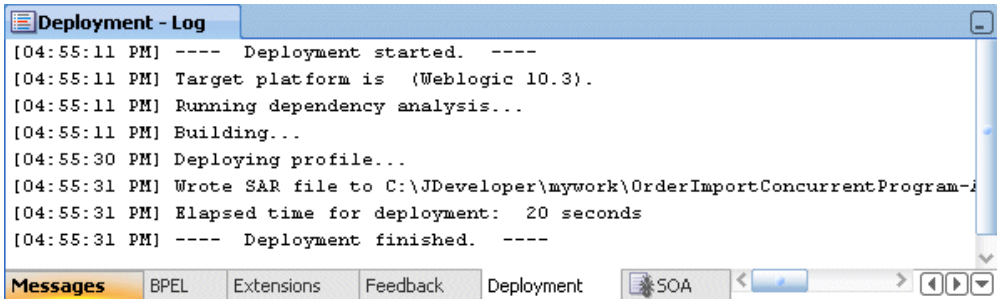


3. The BPEL process is compiled and deployed. You can check the progress in the Messages window.

Messages Log



Deployment Log

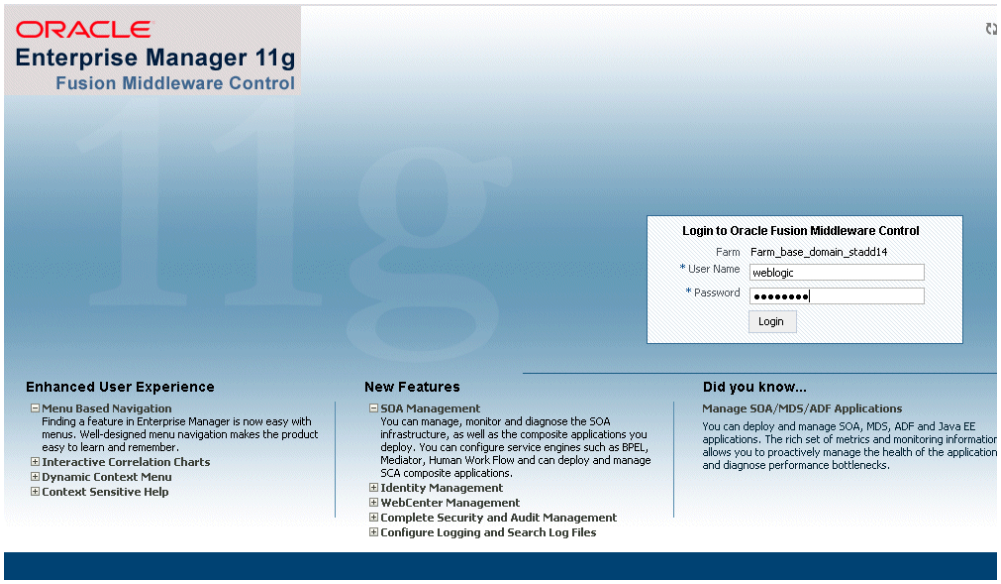


Testing the BPEL Process

Once the BPEL process is deployed, you can manage and monitor the process from the Oracle Enterprise Manager Fusion Middleware Control Console. You can also test the process and the integration interface by manually initiating the process.

To test the BPEL process:

1. Navigate to Oracle Enterprise Manager Fusion Middleware Control Console (`http://<servername>:<portnumber>/em`). The composite you deployed is displayed in the Applications Navigation tree.



2. Enter username (such as `weblogic`) and password and click **Login** to log in to a farm.

   You may need to select an appropriate target instance farm if there are multiple

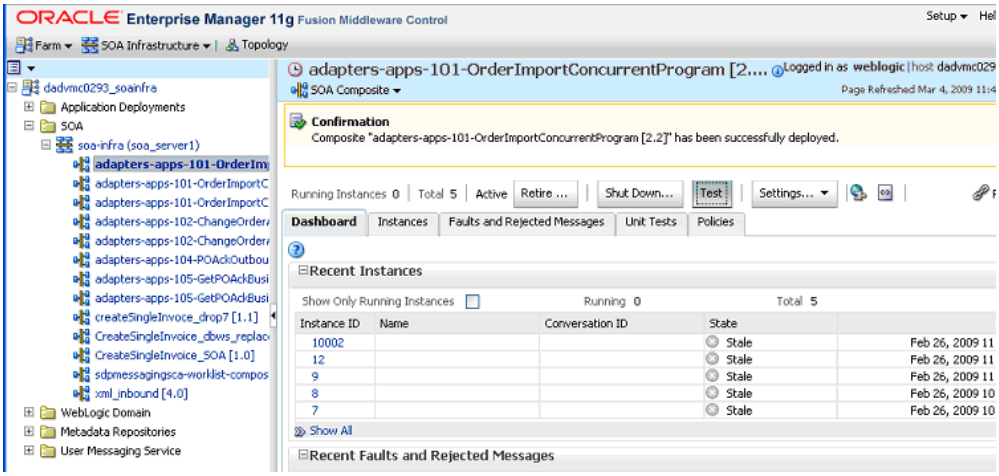
target Oracle Enterprise Manager Fusion Middleware Control Console farms.

3. From the Farm base domain, expand the **SOA >soa-infra** to navigate through the SOA Infrastructure home page and menu to access your deployed SOA composite applications running in the SOA Infrastructure for that managed server.

   Note: The Farm menu always displays at the top of the navigator. As you expand the SOA folder in the navigator and click the links displayed beneath it, the SOA Infrastructure menu becomes available at the top of the page.

   Click the SOA composite application that you want to initiate (such as 'OrderImportConcurrentProgram') from the SOA Infrastructure.

   Viewing Deployed SOA Composites

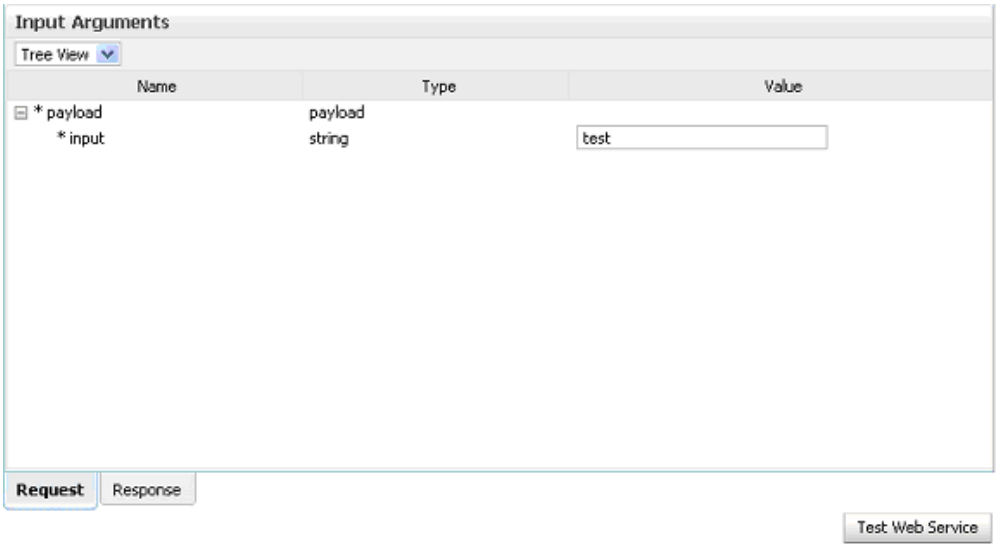


   Click **Test** at the top of the page.

4. The Test Web Service page for initiating an instance appears. You can specify the XML payload data to use in the Input Arguments section.

   Enter the input string required by the process and click **Test Web Service** to initiate the process.

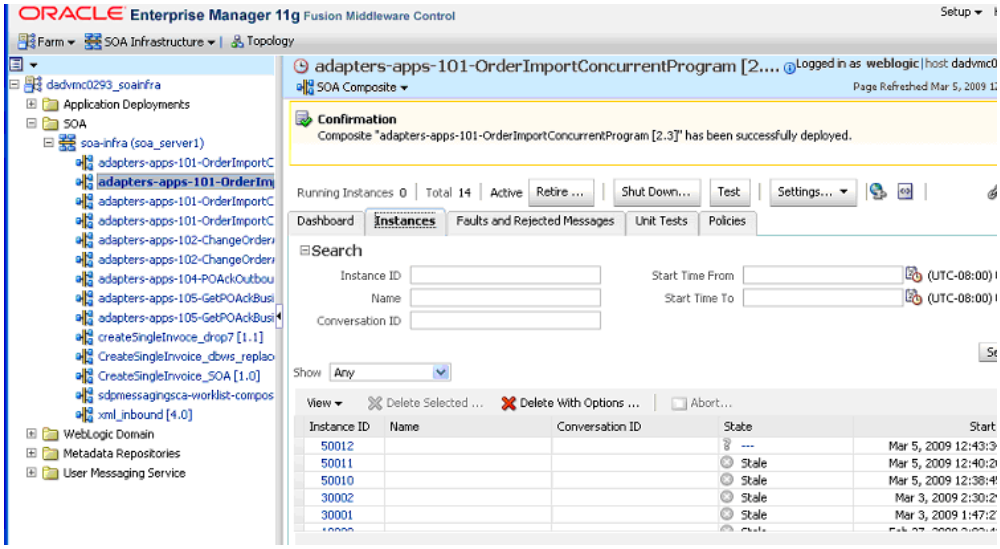
Testing Web Service



The test results appear in the Response tab upon completion.

5. Click on the BPEL process name and then select the Instances tab.

The SOA composite application instance ID, name, conversation ID, most recent known state of each instance since the last data refresh of the page are displayed.



In the Instance ID column, click a specific instance ID to show the message flow through the various service components and binding components. The Flow Trace page is displayed.

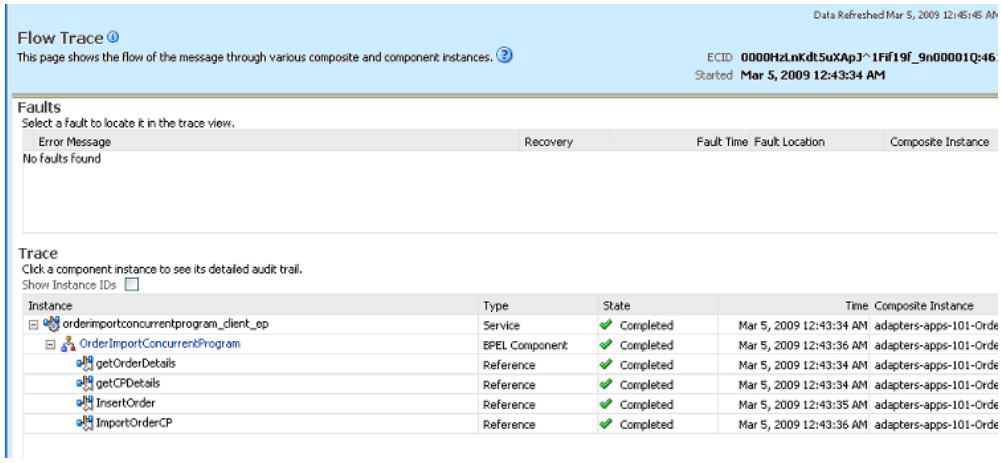
In the Trace section, you should find the sequence of the message flow for the

service binding component (`orderimportconcurrentprogram_client_ep`), BPEL component (`OrderImportConcurrentProgram`), and reference binding components (`getOrderDetails`, `getCPDetails`, `InsertOrder` and `ImportOrderCP`). All involved components have successfully received and processed messages.

Click the Instances tab and then click the latest Instance ID to see the process flow trace.

The Flow Trace page is displayed.

Flow Trace Page



If any error occurred during the test, you should find it in the Faults section.

6. Click your BPEL service component instance link (such as `OrderImportConcurrentProgram`) to display the Instances page where you can view execution details for the BPEL activities in the Audit Trail tab.

   Click the Flow tab to check the BPEL process flow diagram. Click an activity of the process diagram to view the activity details and flow of the payload through the process.

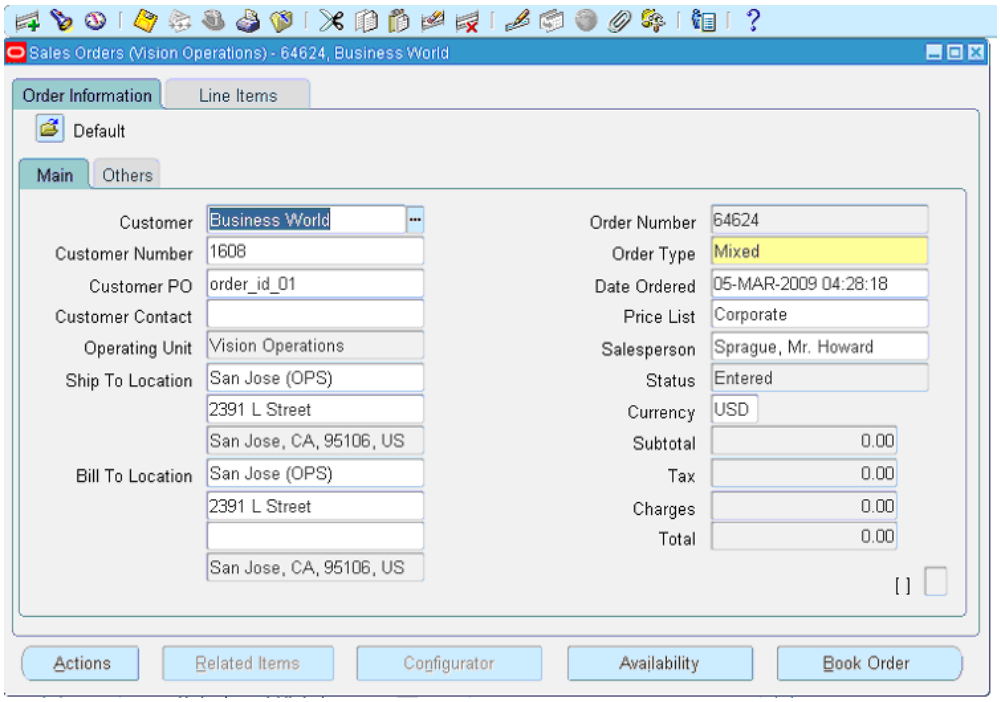
Verifying Records in Oracle Applications

To validate the BPEL process, you can log on to Oracle Applications forms to check the status of the Concurrent Program execution. Once it is successfully executed, the purchase order should also be created in Oracle Applications.

To verify records in Oracle Applications:

1. Log in to Oracle Applications with the Order Management Super User, Vision Operations (USA) responsibility.

2. Choose Orders, Return > Sales Order to open the Sales Orders form.



3. Search for an order by pressing **F11** key. In the Customer PO field, enter the order ID present in the xml file. For example, enter `order_id_01` and press **CTRL+F11** keys to execute the query.

4. You should find the newly created order appears with order ID `order_id_01`.

   Select the Line Items tab to verify the quantity and item type of the order which should be the same as that present in the xml file

5. Alternatively, you can also validate the result by querying the database with the following SQL statement:

```
Select orig_sys_document_ref from oe_order_headers_all where
orig_sys_document_ref = 'order_id_01';
```

Troubleshooting and Debugging

If you experience problems with your concurrent program integration, you can take the following troubleshooting steps:

- If you're using complex (that is, Boolean) datatypes, ensure that PL/SQL wrappers are applied on the target instance of the concurrent program.
- Examine your transaction data in the Open Interface Tracking Form.

- Check the status of the concurrent programs by Request ID.

If you still experience problems with your integration, you can enable debugging.

Enabling Debugging

You can enable debugging using the BPEL Process Manager.

To enable debugging:

1. Log into your BPEL Process Manager domain.
2. Select *yourdomain*.collaxa.cube.ws
3. Select **Debug**.

Debugging information is output to the log file for your domain. To examine the log file in the BPEL Process Manager, navigate to **Home > BPEL Domains >*yourdomain* > Logs**. The log file is ***yourdomain*.log**.

7

Using Interface Tables and Views

This chapter covers the following topics:

- Overview of Interface Tables and Views
- Design-Time Tasks for Interface Tables
- Creating a New BPEL Project
- Adding a Partner Link
- Adding a Partner Link for File Adapter
- Configuring the Invoke Activities
- Configuring the Assign Activity
- Run-Time Tasks for Interface Tables
- Deploying the BPEL Process
- Testing the BPEL Process
- Design-Time Tasks for Views
- Creating a New BPEL Project
- Adding a Partner Link
- Adding a Partner Link for File Adapter
- Configuring the Invoke Activities
- Configuring the Assign Activity
- Run-Time Tasks for Views
- Deploying the BPEL Process
- Testing the BPEL Process

Overview of Interface Tables and Views

Adapter for Oracle Applications uses interface tables to insert and update data in Oracle Applications, and uses interface views to retrieve data from Oracle Applications. This chapter describes the following interfaces:

- Interface Tables
- Interface Views

Interface Tables

Adapter for Oracle Applications use open 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.

Adapter for Oracle Applications use open interface tables to integrate with Oracle Applications through direct database access. The 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. 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.

Views

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.

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 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.

Design-Time Tasks for Interface Tables

This section describes how to configure the Adapter for Oracle Applications to use interface tables. It describes the steps to configure Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper.

BPEL Process Scenario

Take Open Interface tables (`OE_HEADER_IFACE_ALL` and `OE_LINES_IFACE_ALL`) as examples.

When a request of inserting order data into Open Interface tables is received, order details will be retrieved through synchronous read operation from an input file and then inserted into Open Interface tables (`OE_HEADER_IFACE_ALL` and `OE_LINES_IFACE_ALL`).

If the BPEL process is successfully executed after deployment, you can validate the result by fetching the inserted data using SQL statement.

Prerequisites to Configure Interface Tables

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

BPEL Process Flow

Following is a list of the procedures required to accomplish the design-time tasks.

1. Create a new BPEL project., page 7-3
2. Add a partner link., page 7-6
3. Add a partner link for File Adapter., page 7-23
4. Configure the Invoke activities., page 7-28
5. Configure the Assign activity., page 7-31

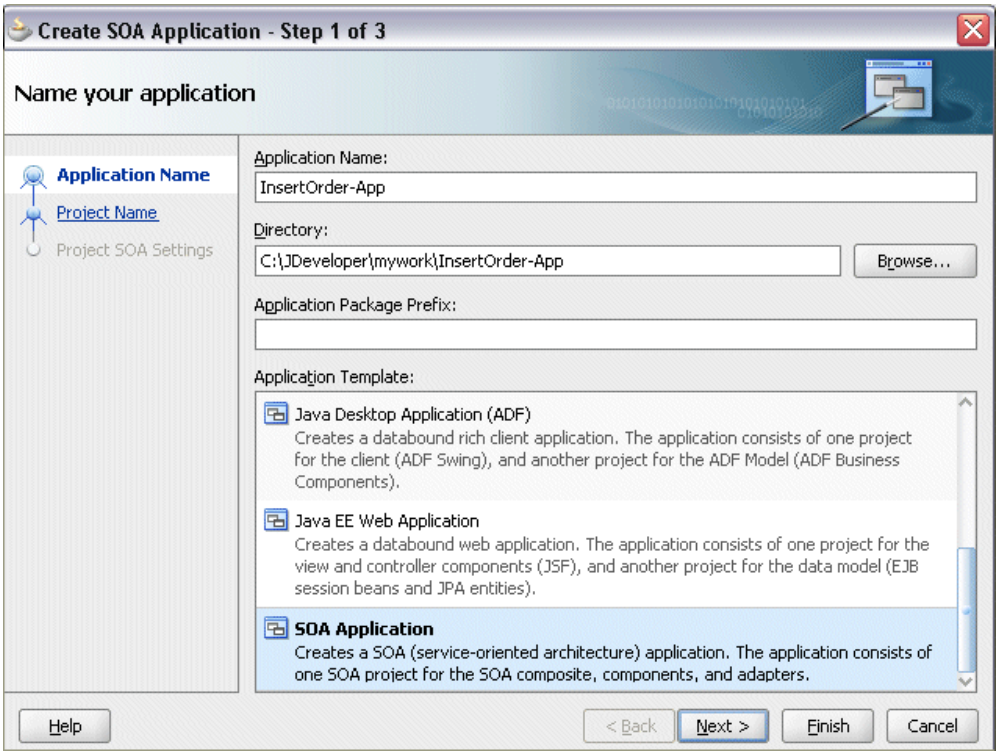
Creating a New BPEL Project

To create a new BPEL project:

1. Open JDeveloper BPEL Designer. Click **New Application** in the Application Navigator.

   The Create SOA Application - Name your application page is displayed.

The Create SOA Application - Name your application Page

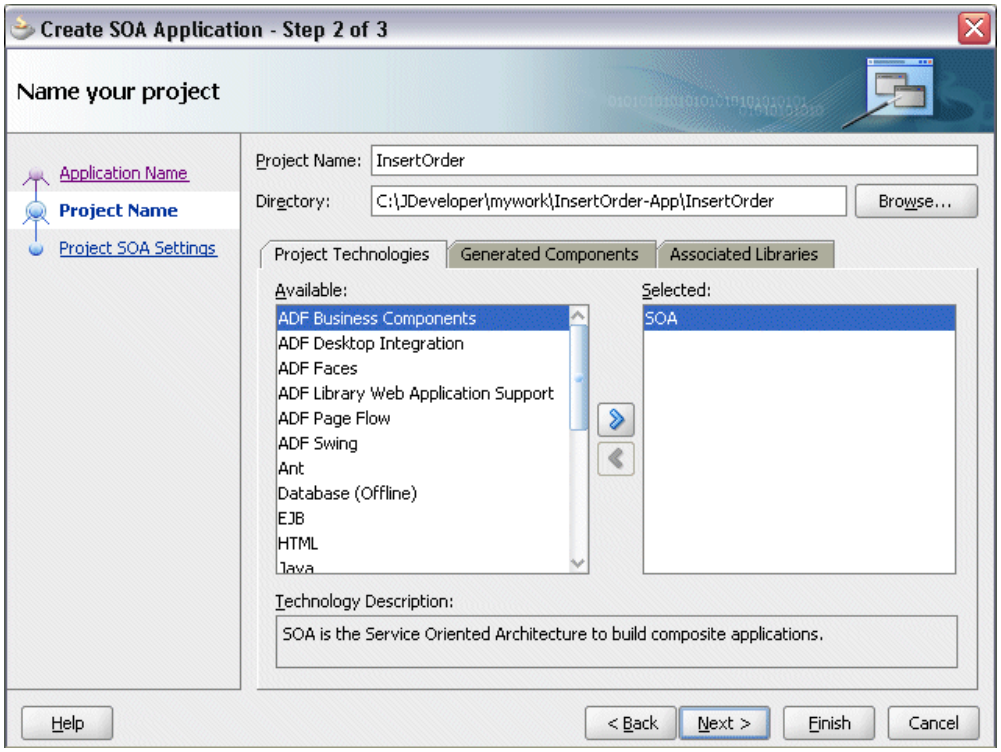


2. Enter an appropriate name for the application in the **Application Name** field and select **SOA Application** from the Application Template list.

   Click **Next**. The Create SOA Application - Name your project page is displayed.

3. Enter an appropriate name for the project in the **Project Name** field. For example, `InsertOrder`.

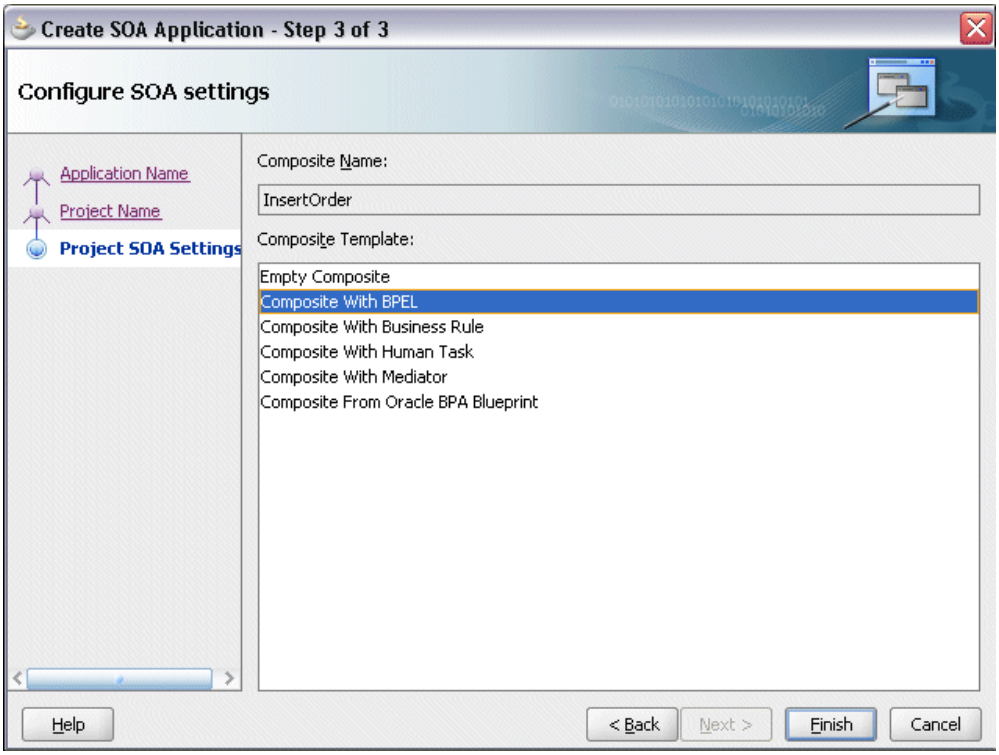
The Create SOA Application - Name your project Page



4. In the Project Technologies tab, ensure that **SOA** is selected from the Available technology list to the Selected technology list.

   Click **Next**. The Create SOA Application - Configure SOA settings page is displayed.

The Create SOA Application - Configure SOA settings Page



5. In the **BPEL Process Name** field, enter a descriptive name. For example, `InsertOrder`.

6. Select **Composite With BPEL** from the Composite Template list, and then click **Finish**. You have created a new application, and an SOA project. This automatically creates an SOA composite.

   The Create BPEL Process page is displayed.

7. Enter an appropriate name for the BEPL process in the **Name** field. For example, `InsertOrder`.

   Select **Asynchronous BPEL Process** in the **Template** field. Click **OK**.

   An asynchronous BPEL process is created with the Receive and Reply activities. The required source files including bpel and wsdl, using the name you specified (for example, `InsertOrder.bpel` and `InsertOrder.wsdl`) and `composite.xml` are also generated.

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 to insert order details into selected Open Interface tables:

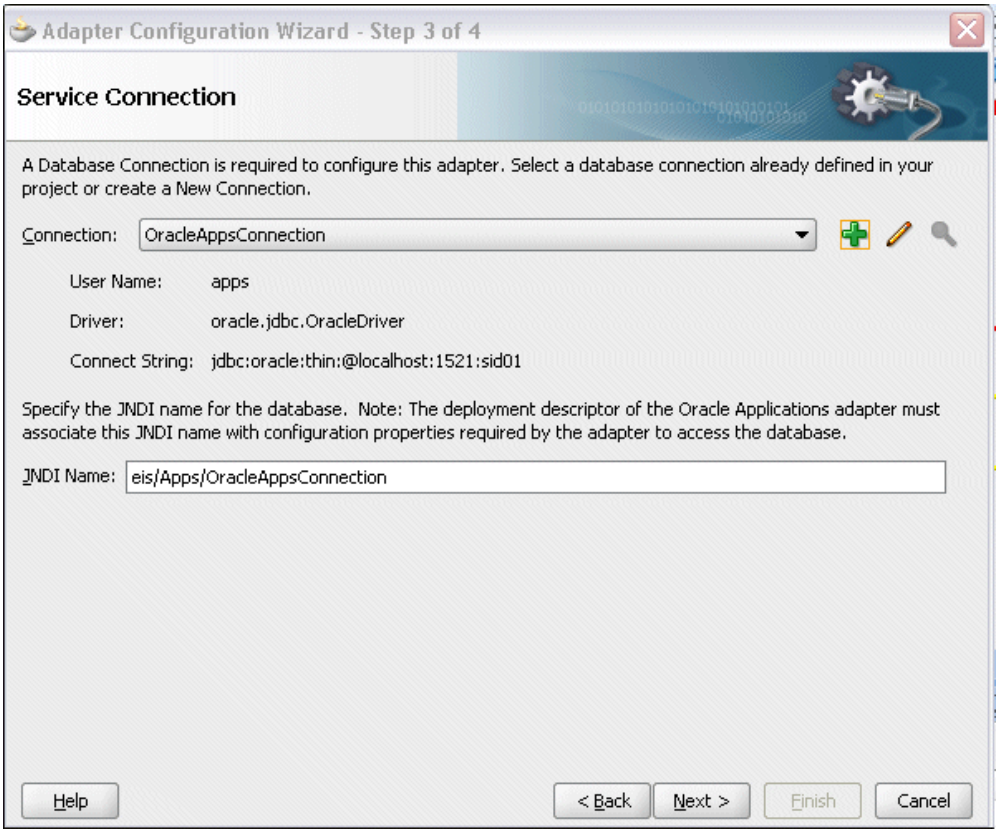
1. Click **BPEL Services** in the Component palette.

   Drag and drop **Oracle Applications** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears. Click **Next**.

2. Enter a service name in the **Service Name** field. For example, `InsertOrder`.

   Click **Next**. The Service Connection dialog appears.

Specifying a Database Service Connection



3. You can perform either one of the following options for your database connection:

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

   - You can create a new database connection by clicking the **Create a New Database Connection** icon.

     How to define a new database connection, see Create a New Database Connection, page 4-13.

   - You can select an existing database connection that you have configured earlier from the **Connection** drop-down list.

     The Service Connection page will be displayed with the selected connection information. The JNDI (Java Naming and Directory Interface) name corresponding to the database connection appears automatically in the **Database Server JNDI Name** field. Alternatively, you can specify a JNDI name.

Note: When you specify a JNDI name, the deployment descriptor of the Oracle Applications adapter must associate this JNDI name with configuration properties required by the adapter to access the database.

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.

Note: For more information about JNDI concepts, refer to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

4. Click **Next** in the Service Connection dialog box. You can add an interface table by browsing through the list of interface tables available in Oracle Applications.

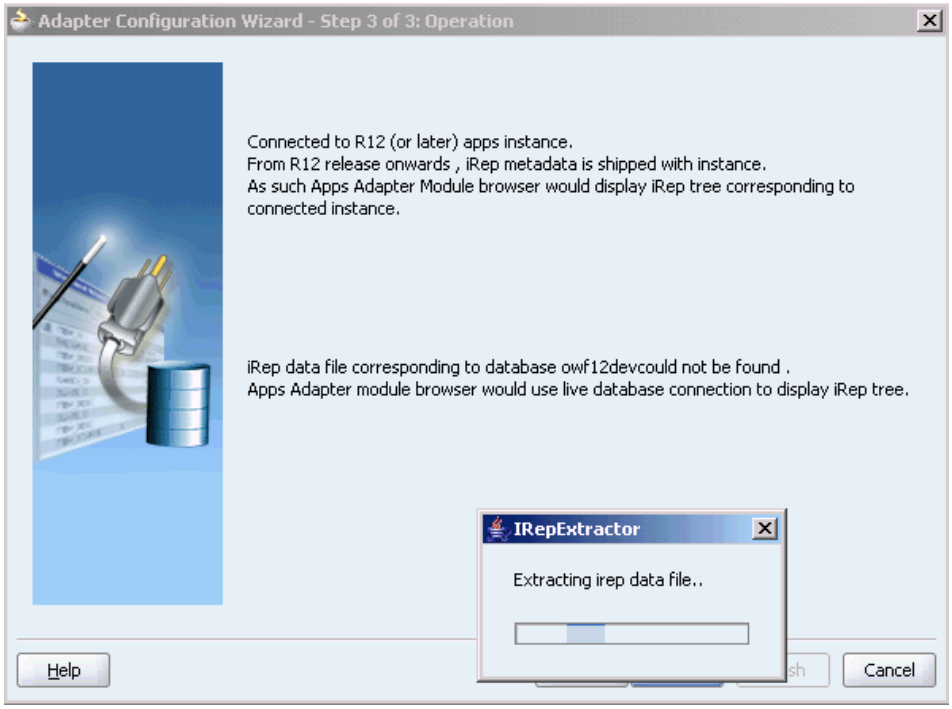
   For Oracle E-Business Suite Release 12:

   If you are connecting to Oracle E-Business Suite Release 12, then the **IREP File not present** dialog box appears indicating that Adapter could not find the Oracle Integration Repository data file corresponding to the database you are connecting in your workspace. Absence of the data file would make browsing or searching of Integration Repository tree considerably slow. You can choose to extract the data file and create a local copy of the Integration Repository data file. Once it is created successfully, Adapter will pick it up automatically next time and retrieve data from your local Integration Repository.

   You can select one of the following options:

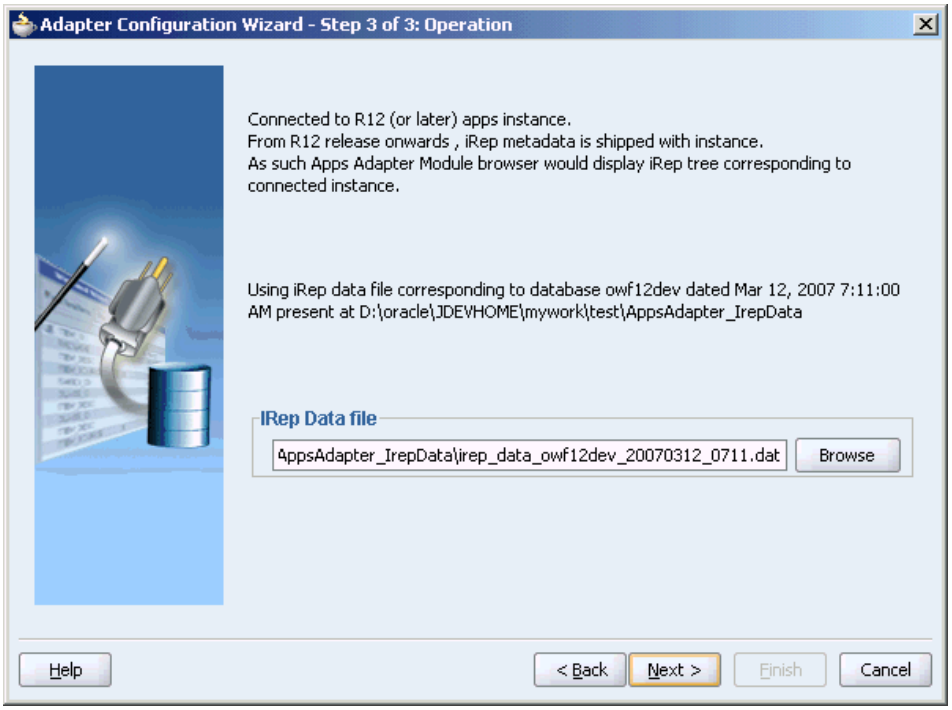
   - Click **Yes** to extract the Integration Repository data file.

Extracting Integration Repository Data File



After the system successfully creates a local copy of the Integration Repository data file, next time when you connect to the database, you will find the **IRep Data File** field appears in the Operation dialog box indicating where your local copy exists with the creation date and time as part of the file name.

Using the Local Integration Repository Data File



- Click **No** to query the Integration Repository data file from the live database you are connecting to display the Integration Repository tree.

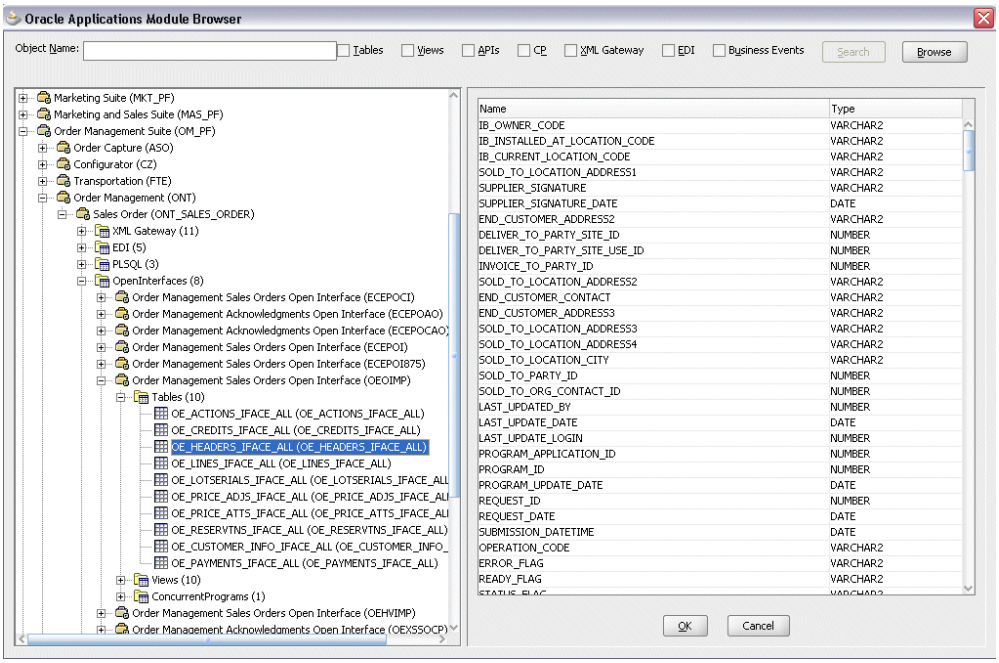
  Note: It is highly recommended that you create a local copy of the Integration Repository data file so that Adapter will query the data next time from the local copy in your workspace to enhance the performance.

For Oracle E-Business Suite pre-Release 11.5.10:

If you are connecting to a pre-11.5.10 Oracle Applications instance, you must select the interface type in the Adapter Configuration Wizard. Select **Tables/Views/APIs/Concurrent Programs** to proceed.

5. Click **Get Object** to open the Oracle Applications Module Browser.

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.

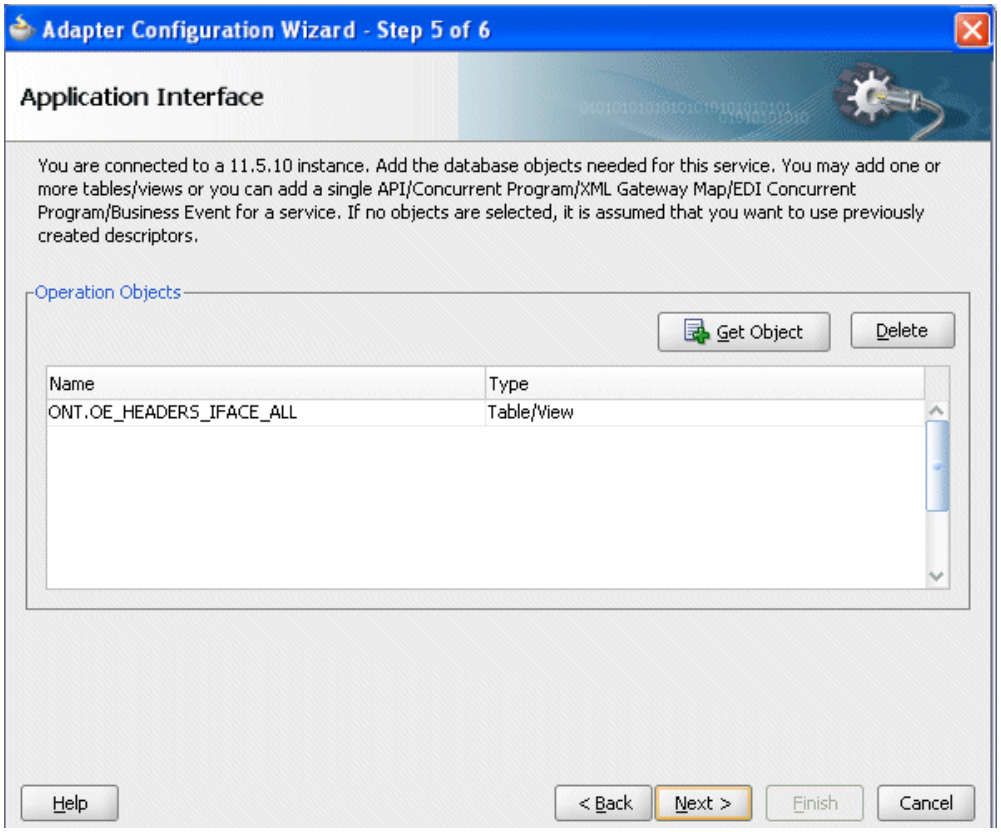
Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide. concurrent programs can be found under the Open Interfaces category.

6. Navigate to *Order Management Suite (OM_PF) > Order Management (ONT) > Sales Order (ONT_SALES_ORDER) > OpenInterfaces >Order Management Sales Orders Open Interface (OEOIMP) > Tables* to select `OE_HEADERS_IFACE_ALL`.

   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, then click **Search**.

Click **OK**. The Application Interface page appears with the selected open table.

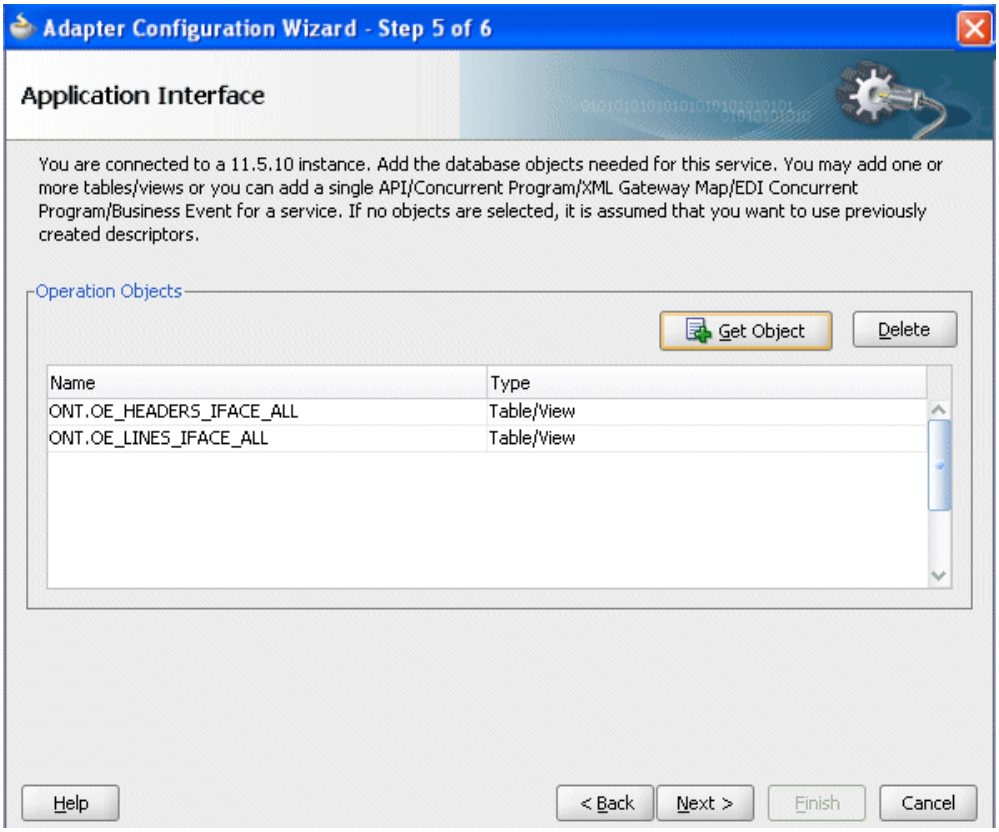
Adapter Configuration Wizard - Application Interface Page



7. Click **Get Object** to open the Oracle Applications Module Browser again to select another open interface table `OE_LINES_IFACE_ALL` using the same navigation path *Order Management Suite (OM_PF) > Order Management (ONT) > Sales Order (ONT_SALES_ORDER) > OpenInterfaces >Order Management Sales Orders Open Interface (OEOIMP) > Tables*.

   Click **OK**. The Application Interface page appears with the two selected tables.

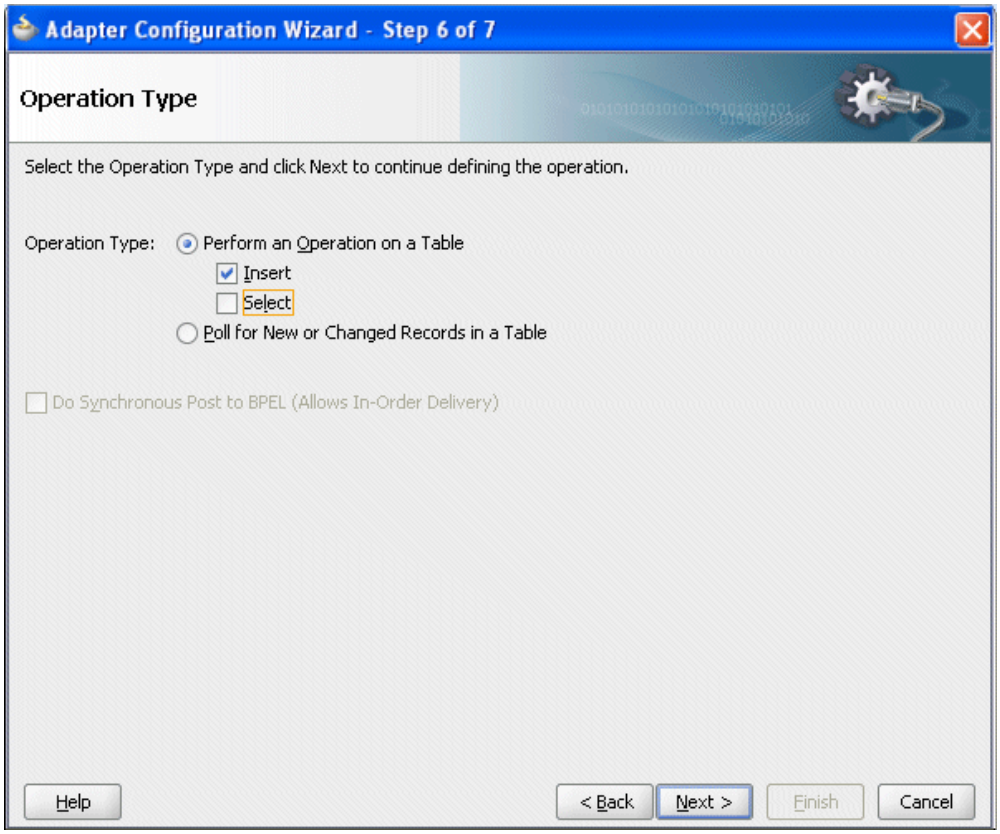
Adapter Configuration Wizard - Application Interface Page



Click **OK**.

8. Click **Next**. The Operation Type page is displayed.

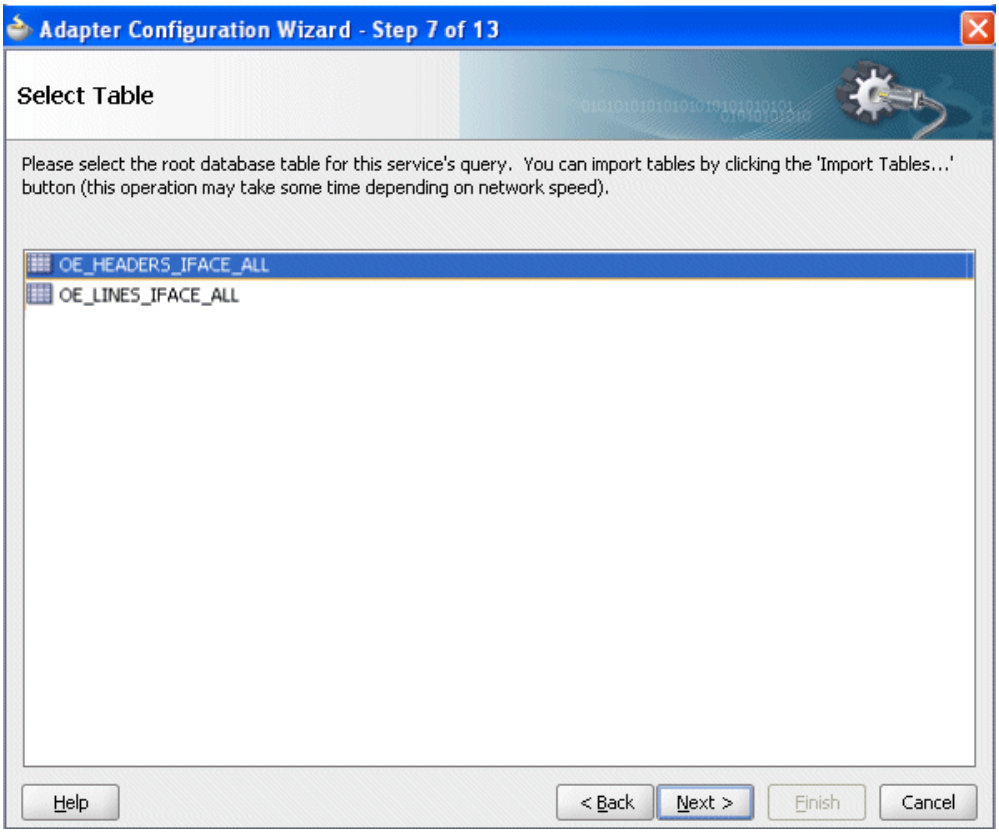
Adapter Configuration Wizard - Operation Type Page



Select the **Perform an Operation on a Table** radio button and then choose the **Insert** check box. Ensure that the **Select** check box is deselected.

9. Click **Next**. The Select Table page appears 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.

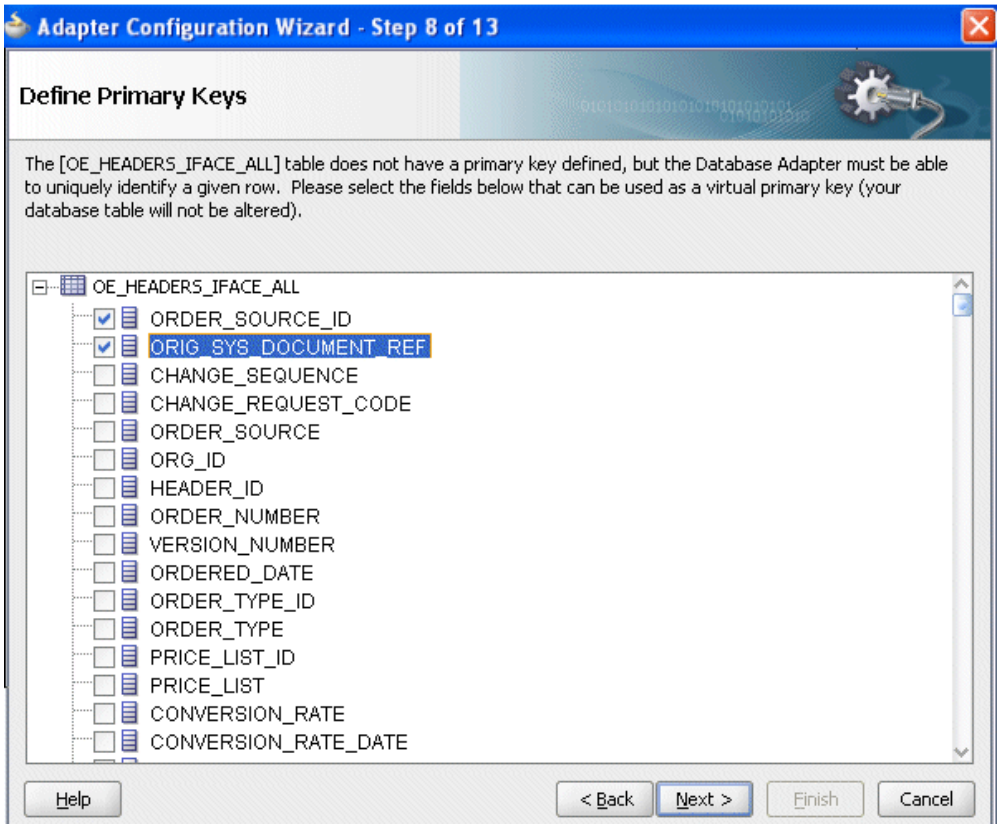
Adapter Configuration Wizard - Select Table Page



Select OE_HEADERS_IFACE_ALL as the root database table for this service's query.

10. Click **Next**. The Define Primary Keys page is displayed.

Adapter Configuration Wizard - Define Primary Keys Page for OE_HEADERS_IFACE_ALL



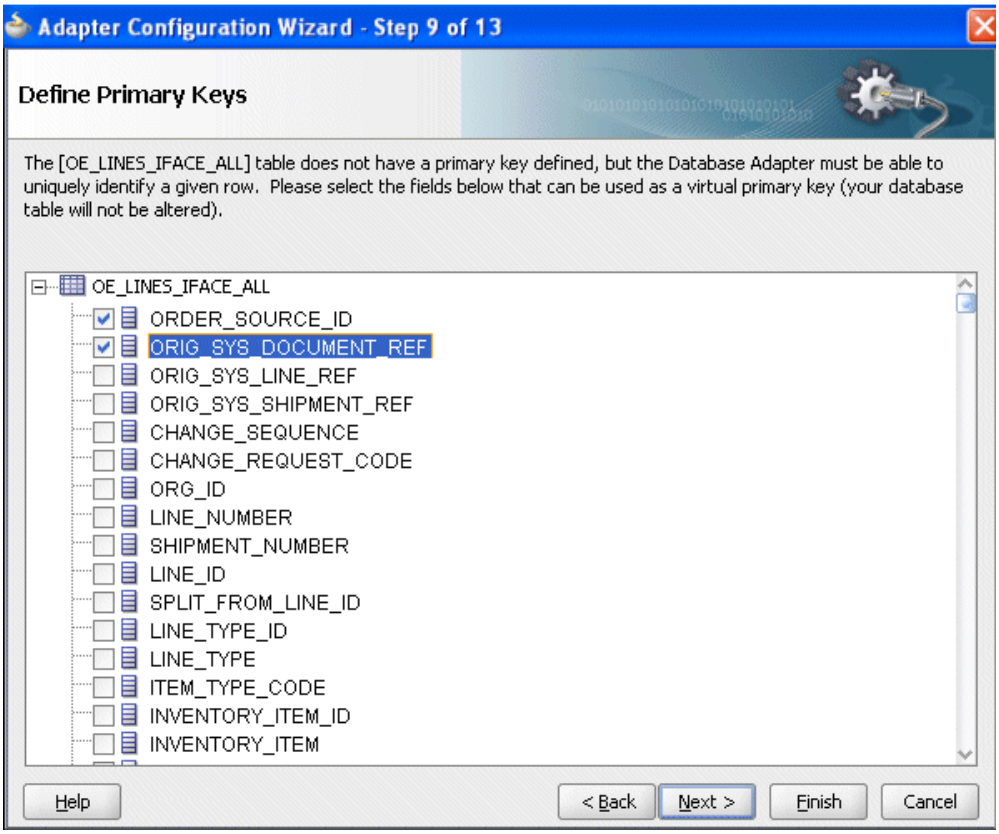
Select the following primary keys for the OE_HEADERS_IFACE_ALL table:

- ORDER_SOURCE_ID
- ORIG_SYS_DOCUMENT_REF

Click **Next**.

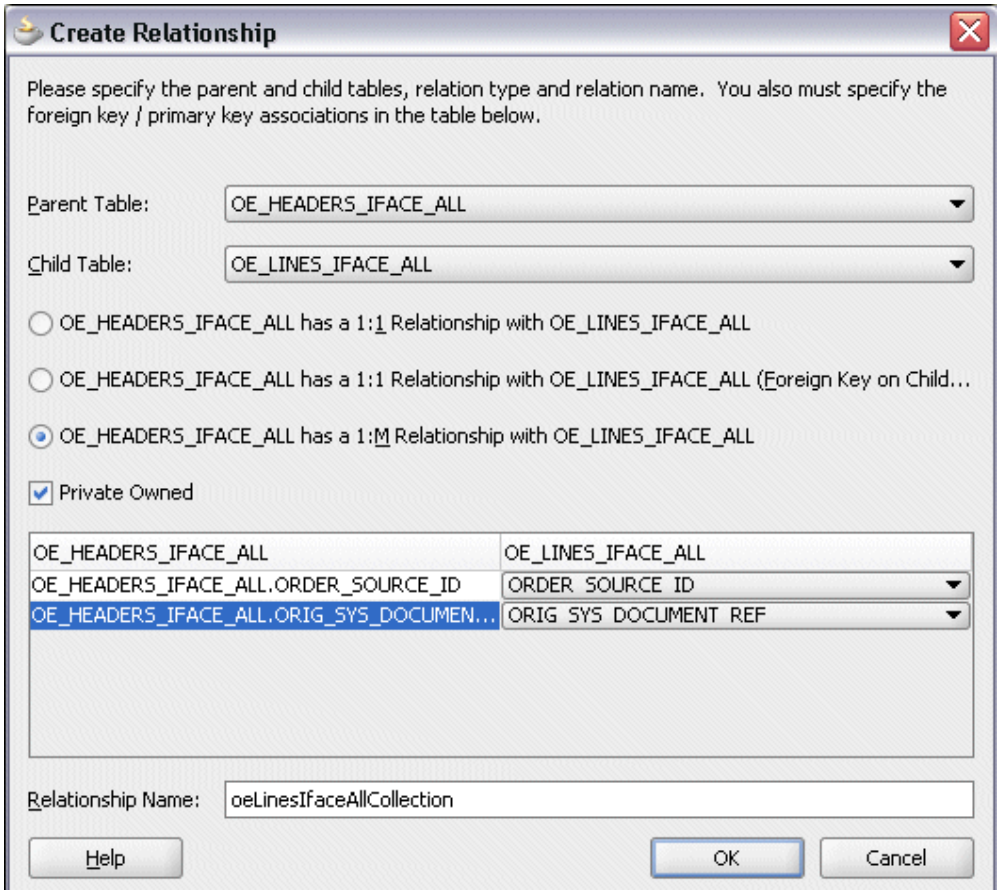
Select the same primary keys for the OE_LINES_IFACE_ALL table.

Adapter Configuration Wizard - Define Primary Keys Page for OE_LINES_IFACE_ALL



11. Click **Next**. The Relationships page appears. Click **Create** to open the Create Relationship dialog.

Defining Relationships



Enter the following information to define the relationship between the header and the detail table:

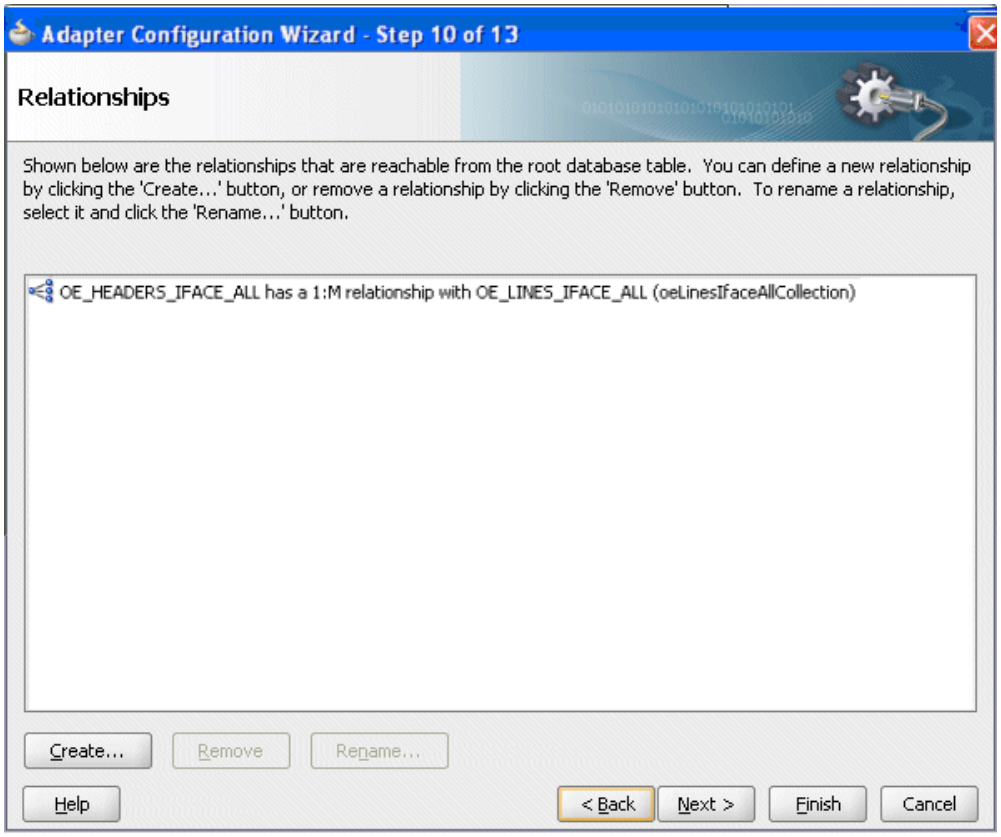
- Select the OE_HEADERS_IFACE_ALL as the parent table and OE_LINES_IFACE_ALL as the child table.
- Select the mapping type: OE_HEADERS_IFACE_ALL has a 1:M Relationship with OE_LINES_IFACE_ALL

> **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.

- Select the **Private Owned** check box.
- Associate the foreign key fields with the primary key fields:
  - OE_HEADERS_IFACE_ALL.ORDER_SOURCE_ID: ORDER_SOURCE_ID
  - OE_HEADERS_IFACE_ALL.ORIG_SYS_DOCUMENT_REF: ORIG_SYS_DOCUMENT_REF
- The Relationship Name field is populated automatically by default. You can optionally specify a new name for the relationship you are creating.

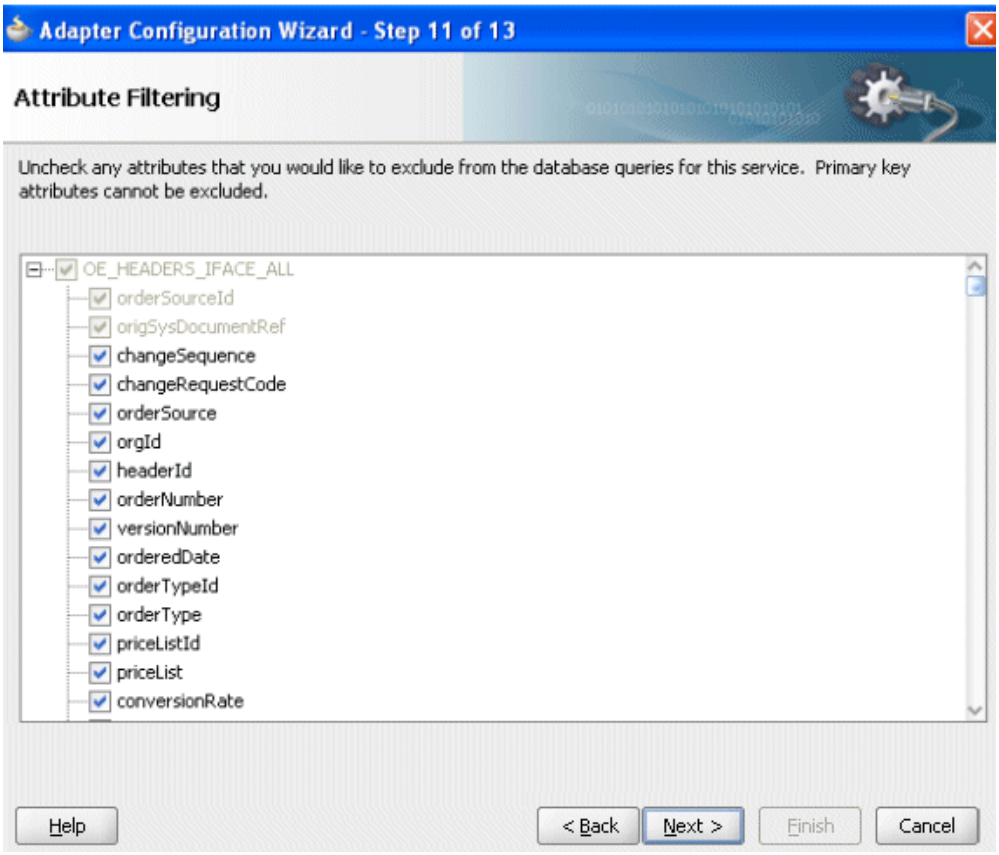
Click **OK**. The newly created relationship information appears in the Relationships page.

Adapter Configuration Wizard - Relationships Page



12. Click **Next**. The Attribute Filtering page appears. Leave default selections unchanged.

Adapter Configuration Wizard - Attribute Filtering Page

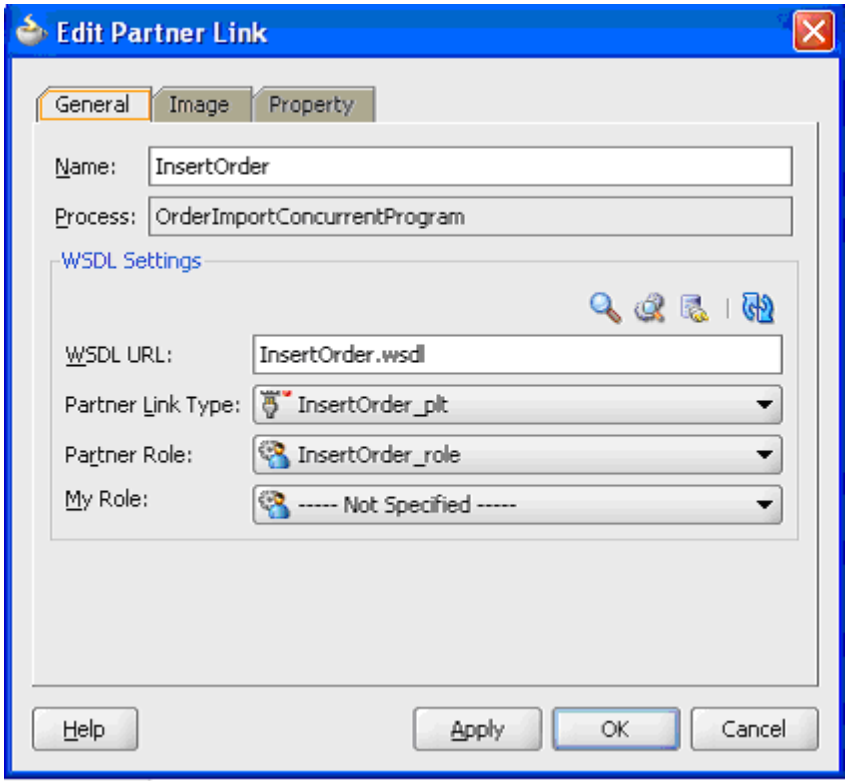


Click **Next**. The Advanced Options page appears.

Click **Next**.

13. Click **Finish**. The wizard generates the WSDL file corresponding to the selected interface. This WSDL file is now available for the partner link.

Completing the Partner Link Configuration



14. Click **Apply** and then **OK**. The partner link is created with the required WSDL settings.

Adding a Partner Link for File Adapter

Use this step to configure a BPEL process by synchronously reading payload from an input file to obtain the purchase order details.

To add a Partner Link for File Adapter to read order details:

1. In JDeveloper BPEL Designer, click **BPEL Services** in the Component palette.

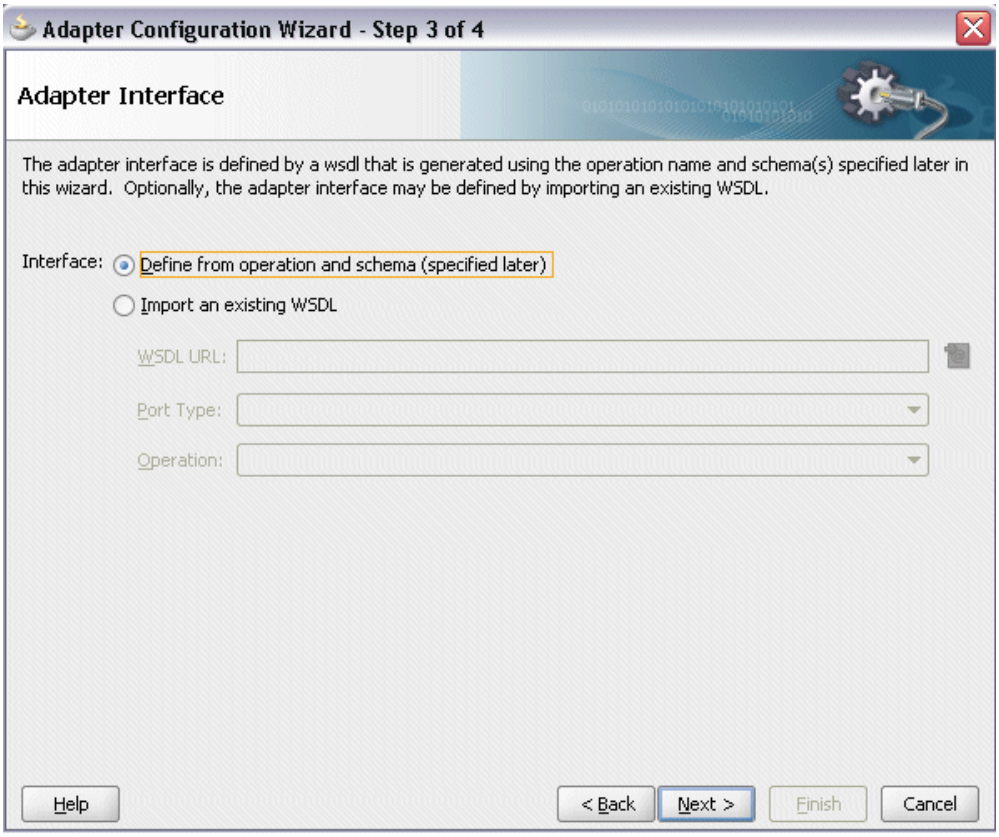
   Drag and drop **File Adapter** from the BPEL Services list into the right Partner Link swim lane before the first partner link `InsertOrder`. The Adapter Configuration Wizard Welcome page appears.

   Click **Next**.

2. In the Service Name dialog, enter a name for the File Adapter service, such as `getOrderDetails`.

3. Click **Next**. The Adapter Interface page appears.

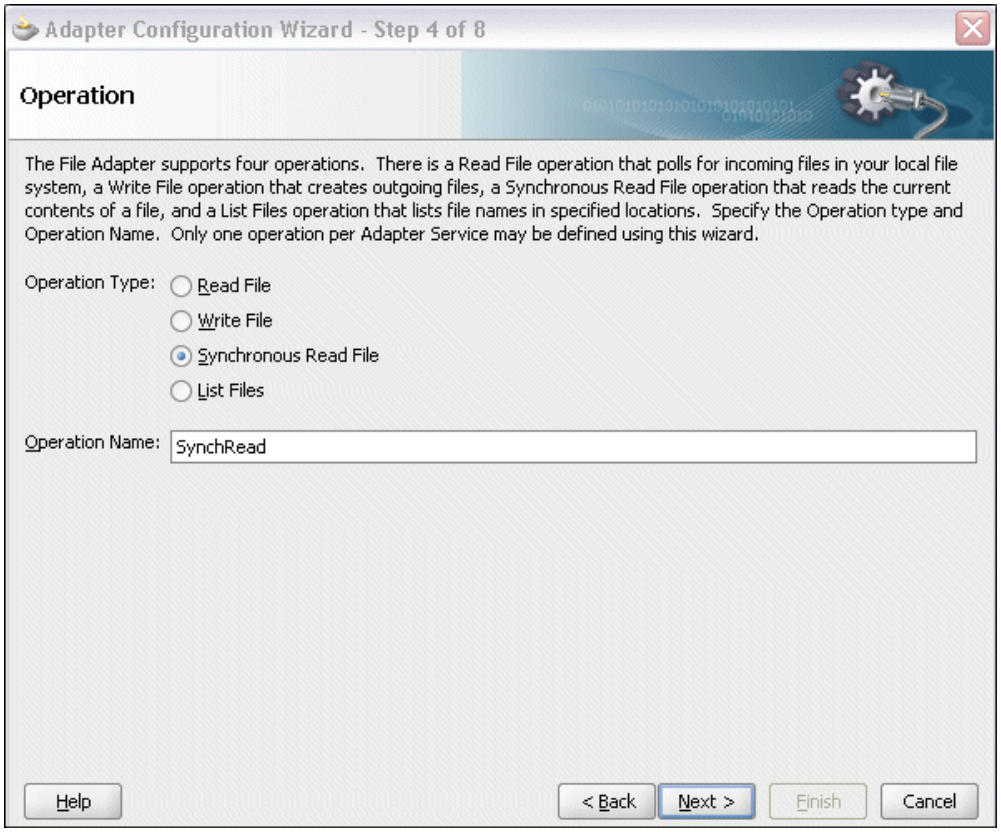
Specifying the Adapter Interface



Select the **Define from operation and schema (specified later)** radio button and click **Next**.

4. In the Operation page, specify the operation type. For example, select the **Synchronous Read File** radio button. This automatically populates the **Operation Name** field.

Specifying the Operation

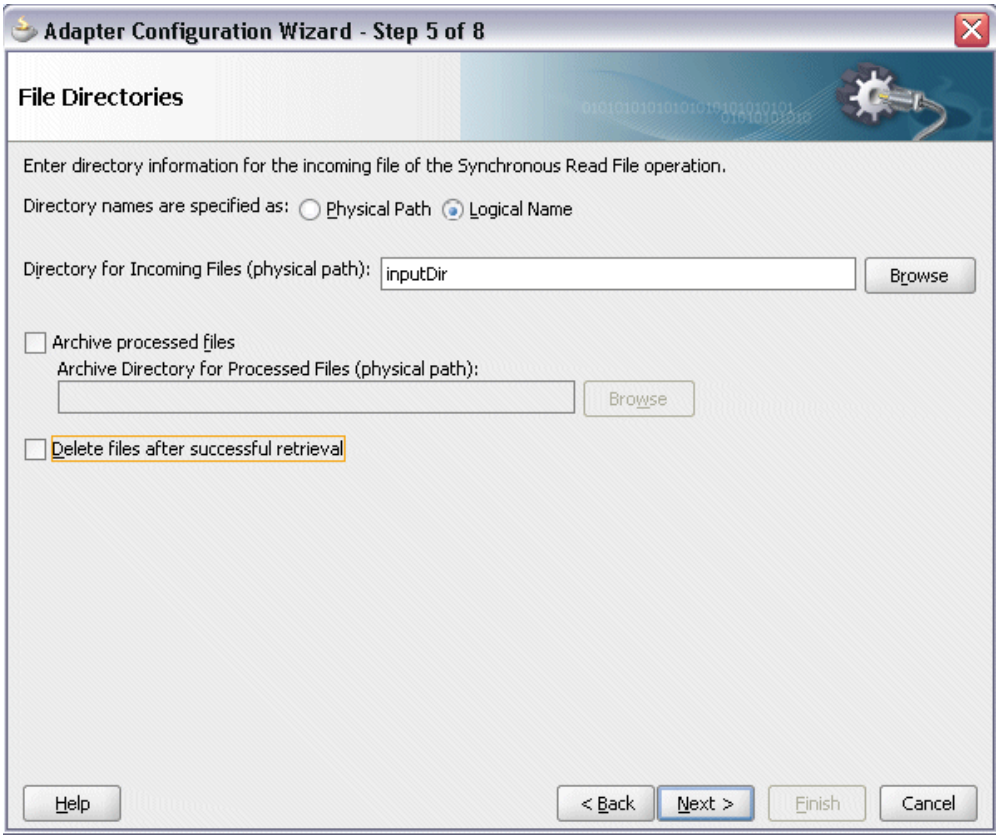


Click **Next** to access the File Directories page.

5. Select the **Logical Name** radio button and specify directory for incoming files, such as `inputDir`.

   Ensure the **Delete files after successful retrieval** check box is not selected.

Configuring the Input File



Click **Next** to open the File Name page.

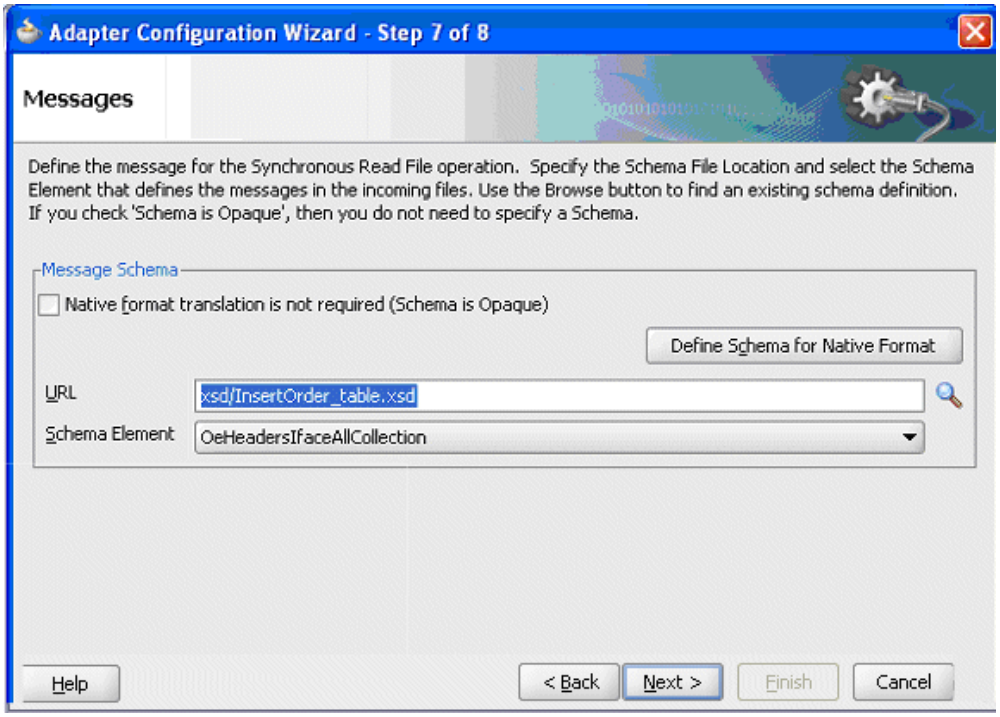
6. Enter the name of the file for the synchronous read file operation. For example, enter `order_data.xml`. Click **Next** to open the Messages page.

7. Select the 'browse for schema file' icon next to the URL field to open the Type Chooser.

   Click Type Explorer and select *Project Schema Files* > *InsertOrder_table.xsd* > *OeHeadersIfaceAllCollection*. Click **OK**.

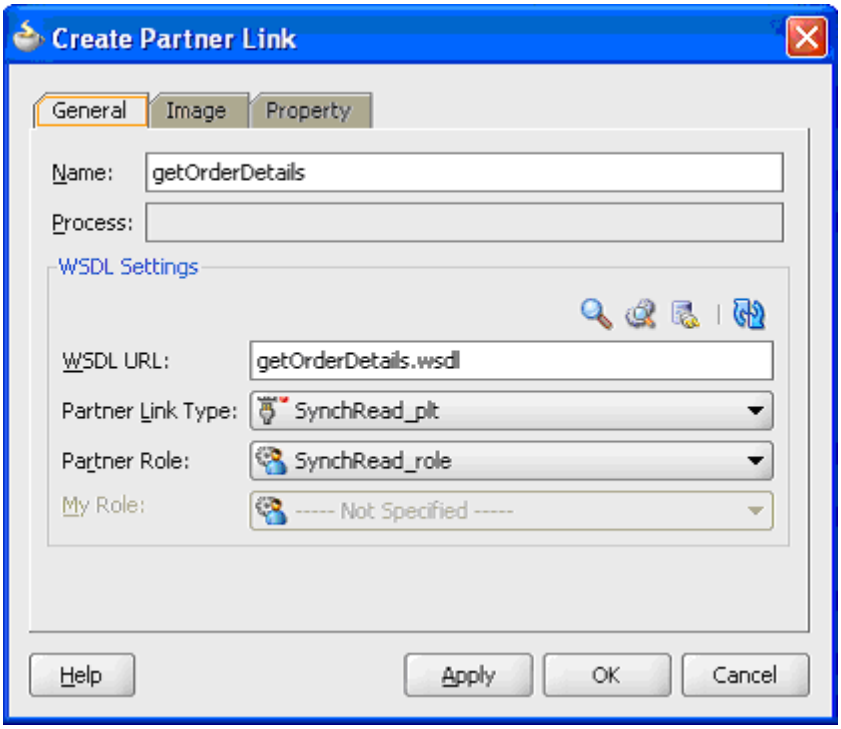
   The selected schema information will be automatically populated in the URL and Schema Element fields.

Specifying Message Schema



8. Click **Next** and then **Finish**. The wizard generates the WSDL file corresponding to the partner link. The main Create Partner Link dialog box appears, specifying the new WSDL file `getOrderDetails.wsdl`.

Completing the Partner Link Configuration



Click **Apply** and **OK** to complete the configuration and create the partner link with the required WSDL settings for the File Adapter service.

The `getOrderDetails` Partner Link appears in the BPEL process diagram.

Configuring the Invoke Activities

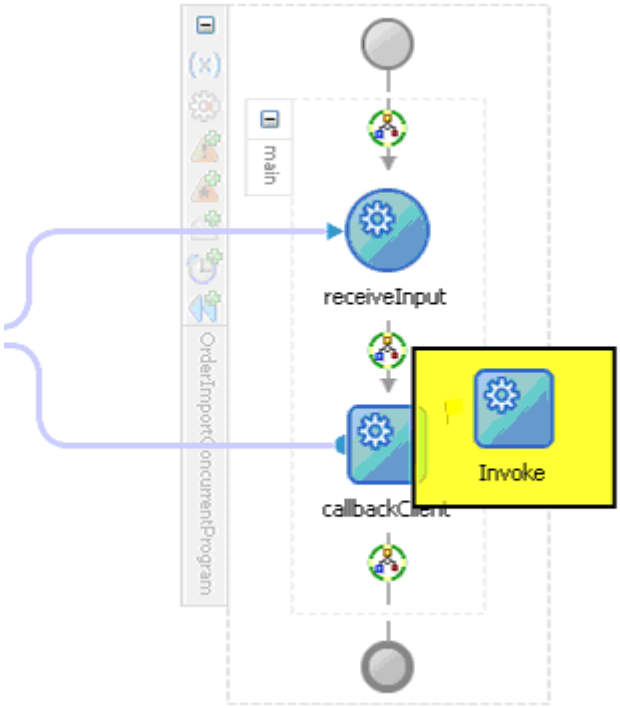
After configuring partner links, you need to configure the following two Invoke activities:

1. To get the acknowledgement data by invoking the `ViewAck` partner link.
2. To write acknowledgement data to an XML file by invoking `WriteAckdata` partner link for File Adapter.

To add the first Invoke activity for a partner link to get acknowledgement data:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette. Drag and drop the first **Invoke** activity into the center swim lane of the process diagram, between the **receiveInput** and **callbackClient** activities.

Adding an Invoke Activity



2. Link the Invoke activity to the `ViewAck` service. The Edit Invoke dialog appears.

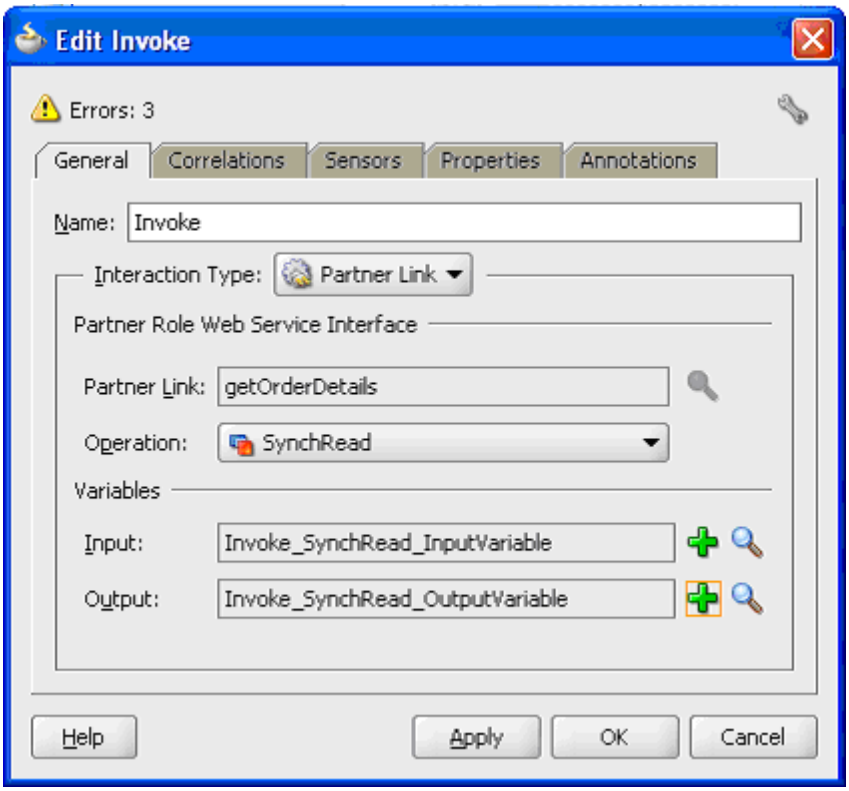
3. Enter a name for the Invoke activity, then click the **Create** icon next to the **Input Variable** field to create a new variable. The Create Variable dialog box appears.

4. Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK**.

5. Click the **Create** icon next to the **Output Variable** field to create a new variable. The Create Variable dialog box appears.

6. Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK**.

   Click **Apply** and **OK** in the Edit Invoke dialog box to finish configuring the Invoke activity.



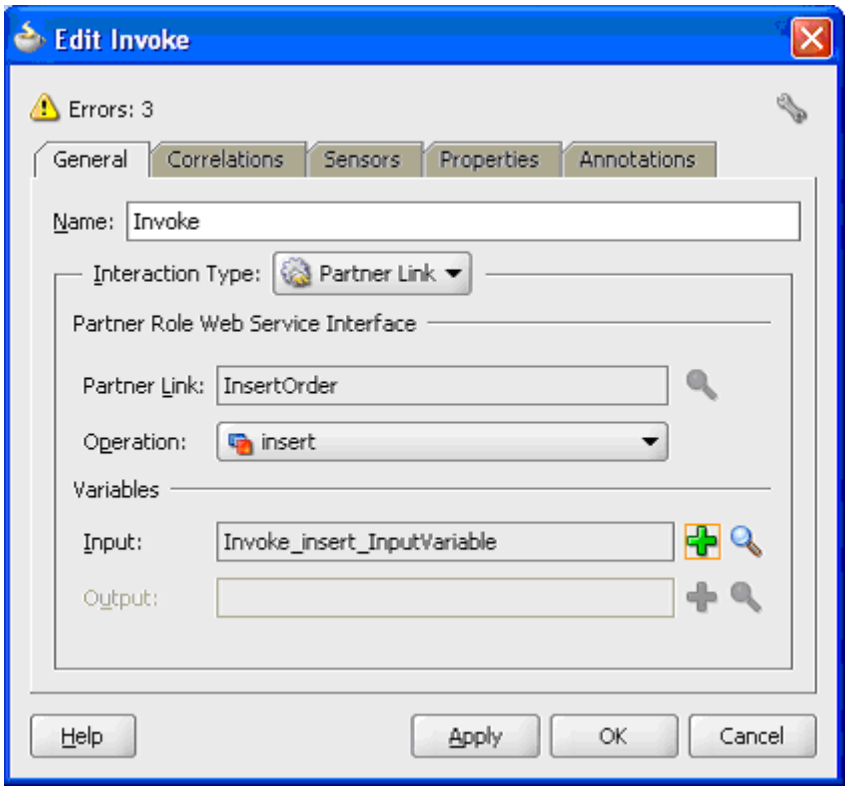
The Invoke activity appears in the process diagram.

To add the second Invoke activity for a partner link to insert order data into Open Interface tables:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette. Drag and drop the second **Invoke** activity into the center swim lane of the process diagram, between the first **Invoke** and **callbackClient** activities.

2. Link the Invoke activity to the `InsertOrder` service. The Edit Invoke dialog box appears.

3. Repeat Step 3 and Step 4 described in the first Invoke activity procedure. Click **Apply** and then **OK** in the Edit Invoke dialog box to finish configuring the second Invoke activity.



Configuring the Assign Activity

The next task is to add an Assign activity to the process map. This is used to pass the output of File Adapter's Synchronous Read (`getOrderDetails`) service as an input to the Open Interface`InsertOrder` service.

To add an Assign activity:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette.

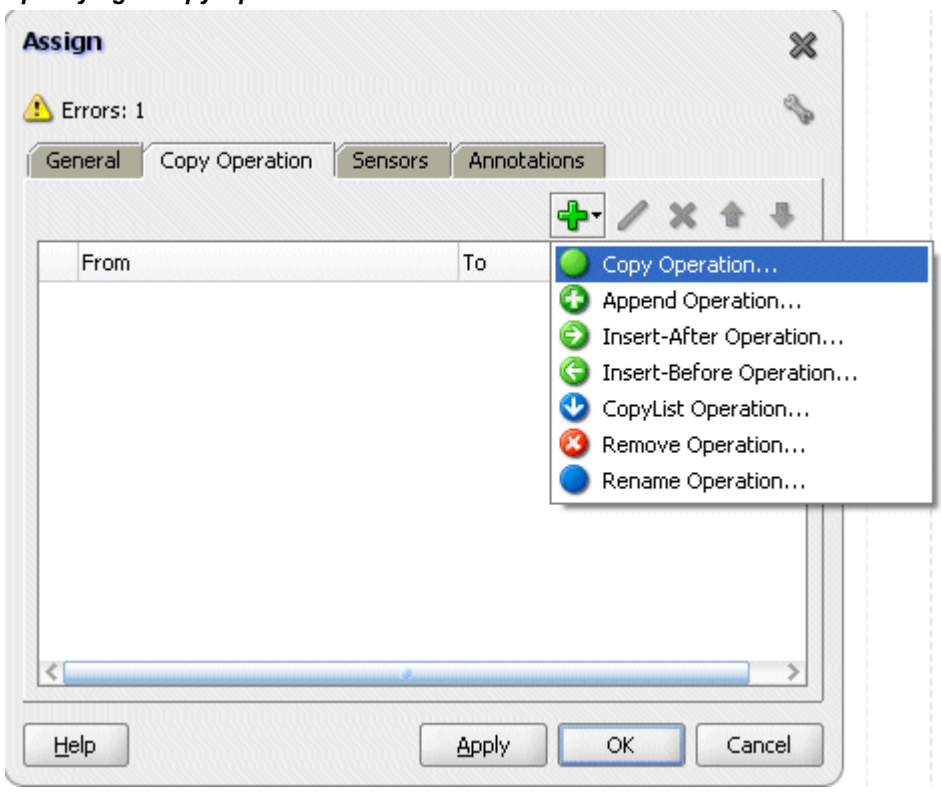
   Drag and drop the **Assign** activity into the center swim lane of the process diagram, between the first and second **Invoke** activities that you just created earlier.

2. Double-click the **Assign** activity to access the Edit Assign dialog.

   Click the General tab to enter a name for the Assign activity. For example, `SetOrderDetails`.

3. Select the Copy Operation tab, click the 'Plus' sign icon and select **Copy Operation** from the menu. The Create Copy Operation window appears.

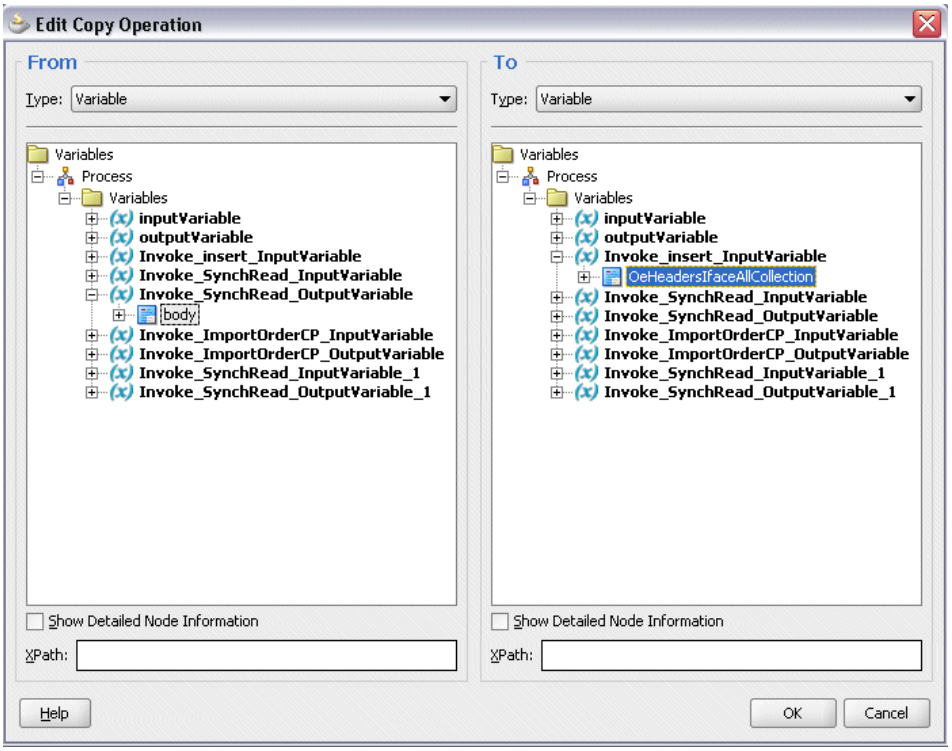
Specifying a Copy Operation Action



4. Enter the following parameter information:

   - In the From navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_SynchRead_OutputVariable** and select **body**.

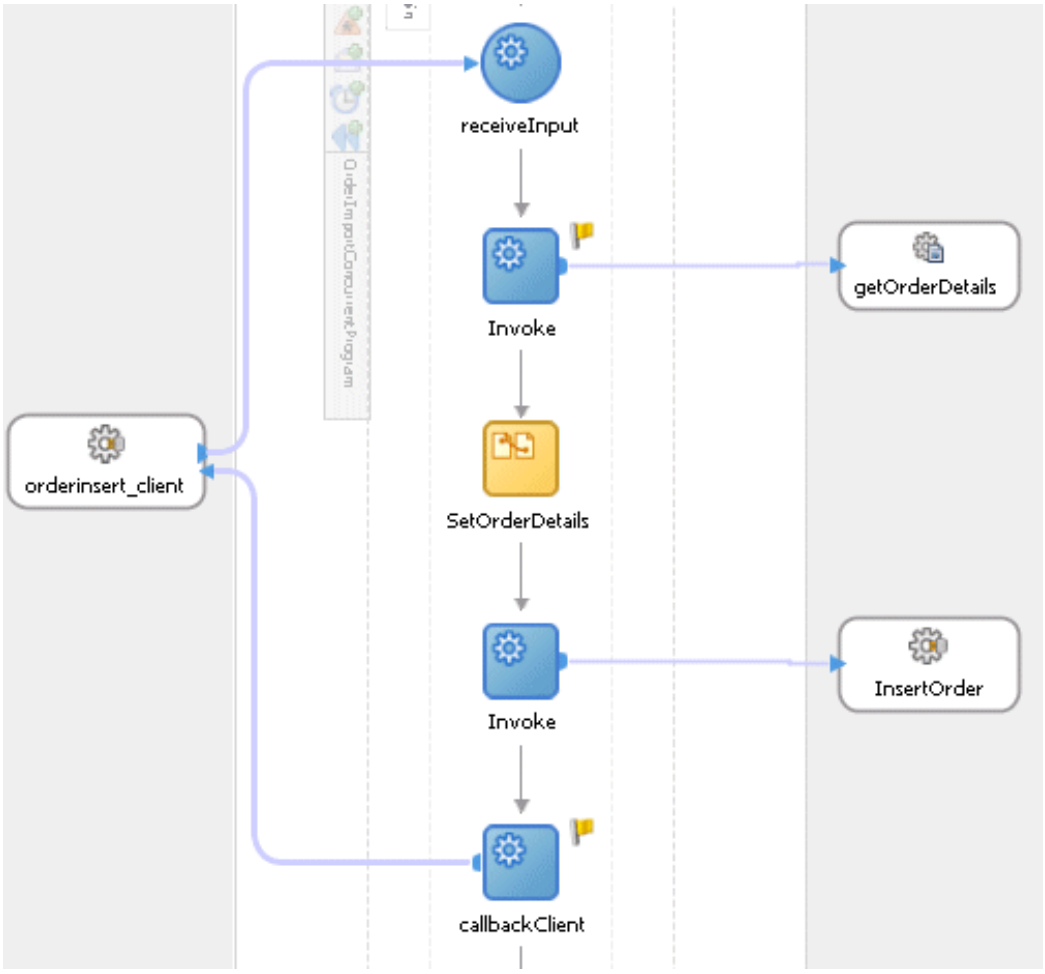
   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_insert_InputVariable** and select **OeHeadersIfaceAllCollection**.



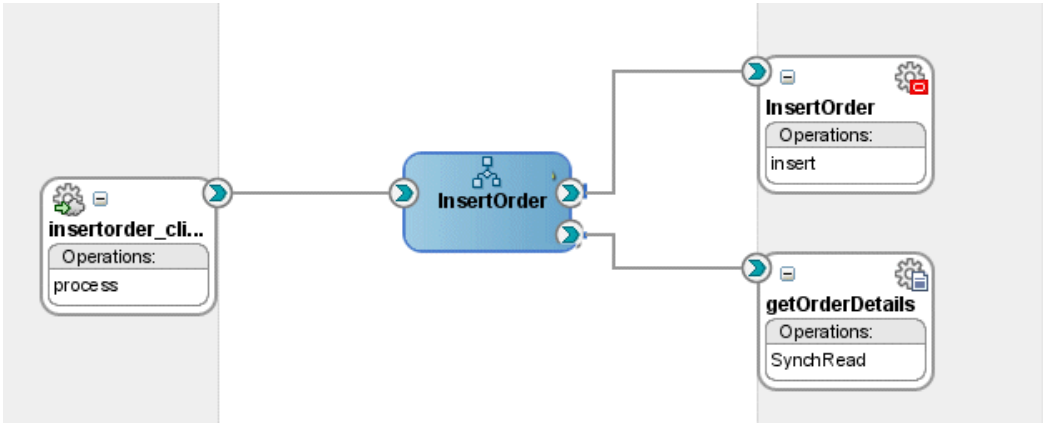
- Click **OK**. The Edit Assign dialog box appears.

5. Click **Apply** and then **OK** to complete the configuration of the Assign activity.

Completed BPEL Process Diagram



Click the `composite.xml` to display the Oracle JDeveloper composite diagram:



Note: Click the Source tab of `composite.xml` to enter a value for the physical directory `inputDir` for the reference `getOrderDetails` (such as `/usr/tmp`).

```
<property name="inputDir" type="xs:string"
many="false" override="may">/usr/tmp</property>
```

Run-Time Tasks for Interface Tables

After designing the BPEL process, the next step is to deploy, run, and monitor it.

1. Deploy the BPEL process., page 7-35
2. Test the BPEL process., page 7-36

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 application server (Oracle WebLogic Server) that you have established the connection.

Prerequisites

Before deploying the BPEL process using Oracle JDeveloper, you must ensure the following:

- You must have established the connectivity between the deign-time environment and an application server.

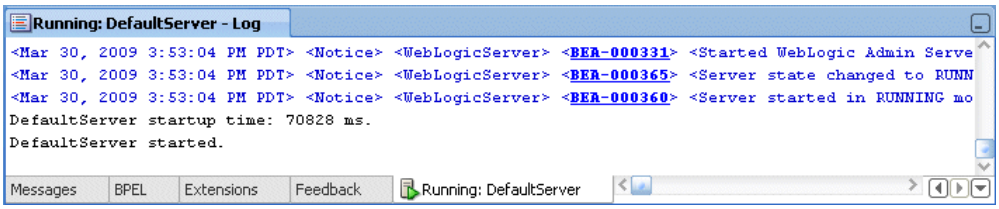
  For more information, see Configuring the Data Source in Oracle WebLogic Server, page A-3 and Creating an Application Server Connection, page A-8.

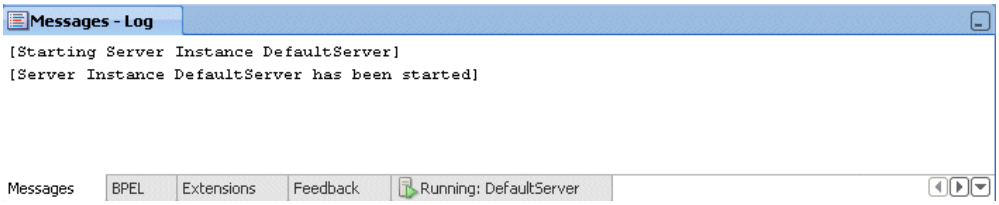
- Oracle WebLogic Server has been started.

Before deploying the BPEL process, you need to start the Oracle WebLogic Server that you have established the connection.

If a local instance of the WebLogic Server is used, start the WebLogic Server by selecting **Run > Start Server Instance** from Oracle JDeveloper.

Once the WebLogic Admin Server "DefaultServer" instance is successfully started, you should find that <Server started in Running mode> and DefaultServer started message in the Running:DefaultServer and Messages logs.





To deploy the BPEL process:

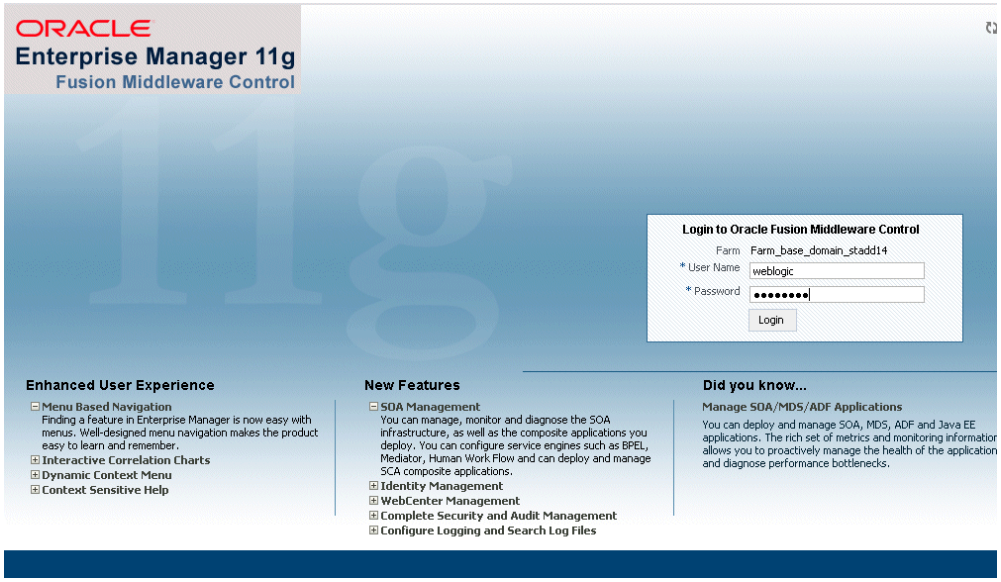
1. Select the BPEL project in the Applications Navigator.
2. Right-click the project name. Select **Deploy** from the menu that appears.
3. Right-click the project name, and then select **Deploy > [project name] > [serverConnection]** from the menu that appears.
4. The BPEL process is compiled and deployed. You can check the progress in the Messages window.

Testing the BPEL Process

Once the BPEL process is deployed, you can manage and monitor the process from the Oracle Enterprise Manager Fusion Middleware Control Console. You can also test the process and the integration interface by manually initiating the process.

To manually initiate and monitor the BPEL process:

1. Navigate to Oracle Enterprise Manager Fusion Middleware Control Console (`http://<servername>:<portnumber>/em`). The composite you deployed is displayed in the Applications Navigation tree.



2. Enter username (such as `weblogic`) and password and click **Login** to log in to a farm.

   You may need to select an appropriate target instance farm if there are multiple target Oracle Enterprise Manager Fusion Middleware Control Console farms.

3. From the Farm base domain, expand the **SOA** >**soa-infra** to navigate through the SOA Infrastructure home page and menu to access your deployed SOA composite applications running in the SOA Infrastructure for that managed server.

   > **Note:** The Farm menu always displays at the top of the navigator. As you expand the SOA folder in the navigator and click the links displayed beneath it, the SOA Infrastructure menu becomes available at the top of the page.

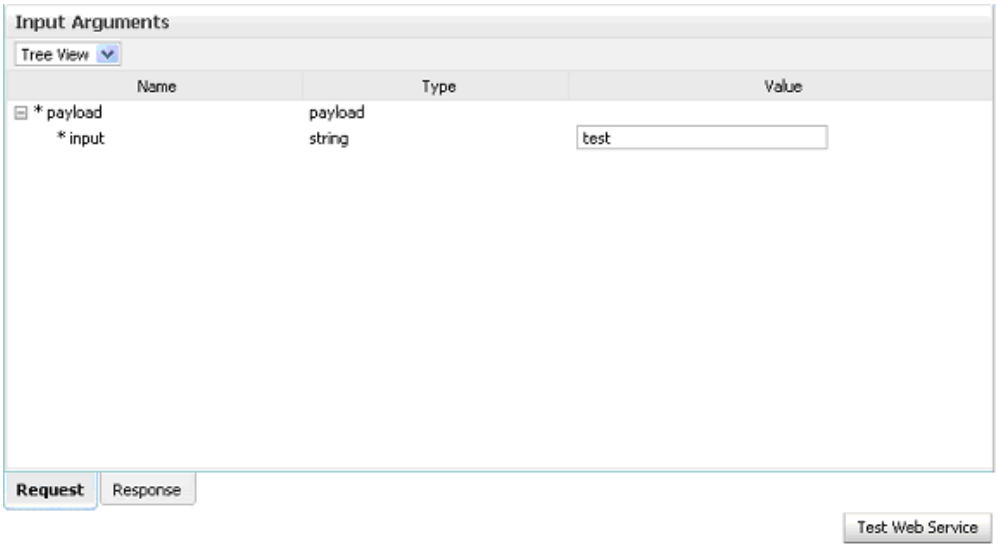
   Click the SOA composite application that you want to initiate (such as 'OrderInsert') from the SOA Infrastructure.

   Click **Test** at the top of the page.

4. The Test Web Service page for initiating an instance appears. You can specify the XML payload data to use in the Input Arguments section.

   Enter the input string required by the process and click **Test Web Service** to initiate the process.

Testing Web Service



The test results appear in the Response tab upon completion.

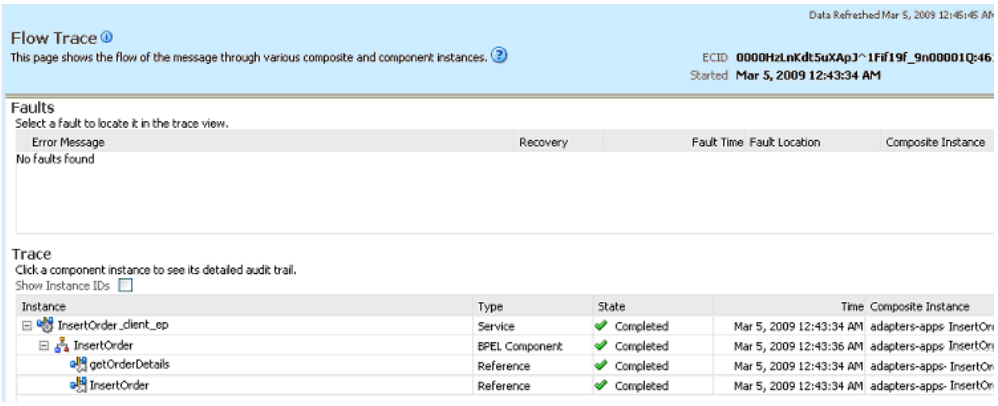
5. Click on the BPEL process name and then click the Instances tab. The SOA composite application instance ID, name, conversation ID, most recent known state of each instance since the last data refresh of the page are displayed.

   In the Instance ID column, click a specific instance ID to show the message flow through the various service components and binding components. The Flow Trace page is displayed.

   In the Trace section, you should find the sequence of the message flow for the service binding component (`insertorder_client_ep`), BPEL component (`InsertOrder`), and reference binding components (`getOrderDetails` and `InsertOrder`). All involved components have successfully received and processed messages.

   If any error occurred during the test, you should find it in the Faults section.

Flow Trace Page



6. Click your BPEL service component instance link (such as `InsertOrder`) to display the Instances page where you can view execution details for the BPEL activities in the Audit Trail tab.

   Click the Flow tab to check the BPEL process flow diagram. Click an activity of the process diagram to view the activity details and flow of the payload through the process.

7. **Validating Records Using SQL**

   To confirm that the records have been inserted into the selected Open Interface tables, you can write SQL `SELECT` statements and fetch the results showing the latest records inserted into the open interface tables.

Design-Time Tasks for Views

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

BPEL Process Scenario

In this example, Adapter for Oracle Applications retrieves purchase order acknowledgement information from Interface Views `OE_HEADER_ACKS_V` and `OE_LINE_ACKS_V`. The acknowledgement information will then be written to an XML file as an output for confirmation.

If the BPEL process is successfully executed after deployment, you can validate the output XML file with the same order book reference ID once an order is approved.

Prerequisites to Configure Views

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

BPEL Process Flow

Based on the scenario, the following design-time tasks are discussed in this chapter.

1. Create a new BPEL project., page 7-40
2. Add a partner link., page 7-44
3. Add a partner link for File Adapter., page 7-58
4. Configure the Invoke activity., page 7-64
5. Configure the Assign activity., page 7-68

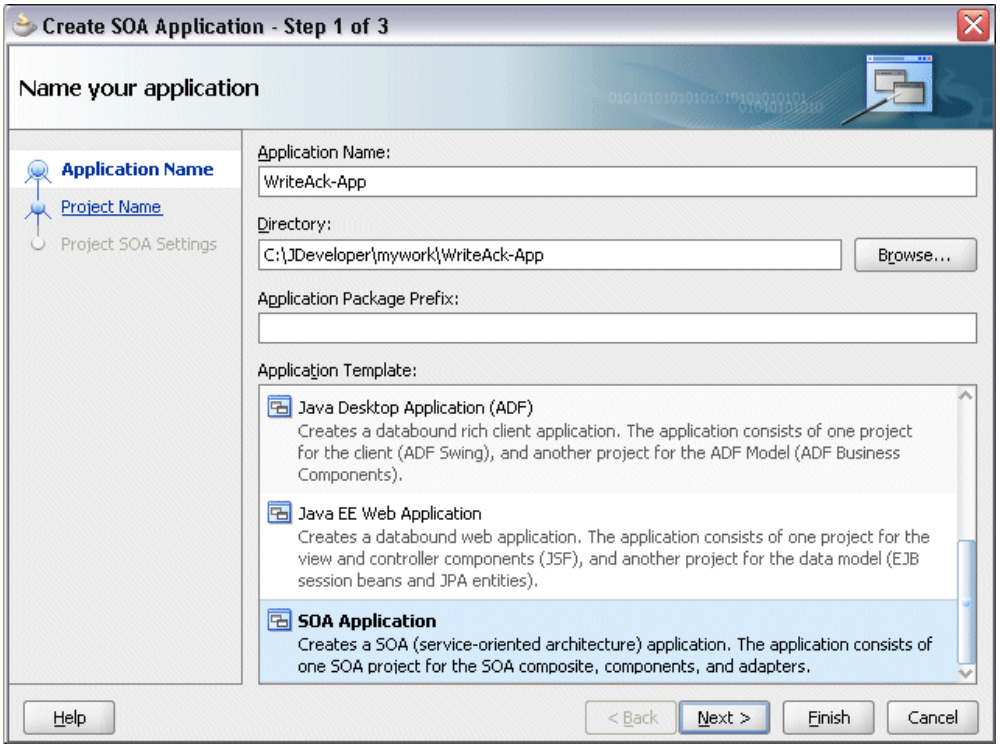
Creating a New BPEL Project

To create a new BPEL project:

1. Open JDeveloper BPEL Designer. Click **New Application** in the Application Navigator.

   The Create SOA Application - Name your application page is displayed.

The Create SOA Application - Name your application Page

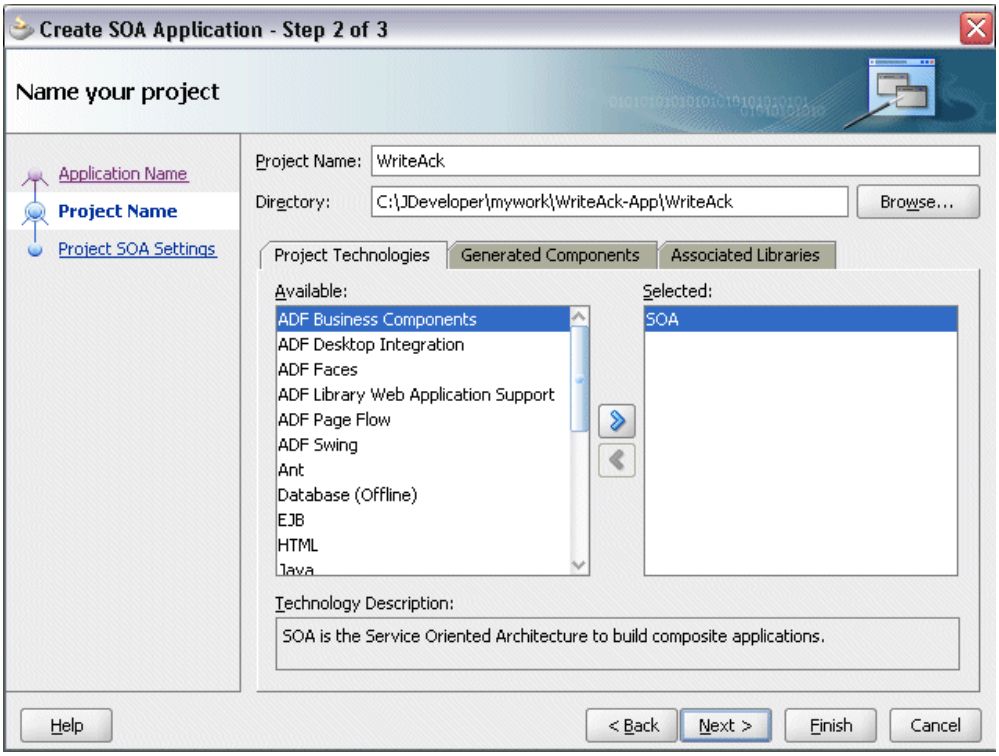


2. Enter an appropriate name for the application in the **Application Name** field and select **SOA Application** from the Application Template list.

   Click **Next**. The Create SOA Application - Name your project page is displayed.

3. Enter an appropriate name for the project in the **Project Name** field. For example, `WriteAck`.

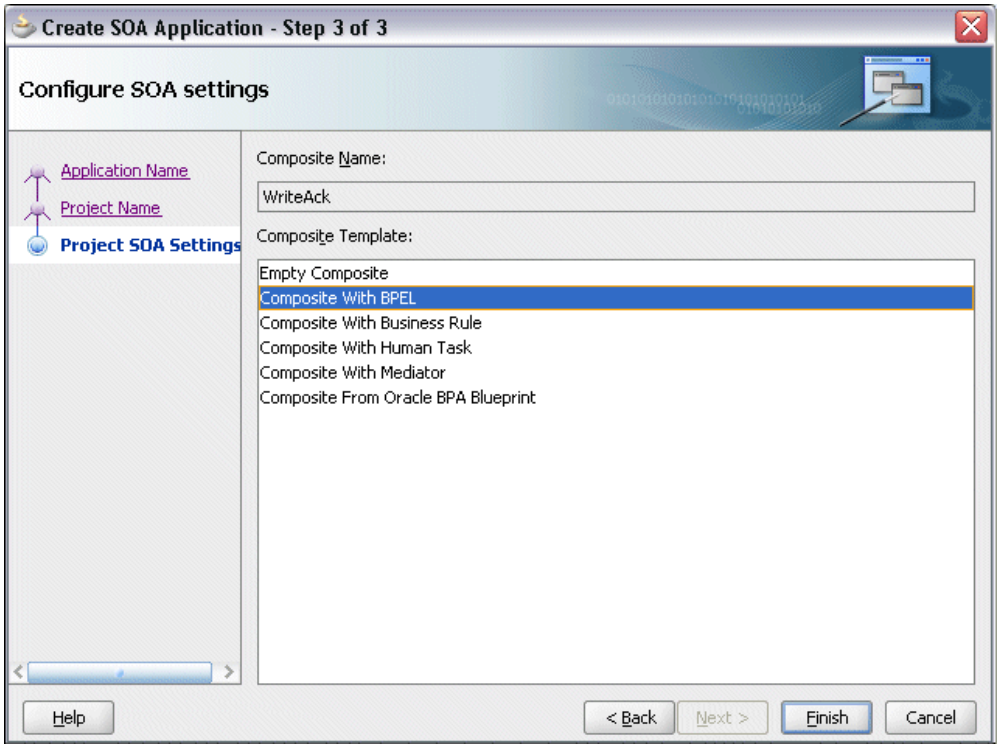
The Create SOA Application - Name your project Page



4. In the Project Technologies tab, ensure that **SOA** is selected from the Available technology list to the Selected technology list.

   Click **Next**. The Create SOA Application - Configure SOA settings page is displayed.

The Create SOA Application - Configure SOA settings Page

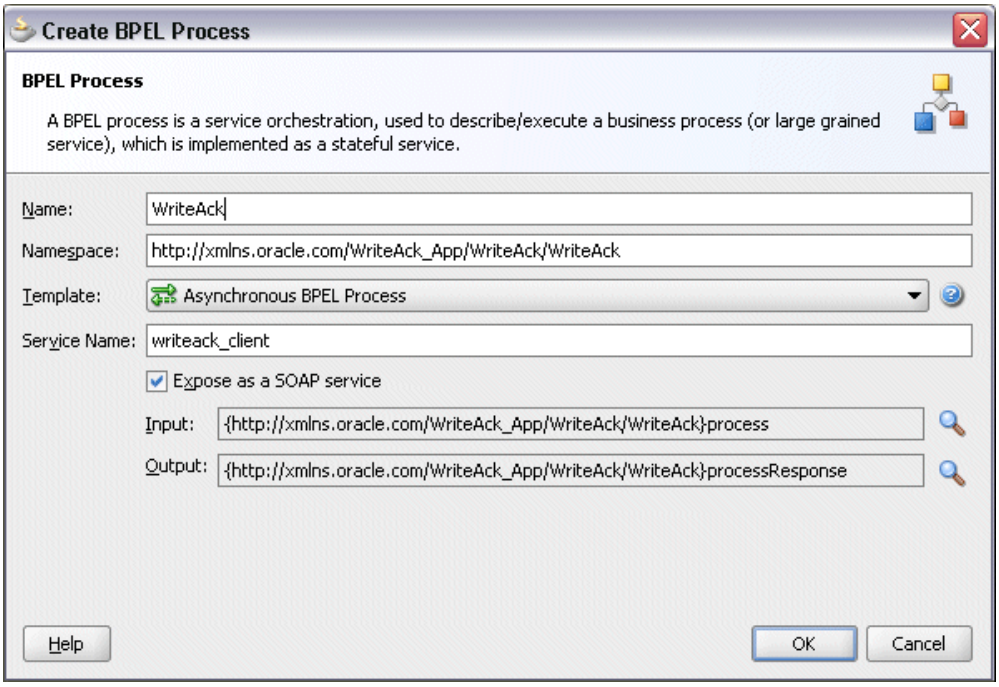


5. In the **BPEL Process Name** field, enter a descriptive name. For example, `WriteAck`.

6. Select **Composite With BPEL** from the Composite Template list, and then click **Finish**. You have created a new application, and an SOA project. This automatically creates an SOA composite.

   The Create BPEL Process page is displayed.

The Create BPEL Process Page



7. Enter an appropriate name for the BEPL process in the **Name** field. For example, `WriteAck`.

   Select **Asynchronous BPEL Process** in the **Template** field. Click **OK**.

   An asynchronous BPEL process is created with the Receive and Reply activities. The required source files including bpel and wsdl, using the name you specified (for example, `WriteAck.bpel` and `WriteAck.wsdl`) and `composite.xml` are also generated.

Adding a Partner Link

A BPEL partner link defines the link name, type, and the role of the BPEL process that interacts with the partner service.

In the scenario described earlier, you need to add a partner link to retrieve order acknowledgement information from selected Open Interface Views.

To add a partner link:

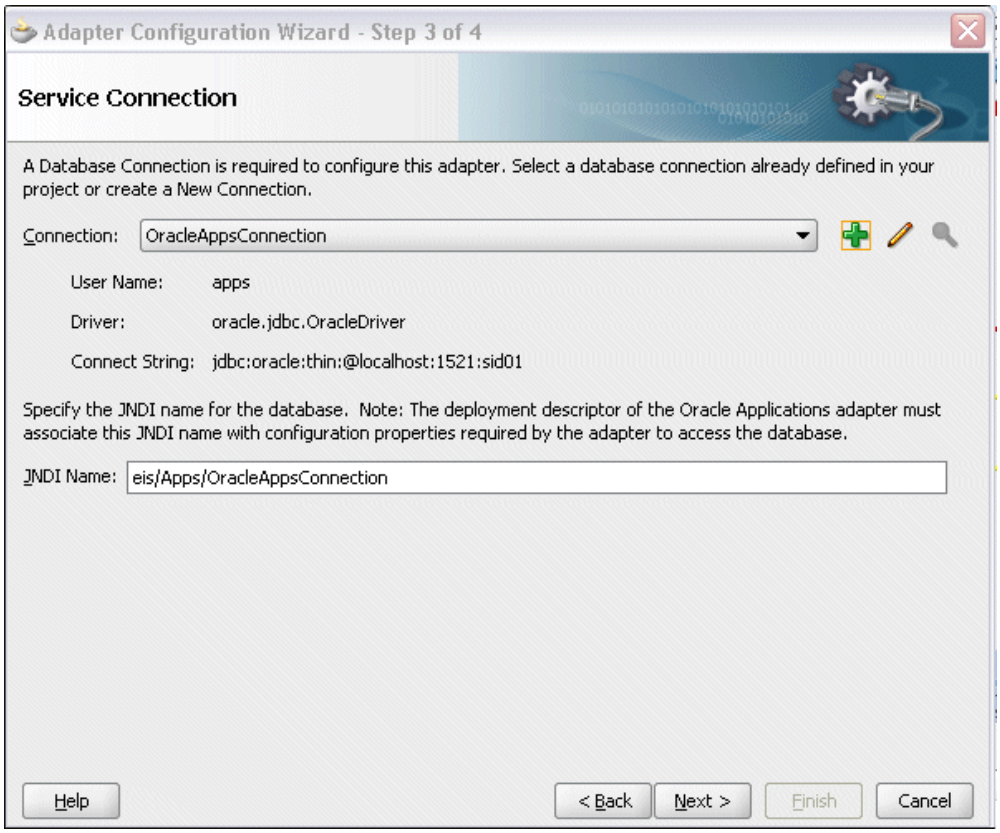
1. Click **BPEL Services** in the Component palette.

   Drag and drop **Oracle Applications** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears. Click **Next**.

2. Enter a service name in the **Service Name** field. For example, `ViewAck`.

   Click **Next**. The Service Connection dialog appears.

Specifying a Database Service Connection



3. You can perform either one of the following options for your database connection:

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

   - You can create a new database connection by clicking the **Create a New Database Connection** icon.

     How to define a new database connection, see Create a New Database Connection, page 4-13.

   - You can select an existing database connection that you have configured earlier from the **Connection** drop-down list.

     The Service Connection page will be displayed with the selected connection information. The JNDI (Java Naming and Directory Interface) name

corresponding to the database connection appears automatically in the **Database Server JNDI Name** field. Alternatively, you can specify a JNDI name.

> **Note:** When you specify a JNDI name, the deployment descriptor of the Oracle Applications adapter must associate this JNDI name with configuration properties required by the adapter to access the database.

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.

> **Note:** For more information about JNDI concepts, refer to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

4. Click **Next** in the Service Connection dialog box. You can add an interface view by browsing through the list of open interface views available in Oracle Applications.

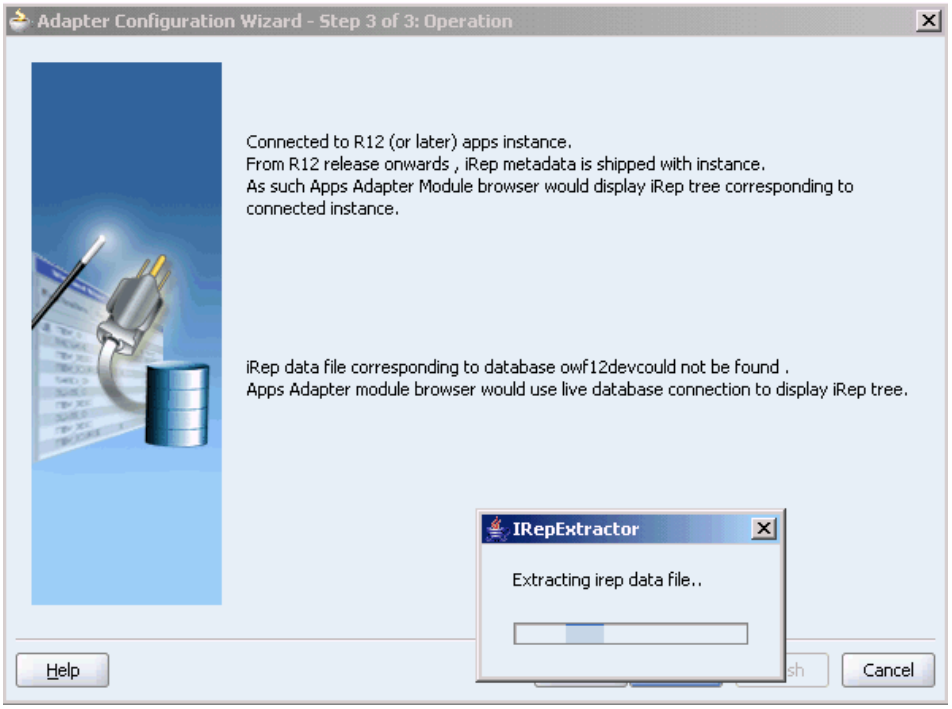
   For Oracle E-Business Suite Release 12:

   If you are connecting to Oracle E-Business Suite Release 12, then the **IREP File not present** dialog box appears indicating that Adapter could not find the Oracle Integration Repository data file corresponding to the database you are connecting in your workspace. Absence of the data file would make browsing or searching of Integration Repository tree considerably slow. You can choose to extract the data file and create a local copy of the Integration Repository data file. Once it is created successfully, Adapter will pick it up automatically next time and retrieve data from your local Integration Repository.

   You can select one of the following options:

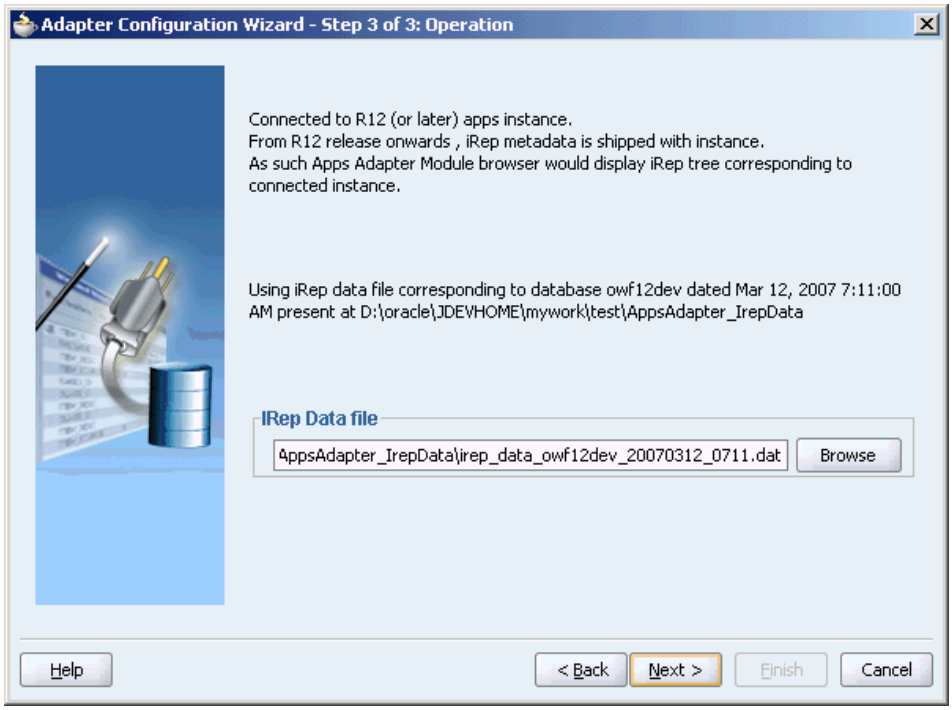
   - Click **Yes** to extract the Integration Repository data file.

Extracting Integration Repository Data File



After the system successfully creates a local copy of the Integration Repository data file, next time when you connect to the database, you will find the **IRep Data File** field appears in the Operation dialog box indicating where your local copy exists with the creation date and time as part of the file name.

Using the Local Integration Repository Data File



- Click **No** to query the Integration Repository data file from the live database you are connecting to display the Integration Repository tree.

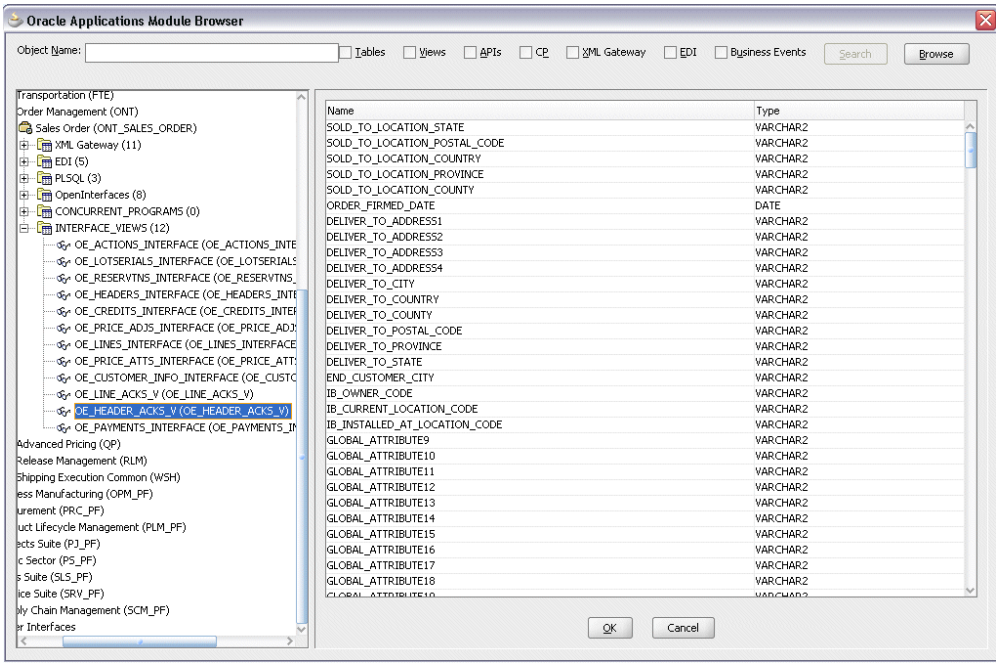
  Note: It is highly recommended that you create a local copy of the Integration Repository data file so that Adapter will query the data next time from the local copy in your workspace to enhance the performance.

For Oracle E-Business Suite pre-Release 11.5.10:

If you are connecting to a pre-11.5.10 Oracle Applications instance, you must select the interface type in the Adapter Configuration Wizard. Select **Tables/Views/APIs/Concurrent Programs** to proceed.

5. Click **Get Object** to open the Oracle Applications Module Browser.

Selecting a view from the Module Browser



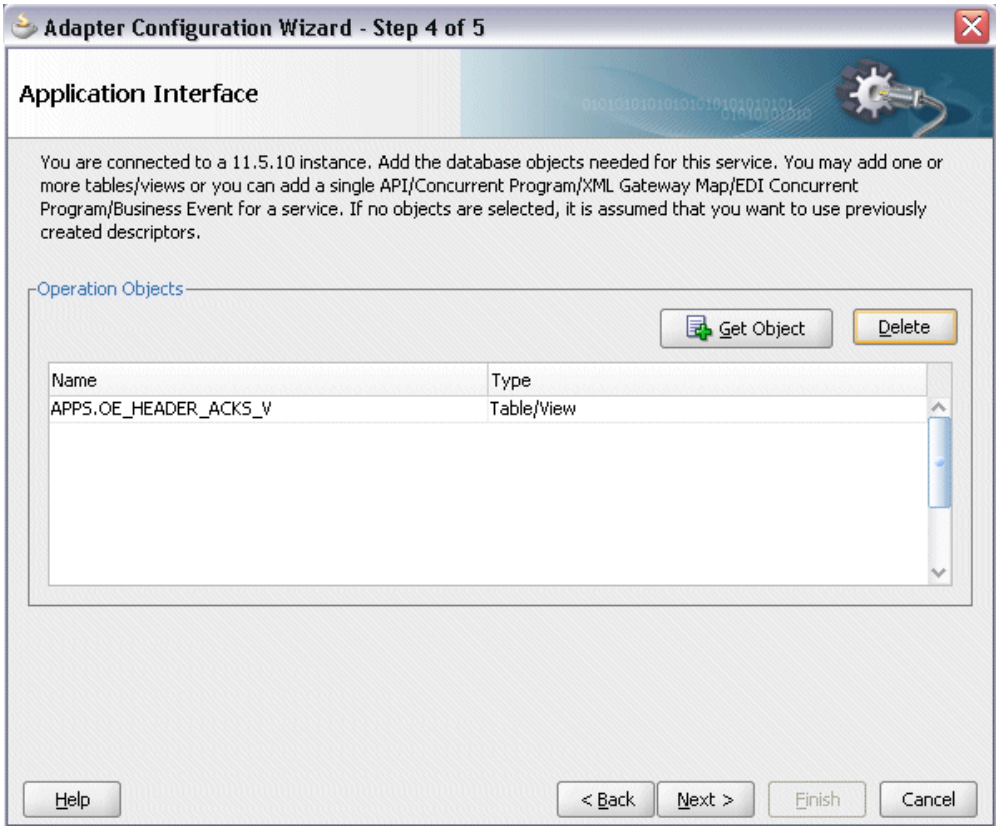
Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. For example, Applications Technology or Order Management Suite are product families in Oracle Applications. The product families contain the individual products. For example, Order Management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

6. Navigate to *Order Management Suite (OM_PF) > Order Management (ONT) > Sales Order (ONT_SALES_ORDER) > Interface_Views* to select `OE_HEADER_ACKS_V`.

   Click **OK**. The Application Interface page appears with the selected interface view.

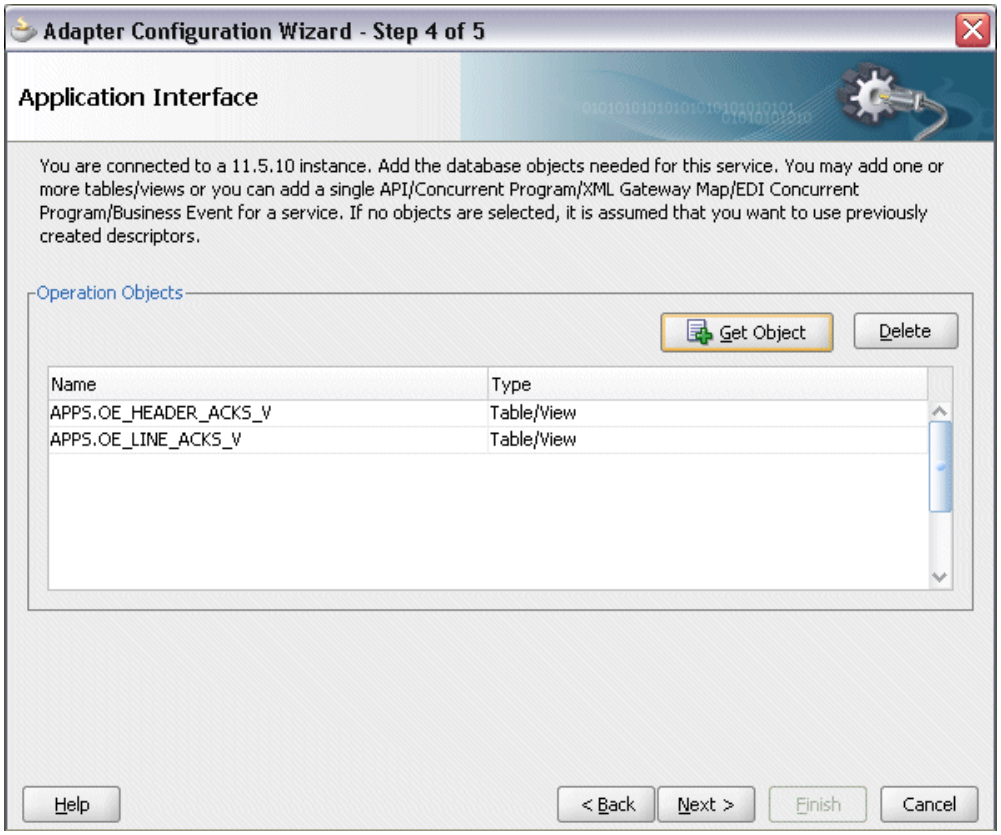
Adapter Configuration Wizard - Application Interface Page



7. Click **Get Object** to open the Oracle Applications Module Browser again to select another interface view`OE_LINE_ACKS_V` using the same navigation path *Order Management Suite (OM_PF) > Order Management (ONT) > Sales Order (ONT_SALES_ORDER) > Interface_Views*.

   Click **OK**. The Application Interface page appears with the two selected views.

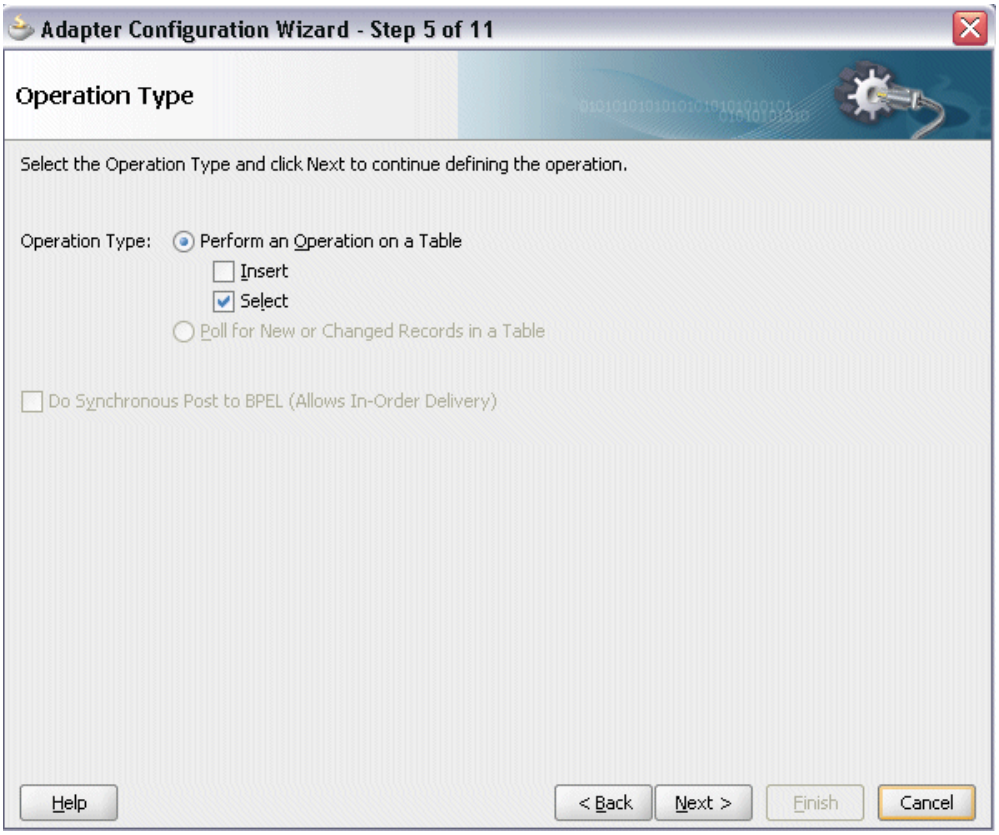
Adapter Configuration Wizard - Application Interface Page



Click **OK**.

8. Click **Next**. The Operation Type page is displayed.

Adapter Configuration Wizard - Operation Type Page

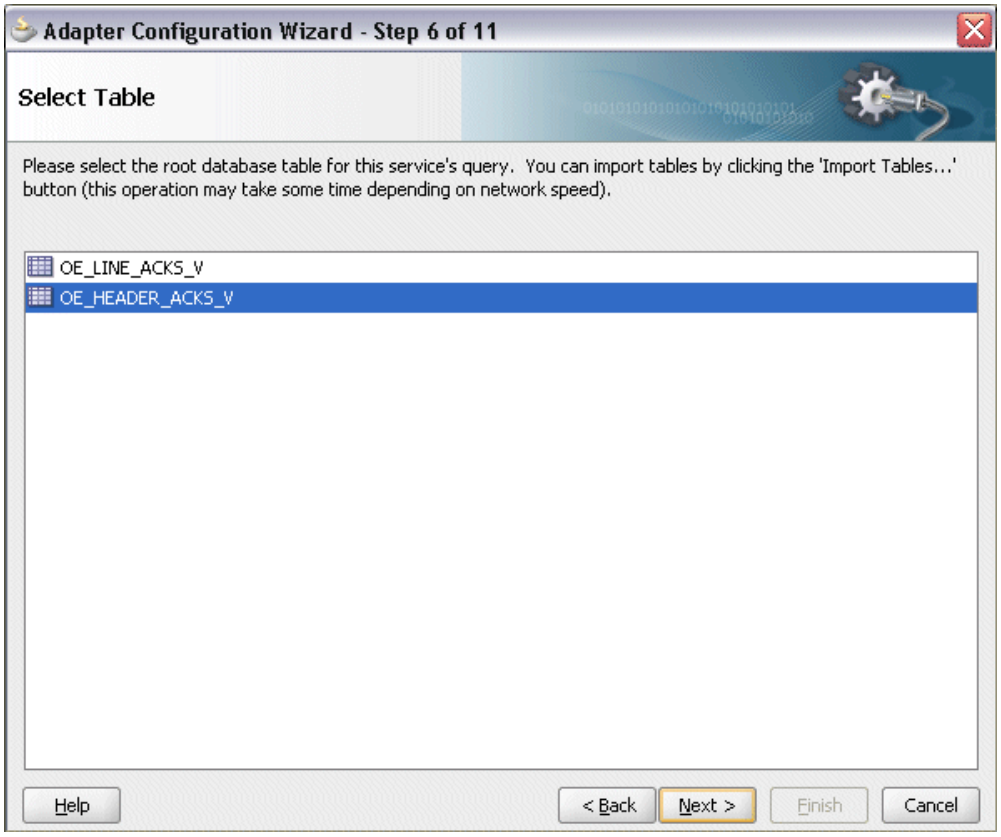


Select the **Perform an Operation on a Table** radio button and then choose the **Select** check box.

> **Note:** Interface views can be used only for Select operations.

9. Click **Next**. The Select Table page is displayed.

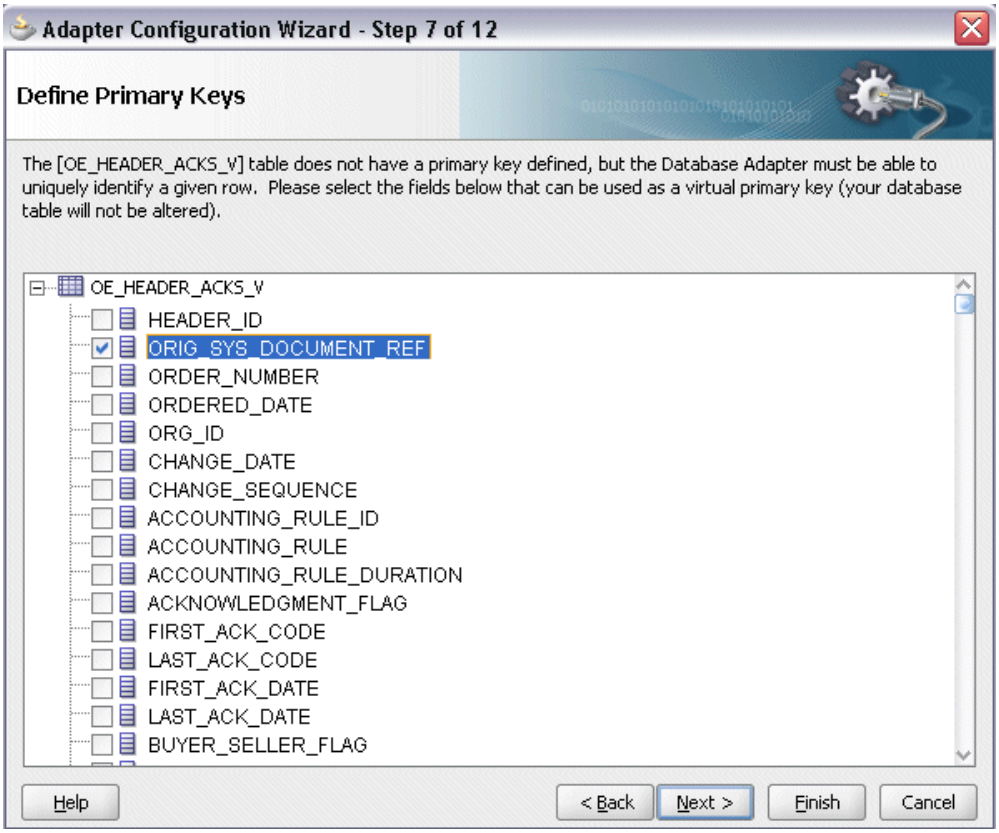
Adapter Configuration Wizard - Select Table Page



Select OE_HEADERS_ACKS_V as the root database table for this service's query.

10. Click **Next**. The Define Primary Keys page is displayed.

Adapter Configuration Wizard - Define Primary Keys Page for OE_HEADERS_ACKS_V



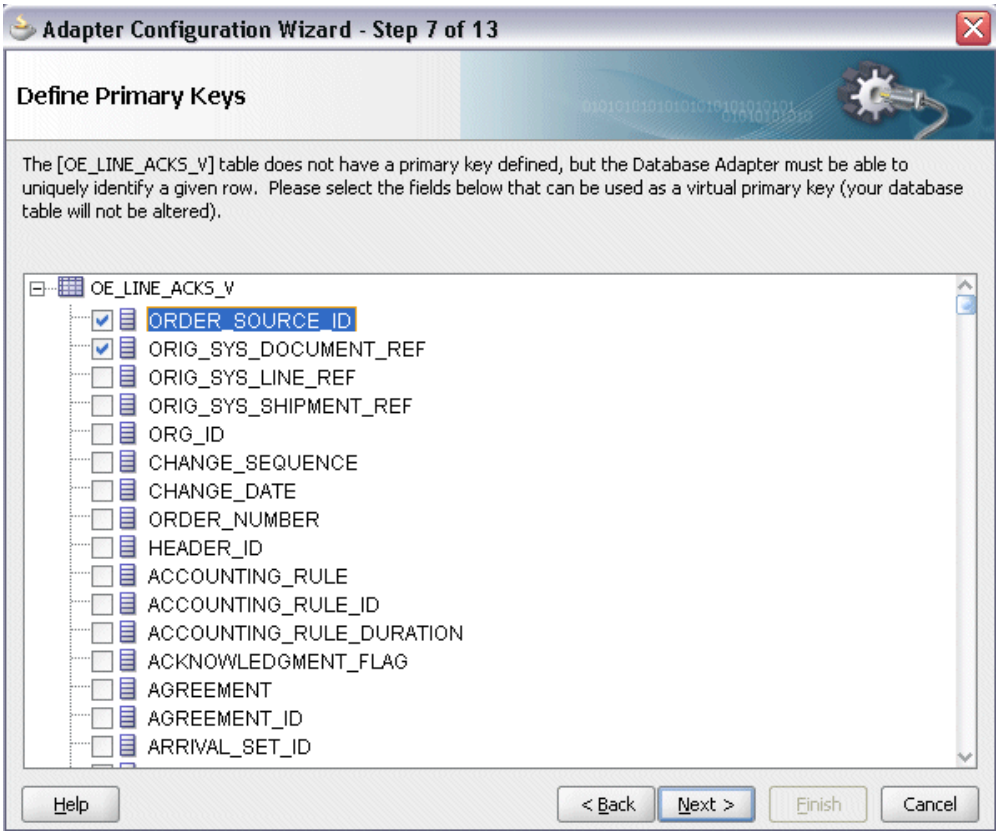
Select the following primary keys for the OE_HEADERS_ACKS_V table:

- ORIG_SYS_DOCUMENT_REF
- ORDER_SOURCE_ID

Click **Next**.

Select the same primary keys for the OE_LINE_ACKS_V table.

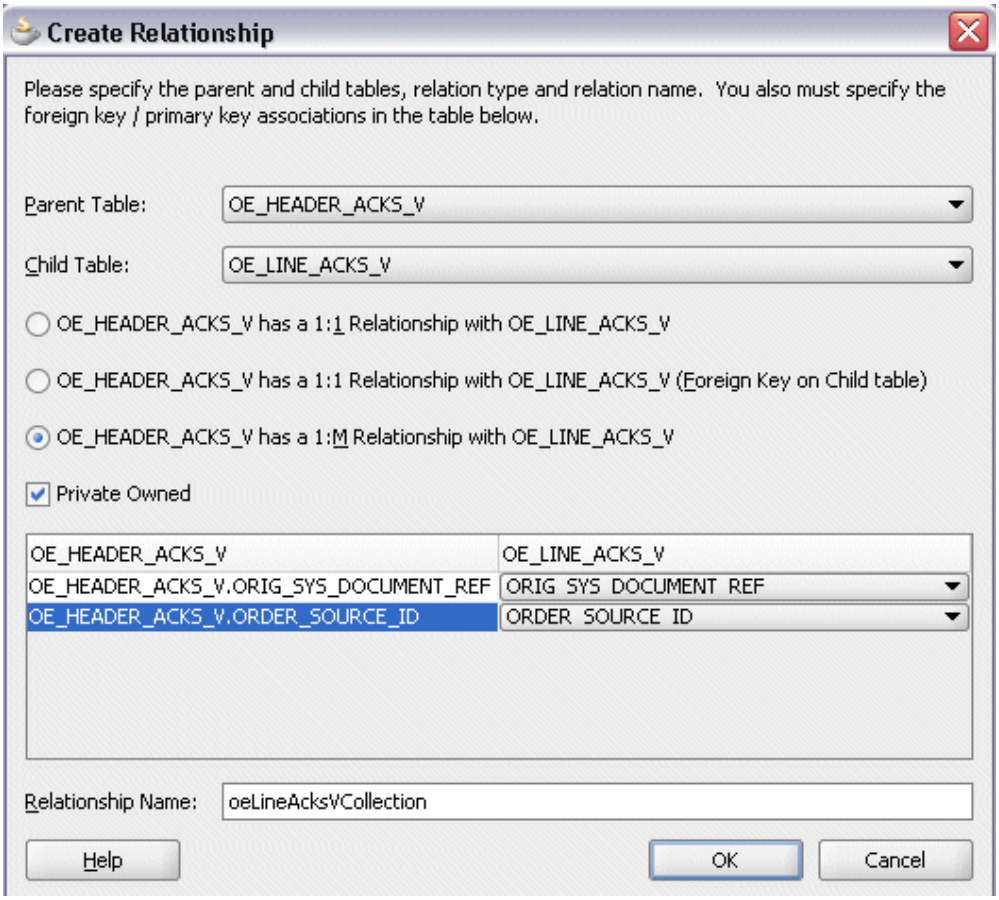
Adapter Configuration Wizard - Define Primary Keys Page for OE_LINE_ACKS_V



11. Click **Next**. The Relationships page appears.

    Click **Create** to open the Create Relationship dialog.

Defining Relationships



Enter the following information to define the relationship between the header and the detail table:

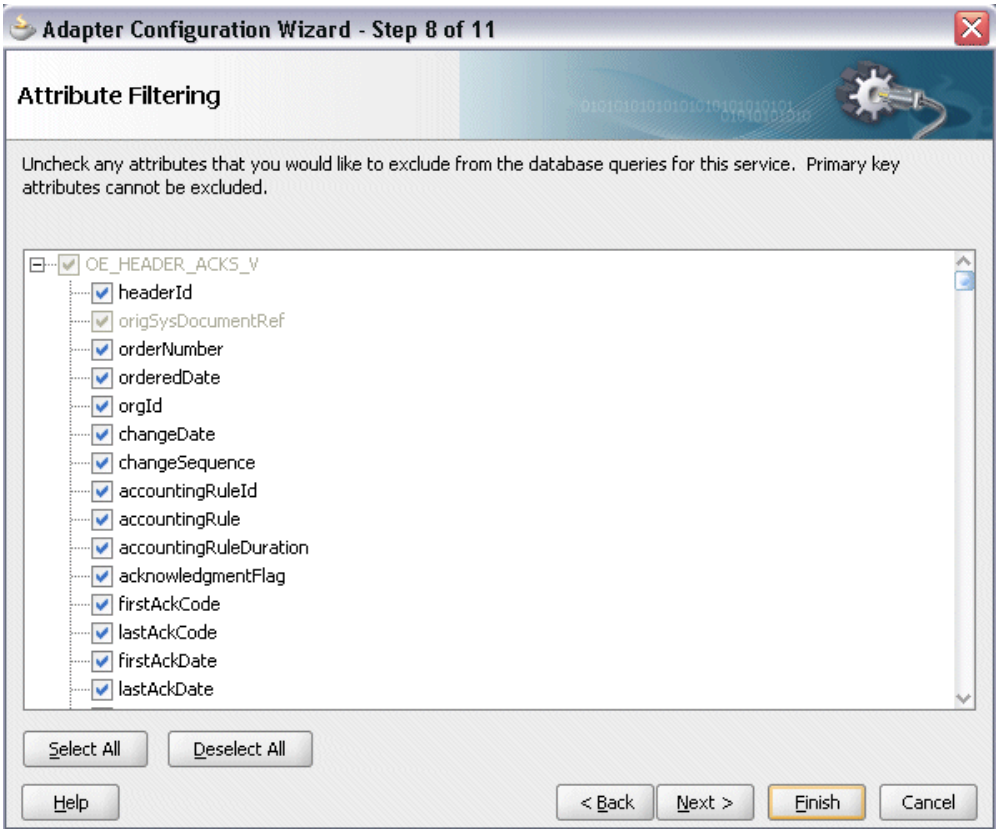
- Select the OE_HEADERS_ACKS_V as the parent table and OE_LINES_ACKS_V as the child table.
- Select the mapping type: OE_HEADERS_ACKS_V has a 1:M Relationship with OE_LINES_ACKS_V

> **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.

- Select the **Private Owned** check box.
- Associate the foreign key fields with the primary key fields:
  - OE_HEADERS_ACKS_V.ORDER_SOURCE_ID: ORDER_SOURCE_ID
  - OE_HEADERS_ACKS_V.ORIG_SYS_DOCUMENT_REF: ORIG_SYS_DOCUMENT_REF
- The Relationship Name field is populated automatically by default. You can optionally specify a new name for the relationship you are creating.

12. Click **Next**. The Attribute Filtering page appears. Leave default selections unchanged.

Adapter Configuration Wizard - Attribute Filtering Page



Click **Next**. The Define Selection Criteria page appears.

Click **Add** to add a new parameter if necessary. Click **Edit** to use the graphical expression builder to create the expression. You can define your own custom Select SQL by modifying the predefined SQL string.

13. Click **Next**. The Advanced Options page appears.

    Click **Next**.

14. Click **Finish**. The wizard generates the WSDL file corresponding to the selected interface. This WSDL file is now available for the partner link.

    Click **Apply** and then **OK**. The partner link is created with the required WSDL settings.

Adding a Partner Link for File Adapter

Use this step to configure a BPEL process by adding a partner link for File Adapter. This allows the acknowledgement data to be written to an XML file.

To add a partner link for the file adapter:

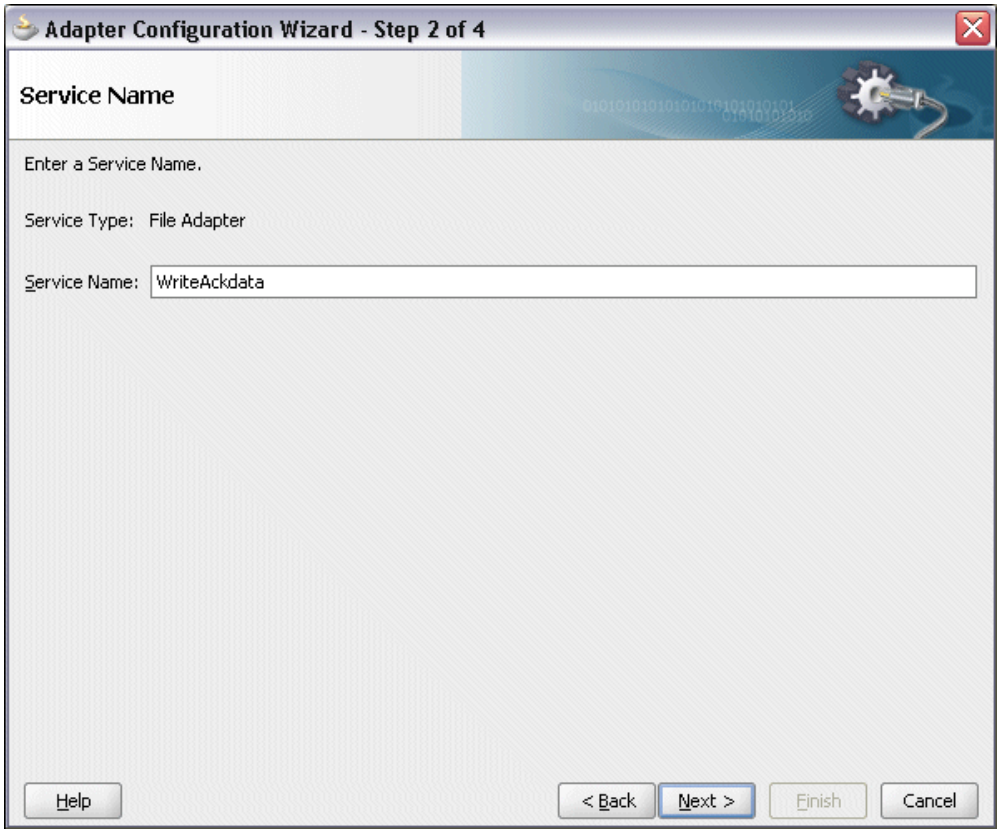
1. In JDeveloper BPEL Designer, click **BPEL Services** in the Component palette.

   Drag and drop **File Adapter** from the BPEL Services list into the right Partner Link swim lane of the process diagram right after the partner link you just created. The Adapter Configuration Wizard Welcome page appears.

   Click **Next**.

2. In the Service Name page, enter a name for the file adapter service, such as `WriteAckdata`.

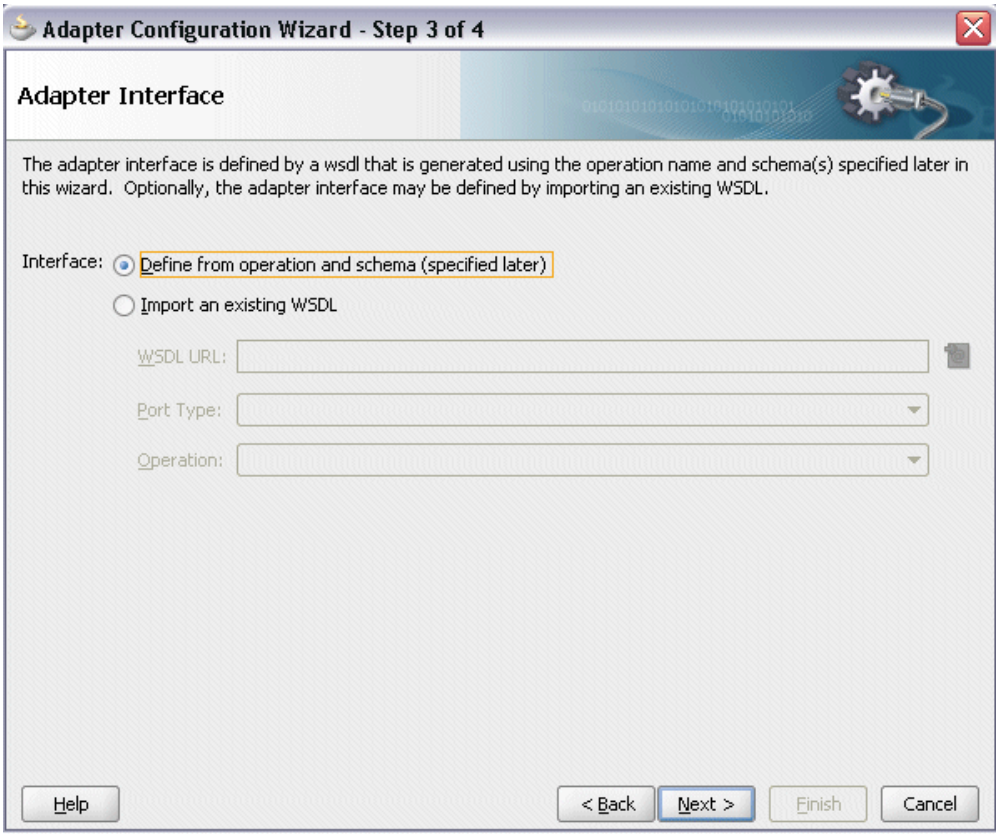
   Specifying the Service Name



3. Enter a name for the file adapter service, such as `WriteAckData`.

4. Click **Next**. The Adapter Interface page appears.

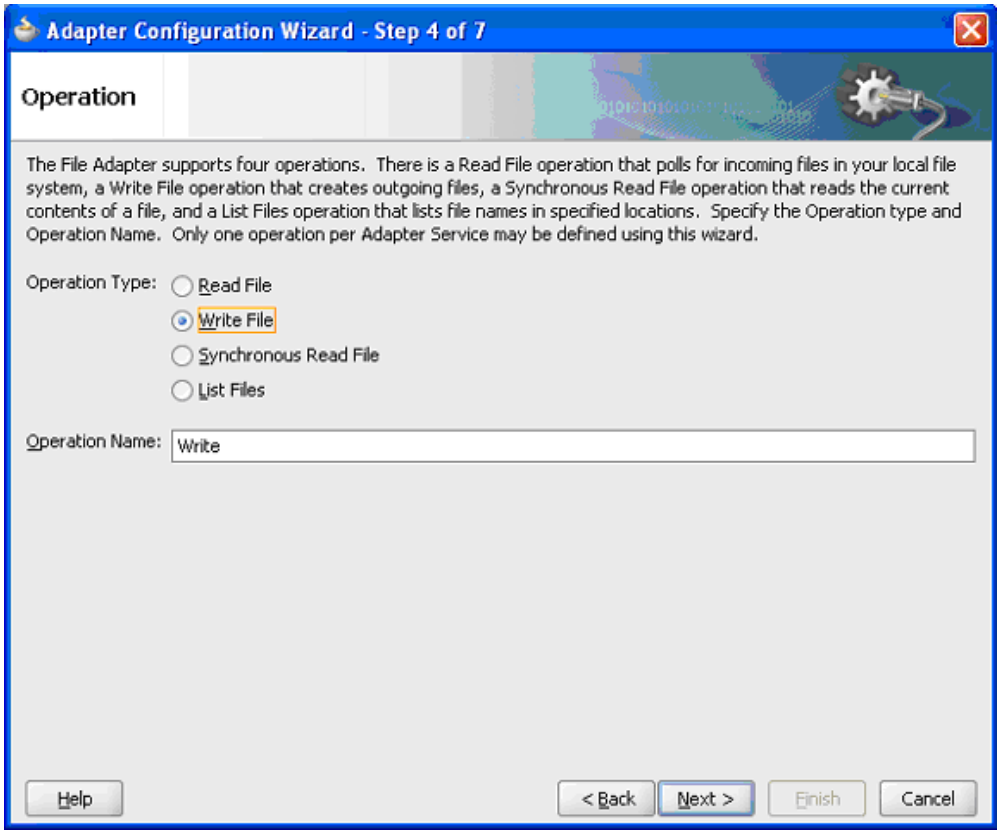
Specifying the Adapter Interface



Select the **Define from operation and schema (specified later)** radio button and click **Next**.

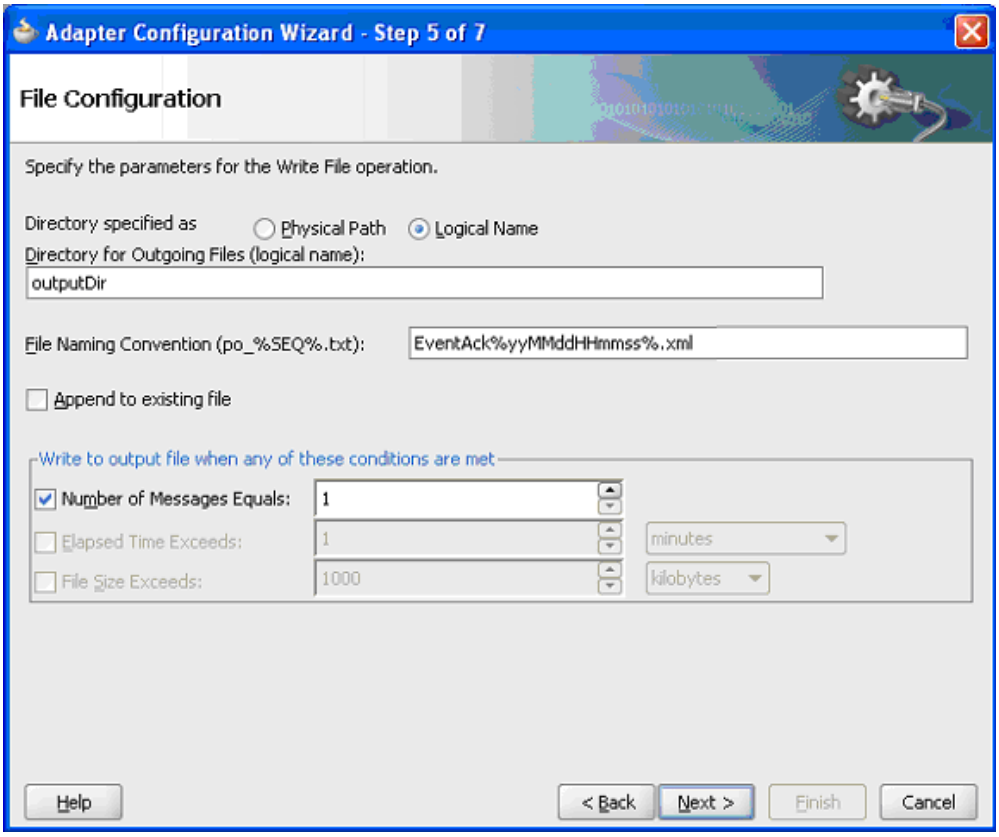
5. In the Operation page, specify the operation type. For example, select the **Write File** radio button. This automatically populates the **Operation Name** field.

Specifying the Operation



6. Click **Next** to access the File Configuration page.

Configuring the Output File



7. For the **Directory specified as** field, select the **Logical Name** radio button. Enter `outputDir` as the **Directory for Outgoing Files (logical name)** and specify a naming convention for the output file, such as `EventAck%yyMMddHHmmss%.xml`.

   > **Tip:** When you type a percent sign (`%`), you can choose from a list of date variables or a sequence number variable (`SEQ`) as part of the filename.

   Confirm the default write condition: **Number of Messages Equals 1**.

8. Click **Next**, and the Messages page appears. For the output file to be written, you must provide a schema.

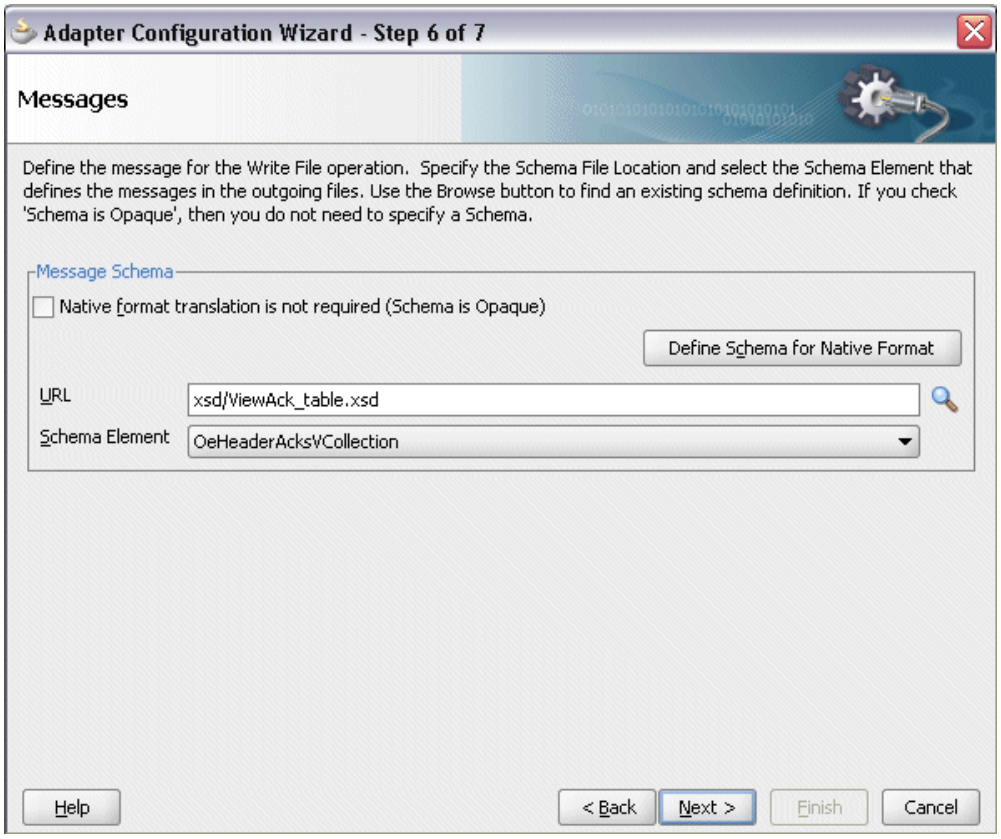
9. Click **Browse** to access the Type Chooser.

10. Expand the node by clicking **Project Schema Files > ViewAck_table.xsd** and selecting **OeHeaderAcksVCollection**. Click **OK**.

    The selected schema information will be automatically populated in the URL and

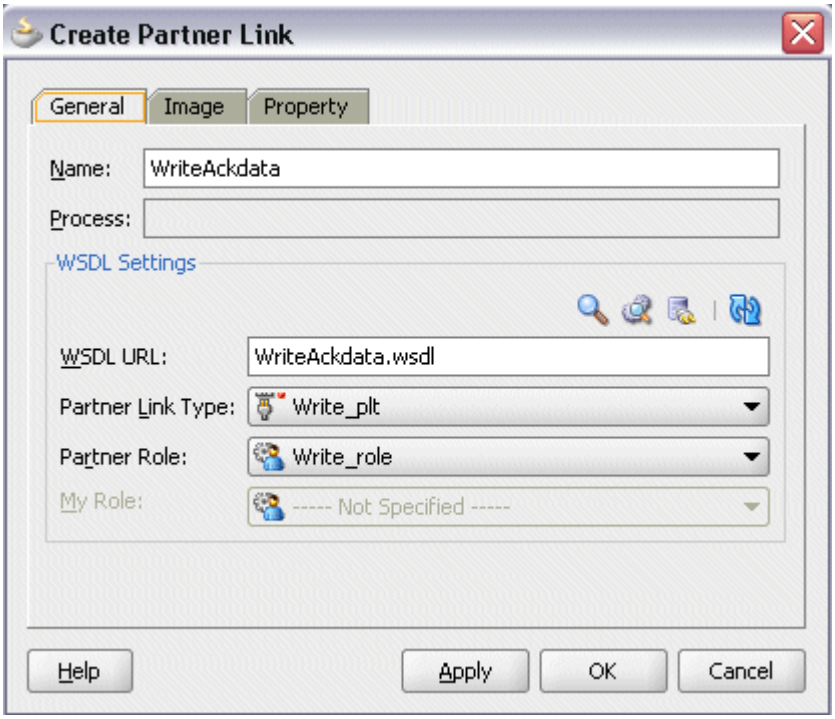
Schema Element fields.

Populating the Selected Message Schema



11. Click **Next** and then **Finish**. The wizard generates the WSDL file corresponding to the partner link. The main Create Partner Link dialog box appears, specifying the new WSDL file.

Completing the Partner Link Configuration



12. Click **Apply** and then **OK** to complete the configuration and create the partner link with the required WSDL settings for the File Adapter service.

Configuring the Invoke Activities

Based on the scenario described earlier, you need to configure the following two Invoke activities:

1. To get the purchase order acknowledgement details by invoking the `ViewAck` partner link.

2. To write the purchase order acknowledgement information to an XML file by invoking `WriteAckdata` partner link for File Adapter.

To add the first Invoke activity for a partner link to get acknowledgement details:

1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the first **Invoke** activity into the center swim lane of the process diagram, between the **receiveInput** and **callbackClient** activities.

2. Link the Invoke activity to the `ViewAck` service. The Edit Invoke dialog appears.

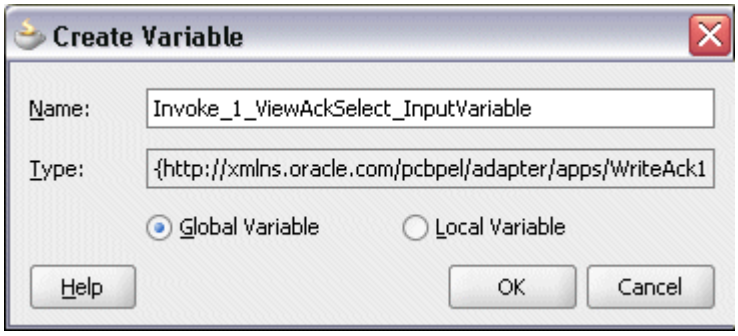
The value of the Operation field is automatically selected based on the associated partner link. For example, this Invoke activity is associated with an interface table partner link for 'select' operation only, the 'ViewAckSelect' service name with 'Select' operation is populated as the value.

3. Enter a name for the Invoke activity, then click the **Create** icon next to the **Input Variable** field to create a new variable. The Create Variable dialog box appears.

4. Select **Global Variable**, then enter a name for the variable. You can also accept the default name.

   Click **OK**.

Create Variable

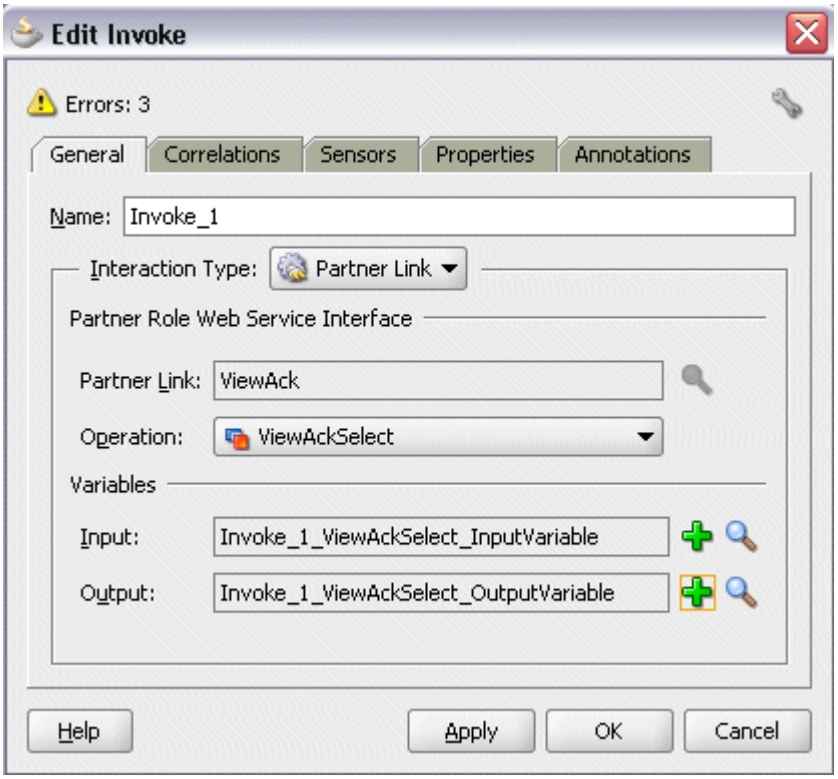


5. Click the **Create** icon next to the **Output Variable** field to create a new variable. The Create Variable dialog box appears.

6. Select **Global Variable**, then enter a name for the variable. You can also accept the default name.

   Click **OK** to close the Create Variable dialog.

   Click **Apply** and then **OK** in the Edit Invoke dialog box to finish configuring the Invoke activity.



The Invoke activity appears in the process diagram.

To add the second Invoke activity for a File Adapter partner link to write acknowledgement details in an XML file:

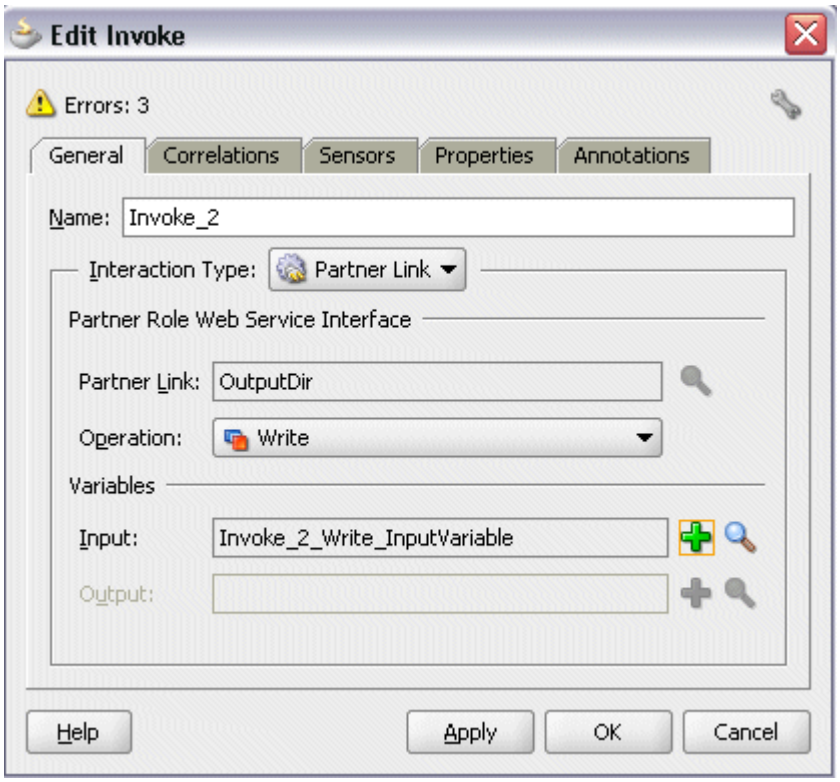
1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the first **Invoke** activity into the center swim lane of the process diagram, right after the first **Invoke** activity.

2. Link the Invoke activity to the `WriteAckdata` service for File Adapter. The Edit Invoke dialog appears.

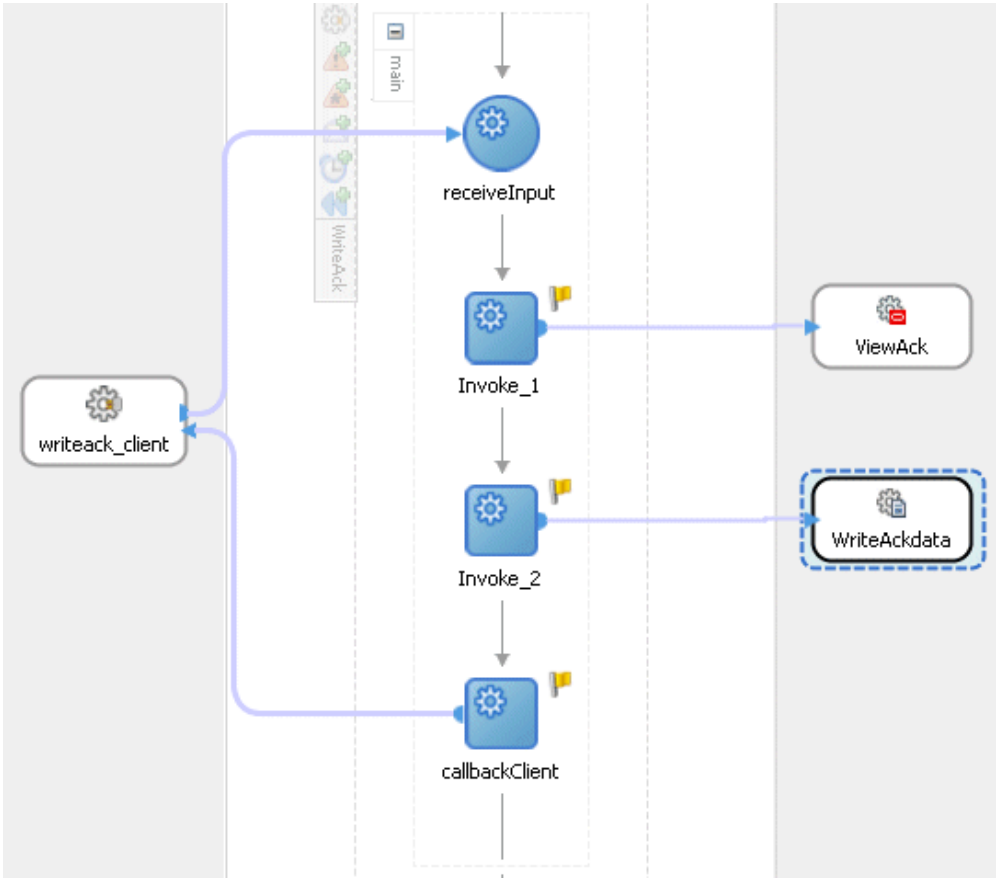
3. Enter a name for the Invoke activity, then click the **Create** icon next to the **Input Variable** field to create a new variable. The Create Variable dialog appears.

4. Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK** to return to the Edit Invoke dialog box.

   Click **Apply** and then **OK** to finish configuring the Invoke activity.



The second Invoke activity appears in the process diagram.



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.

To configure the Assign activity:

1. In JDeveloper BPEL Designer, select **BPEL Activities and Components** in the component palette.

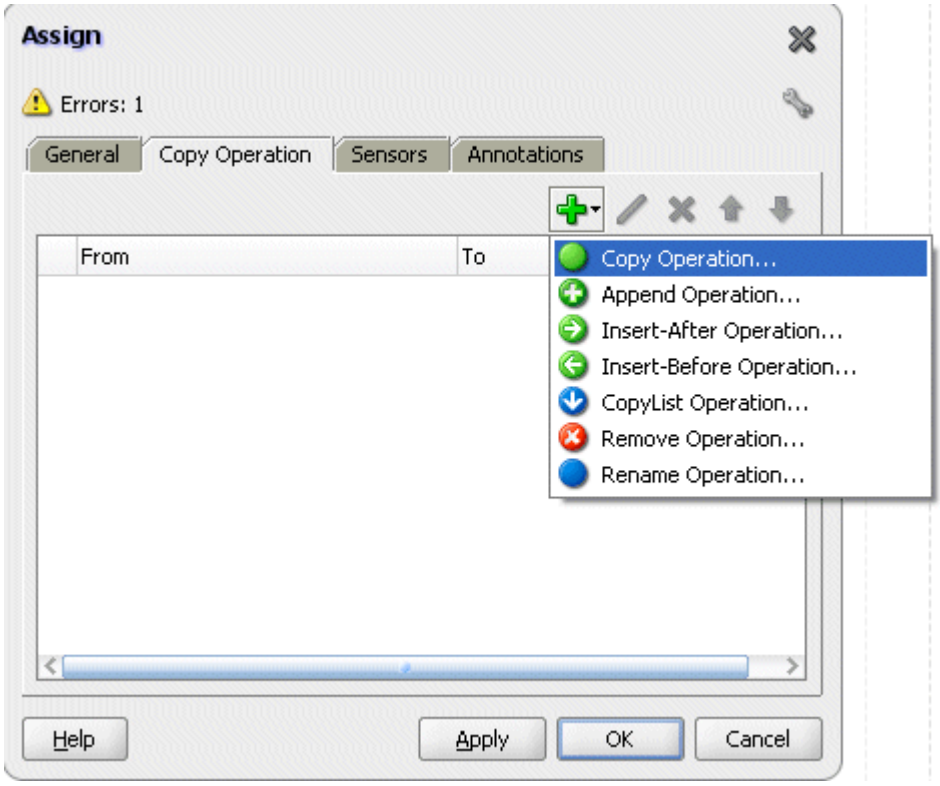
   Drag and drop the **Assign** activity into the center swim lane of the process diagram between the two **Invoke** activities that you just created earlier.

2. Double-click the **Assign** activity to access the Edit Assign dialog.

   Click the General tab to enter a name for the Assign activity. For example, `setAckdata`.

3. Select the Copy Operation tab, click the 'Plus' sign icon and select **Copy Operation** from the menu. The Create Copy Operation window appears.

Specifying a Copy Operation Action

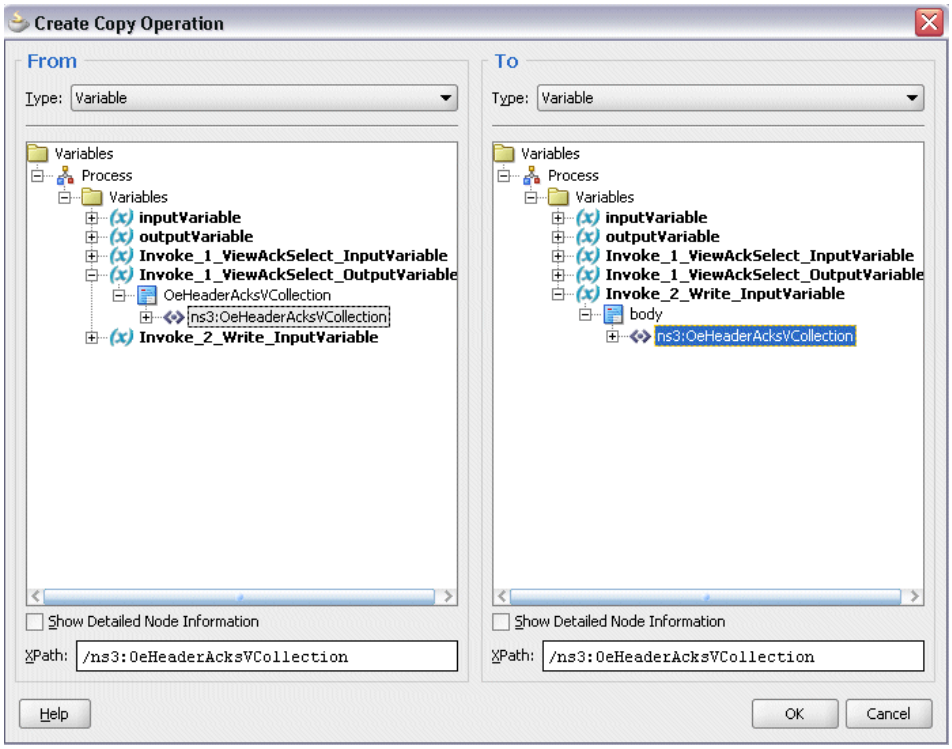


4. Enter the parameter information:

   - In the From navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_1_ViewAckSelect_OutputVariaable >OeHeaderAcksVCollection** and select **ns3:OeHeaderAcksVCollection**.

     The XPath field should contain your selected entry.

   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_2_Write_InputVariable > body** and select **ns3:OeHeaderAcksVCollection**. The XPath field should contain your selected entry.

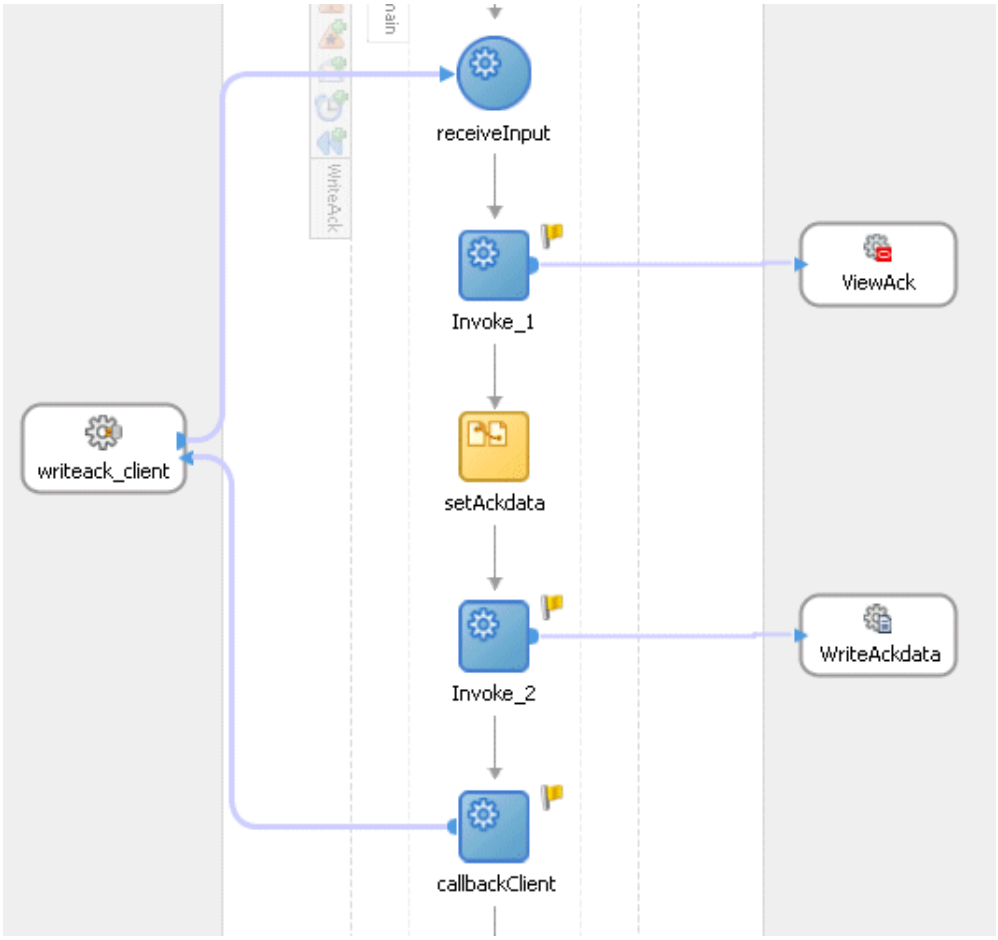


- Click **OK**. The Edit Assign dialog box appears.

After assigning values to the input variables, click **Apply** and then **OK**.

The complete BPEL process diagram should be shown:

BPEL Process Diagram



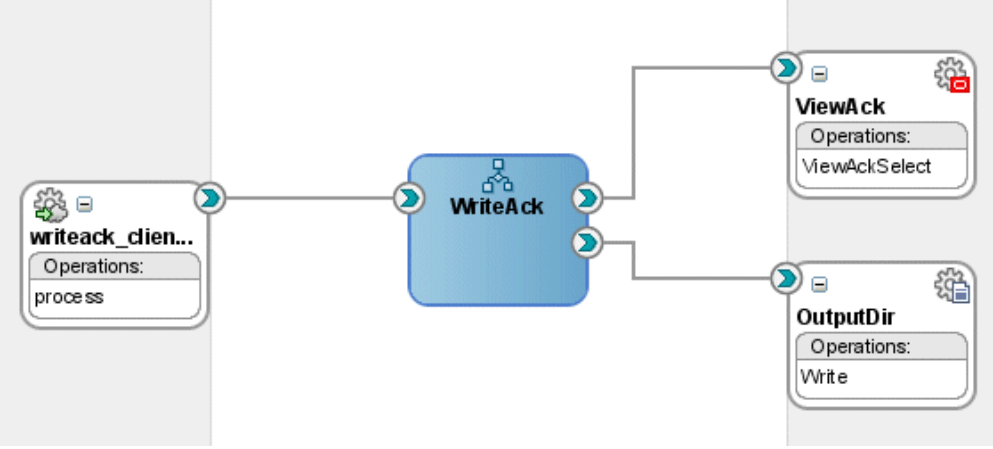
Click the `composite.xml` to display the Oracle JDeveloper composite diagram:

> **Note:** Click the Source tab of `composite.xml` to enter a value for the physical directory `outputDir` for the reference WriteAckdata (such as `/usr/tmp`).
>
> ```
> <property name="outputDir" type="xs:string"
> many="false" override="may">/usr/tmp</property>
> ```

Specifying the Physical Directory for the Property

```
                    callbackInterface="http://xmlns.oracle.com/writeAck1/writeAck1/writeA
    <binding.ws port="http://xmlns.oracle.com/WriteAck1/WriteAck1/WriteAck#wsdl.endpoint(
    <callback>
      <binding.ws port="http://xmlns.oracle.com/WriteAck1/WriteAck1/WriteAck#wsdl.endpoin
    </callback>
  </service>
  <component name="WriteAck">
    <implementation.bpel src="WriteAck.bpel"/>
  </component>
  <reference name="ViewAck" ui:wsdlLocation="ViewAck.wsdl">
    <interface.wsdl interface="http://xmlns.oracle.com/pcbpel/adapter/apps/WriteAck1/Writ
    <binding.jca config="ViewAck_apps.jca"/>
    <property name="jca.retry.count" type="xs:int" many="false" override="may">9</propert
    <property name="jca.retry.interval" type="xs:int" many="false"
              override="may">1</property>
    <property name="jca.retry.backoff" type="xs:int" many="false"
              override="may">2</property>
  </reference>
  <reference name="OutputDir" ui:wsdlLocation="OutputDir.wsdl">
    <interface.wsdl interface="http://xmlns.oracle.com/pcbpel/adapter/file/WriteAck1/Writ
    <binding.jca config="OutputDir_file.jca"/>
    <property name="outputDir" type="xs:string" many="false" override="may"></property>
  </reference>
  <wire>
    <source.uri>writeack_client_ep</source.uri>
    <target.uri>WriteAck/writeack_client</target.uri>
  </wire>
  <wire>
    <source.uri>WriteAck/ViewAck</source.uri>
    <target.uri>ViewAck</target.uri>
  </wire>
  <wire>
```

Oracle JDeveloper Composite Diagram



Run-Time Tasks for Views

After designing the BPEL process, the next steps are to deploy, run and monitor it.

1. Deploy the BPEL process., page 7-73

2. Test the BPEL process., page 7-74

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 application server (Oracle WebLogic Server) that you have established the connection.

Prerequisites

Before deploying the BPEL process using Oracle JDeveloper, you must ensure the following:

- You must have established the connectivity between the deign-time environment and an application server.

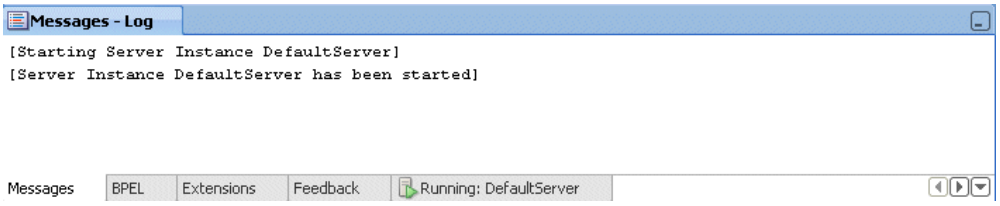
  For more information, see Configuring the Data Source in Oracle WebLogic Server, page A-3 and Creating an Application Server Connection, page A-8.

- Oracle WebLogic Server has been started.

  Before deploying the BPEL process, you need to start the Oracle WebLogic Server that you have established the connection.

  If a local instance of the WebLogic Server is used, start the WebLogic Server by selecting **Run > Start Server Instance** from Oracle JDeveloper.

  Once the WebLogic Admin Server "DefaultServer" instance is successfully started, you should find that <Server started in Running mode> and DefaultServer started message in the Running:DefaultServer and Messages logs.



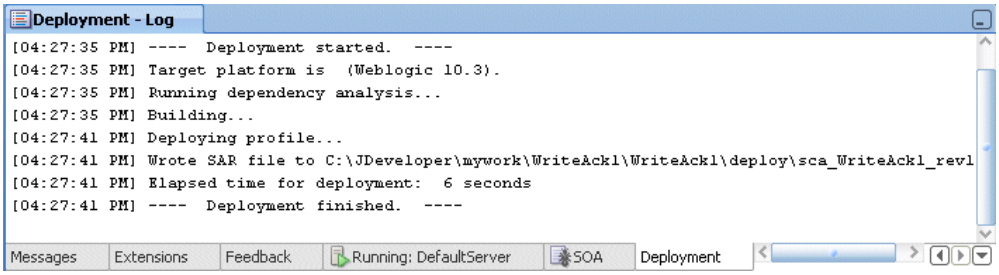
To deploy the BPEL process:

1. Select the BPEL project in the Applications Navigator.

2. Right-click the project name, and then select **Deploy > [project name] > [serverConnection]** from the menu that appears.

3. The BPEL process is compiled and deployed. You can check the progress of the compilation in the Messages window.

Messages Window

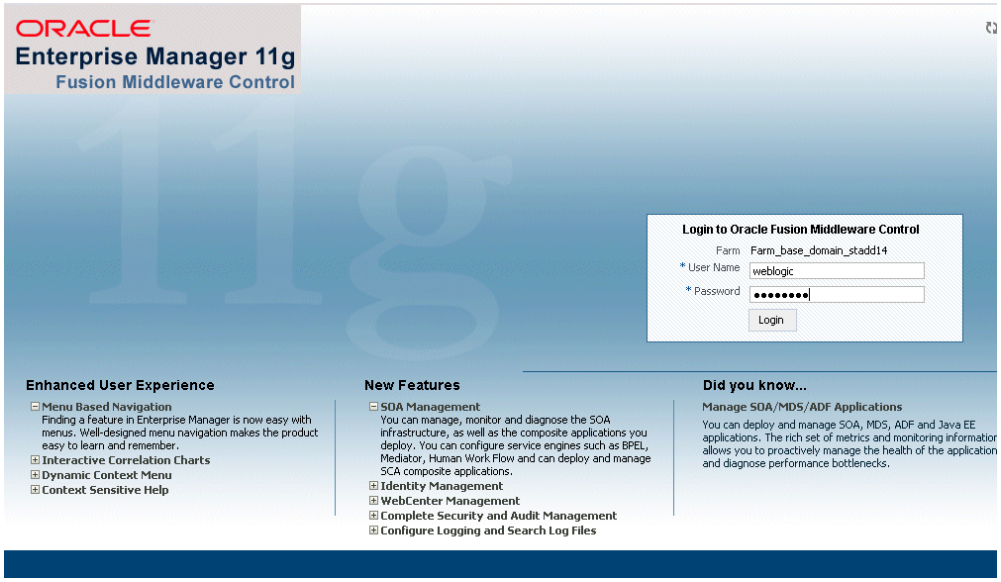


Testing the BPEL Process

Once the BPEL process is deployed, you can manage and monitor the process from the Oracle Enterprise Manager Fusion Middleware Control Console. You can also test the process and the integration interface by manually initiating the process.

To manually initiate and monitor the BPEL process:

1. Navigate to Oracle Enterprise Manager Fusion Middleware Control Console (`http://<servername>:<portnumber>/em`). The composite you deployed is displayed in the Applications Navigation tree.



2. Enter username (such as `weblogic`) and password and click **Login** to log in to a farm.

   You may need to select an appropriate target instance farm if there are multiple target Oracle Enterprise Manager Fusion Middleware Control Console farms.

3. From the Farm base domain, expand the **SOA** >**soa-infra** to navigate through the SOA Infrastructure home page and menu to access your deployed SOA composite applications running in the SOA Infrastructure for that managed server.

   > **Note:** The Farm menu always displays at the top of the navigator. As you expand the SOA folder in the navigator and click the links displayed beneath it, the SOA Infrastructure menu becomes available at the top of the page.

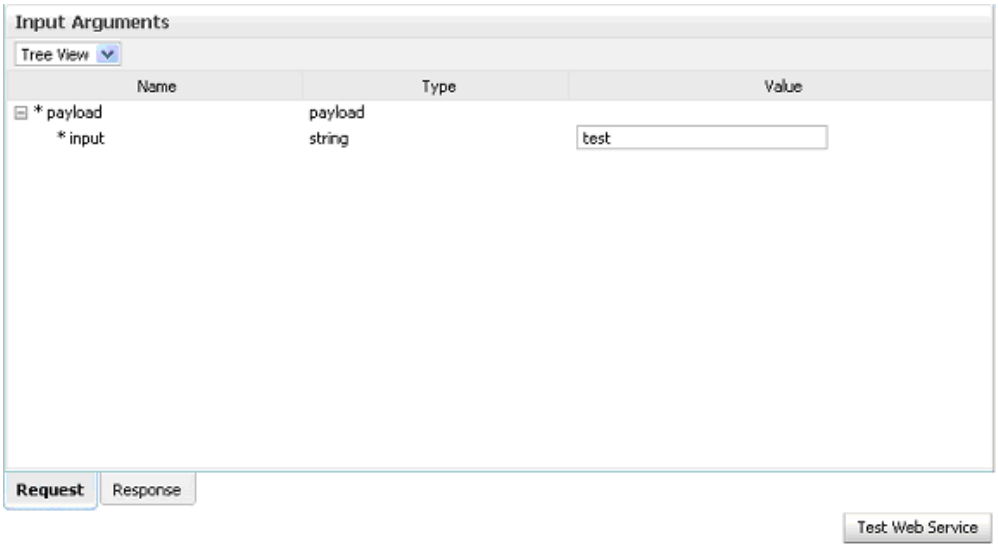
   Click the SOA composite application that you want to initiate (such as 'WriteAck') from the SOA Infrastructure.

   Click **Test** at the top of the page.

4. The Test Web Service page for initiating an instance appears. You can specify the XML payload data to use in the Input Arguments section.

   Enter the input string required by the process and click **Test Web Service** to initiate the process.

Testing Web Service



The test results appear in the Response tab upon completion.

5. Click on the BPEL process name and then click the Instances tab. The SOA composite application instance ID, name, conversation ID, most recent known state of each instance since the last data refresh of the page are displayed.

   In the Instance ID column, click a specific instance ID to show the message flow through the various service components and binding components. The Flow Trace page is displayed.

   In the Trace section, you should find the sequence of the message flow for the service binding component (`writeack_client_ep`), BPEL component (`WriteAck`), and reference binding components (`ViewAck` and `WriteAckdata`). All involved components have successfully received and processed messages.

   If any error occurred during the test, you should find it in the Faults section.

6. Click your BPEL service component instance link (such as `WriteAck`) to display the Instances page where you can view execution details for the BPEL activities in the Audit Trail tab.

   Click the Flow tab to check the BPEL process flow diagram. Click an activity of the process diagram to view the activity details and flow of the payload through the process.

7. **Verifying Records in Oracle Applications**

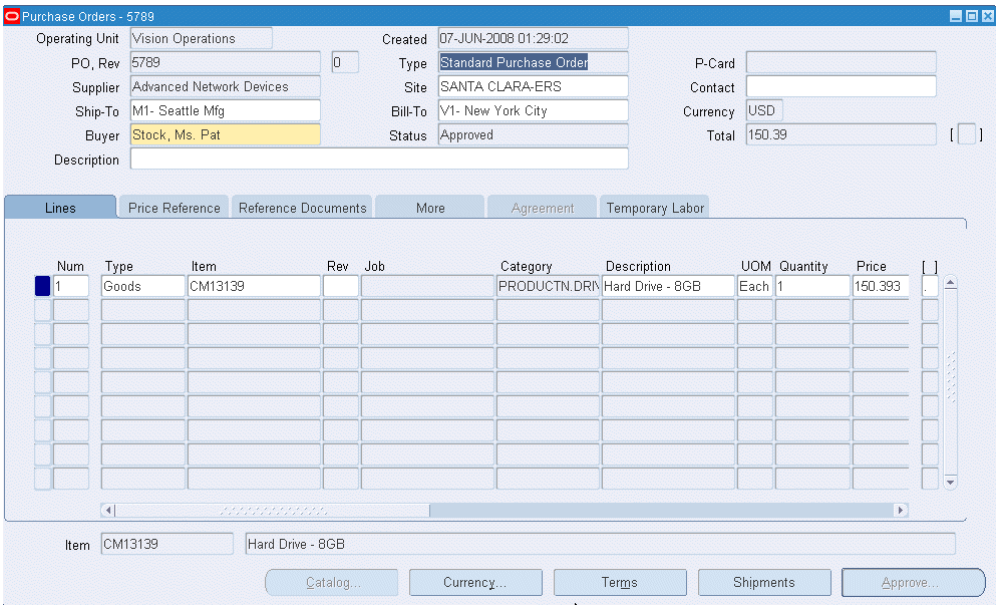
   Log on to Oracle Applications with Purchasing, Vision Operations (USA) responsibility and select Purchase Order from the navigation menu.

The Oracle Applications Forms open up with the Purchase Order forms.

8. Create a purchase order with the following header values:
   - Supplier: Enter a supplier information, such as 'Advanced Network Devices'.
   - Site: Select a site information, such as 'SANTA CLARA-ERS'.
9. On the Lines tab, enter a data row with the following values:
   - Type: Goods
   - Item: CM13139
   - Quantity: 1
   - Description: Hard Drive - 8GB
   - Promised: Enter any future date in the format of dd-mmm-yyyy (such as 23-JUN-2009)
10. Save your purchase order. The status of the purchase order is 'Incomplete'.
11. Click **Approve**. The Approve Document form appears.

    Click **OK** to confirm the approval.

    Note: Because the trading partner is set up and valid, the transmission method is automatically set to **XML**.



The status of the purchase order is now changed to 'Approved'. For future reference, note the value of the PO, Rev field. For example, the PO number 5789.

Once the purchase order is approved, the order details should be recorded in the system.

After deploying the BEPL process, the order acknowledgement information should be retrieved from the `OE_HEADER_ACKS_V` and `OE_LINE_ACKS_V` interface views.

Validate the output file `EventAck%yyMMddHHmmss%.xml` in the specified output directory by opening the xml file to confirm the purchase order details. The document number in the xml file should be the same number you just approved in Oracle Applications.

8

Using PL/SQL APIs

This chapter covers the following topics:

- Overview of PL/SQL APIs
- Design-Time Tasks for PL/SQL APIs
- Creating a New BPEL Project
- Adding Partner Links
- Adding a Partner Link for File Adapter
- Defining Wrapper APIs
- Declaring Parameters with a DEFAULT Clause
- Configuring the Invoke Activities
- Configuring the Transform Activity
- Configuring an Assign Activity
- Run-Time Tasks for PL/SQL APIs
- Deploying the BPEL Process
- Testing the BPEL Process
- Troubleshooting and Debugging

Overview of PL/SQL APIs

Adapter for Oracle Applications uses PL/SQL application programming interfaces (APIs) to insert and update data in Oracle Applications. APIs are stored procedures that enable you to insert and update data in Oracle Applications. Additionally, you can use PL/SQL APIs to retrieve data. For example, by using PL/SQL APIs, you can insert a customer record in Oracle Applications.

Note: For more information about PL/SQL procedure limitations, refer

to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

Design-Time Tasks for PL/SQL APIs

This section describes how to configure the Adapter for Oracle Applications to use PL/SQL APIs. It describes the tasks required to configure Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper.

BPEL Process Scenario

In this example, Adapter for Oracle Applications exposes the following stored procedures as Web services in a BPEL process to update the 'quantity' field of an existing purchase order based on user input.

- A custom Order Management stored procedure `GET_G_MISS_LINE_REC (GET_G_MISS_LINE_REC)` to initialize the purchase order
- A PL/SQL Process Order Line (`PROCESS_LINE`) to update the existing purchase order

When a change order request is received, the purchase order information including order quantity and other line item details will be retrieved. Based on user input, a new order quantity will be updated in Oracle Order Management.

If the BPEL process is successfully executed after deployment, you can validate the process by querying it directly from Order Management tables or validate it from the Formed-based Oracle Order Management application. The retrieved `ordered_quantity` value from the query table or you find in the Order Management application should be the same as the quantity value given by the user through `changeorder_data.xml` file.

Prerequisites to Configure PL/SQL APIs

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.

Populating Application Context Header Variables

You need to populate certain variables in the BPEL PM in order to provide context information for Oracle Applications. The context information is required for an API transaction in order for an Oracle Applications user that has sufficient privileges to run the program.

The context is set taking into account the values passed for the header properties including *Username*, *Responsibility*, *Responsibility Application*, *Security Group*, and *NLS Language*. If the values for the new header properties *Responsibility Application*, *Security Group*, and *NLS Language* are not passed, context information will be determined based on *Username* and *Responsibility*.

The default Username is `SYSADMIN`, the default Responsibility is `SYSTEM`

ADMINISTRATOR, the default Security Group Key is Standard, and the default NLS Language is US.

You can change the default values specified in the generated WSDL. 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 values. The context information can be specified by configuring an Invoke activity.

For more information about applications context, see Supporting for Normalized Message Properties, page 3-8.

Populating Default Values for Record Types

Certain PL/SQL APIs exposed from Oracle E-Business Suite take record types as input. Such APIs expect default values to be populated for parameters within these record types for successful execution.

The default values are FND_API.G_MISS_CHAR for characters, FND_API.G_MISS_DATE for dates, and FND_API.G_MISS_NUM for numbers. Adapter for Oracle Applications can default these values when the parameters within the record type are passed as nil values, for example, as shown below:

```
<PRICE_LIST_REC>
<ATTRIBUTE1 xsi:nil="true"/>
<ATTRIBUTE2 xsi:nil="true"/>
<ATTRIBUTE3 xsi:nil="true"/>
...
</PRICE_LIST_REC>
```

This can be achieved with the help of a function in a Transform activity, or by directly passing the XML input with nil values and then assigning them to the record types within an Assign activity.

Following is a list of the procedures required to accomplish the design-time tasks.

1. Create a new BPEL project., page 8-4
2. Add partner links., page 8-8

   Add a partner link for File Adapter., page 8-22
3. Define wrapper APIs., page 8-27
4. Declare parameters with a DEFAULT clause., page 8-30
5. Configure the Invoke Activities, page 8-33
6. Configure the Transform Activity, page 8-38
7. Configure an Assign Activity, page 8-40

Creating a New BPEL Project

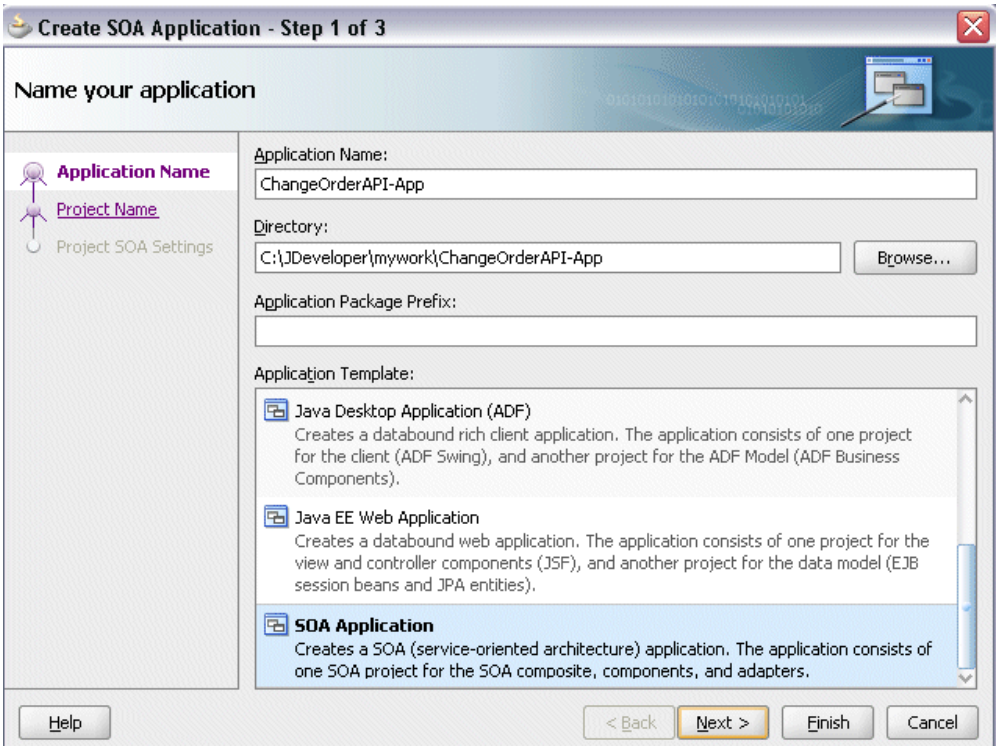
To create a new BPEL project:

1. Launch Oracle JDeveloper.

2. Click **New Application** in the Application Navigator.

   The Create SOA Application - Name your application page is displayed.

The Create SOA Application - Name your application Page

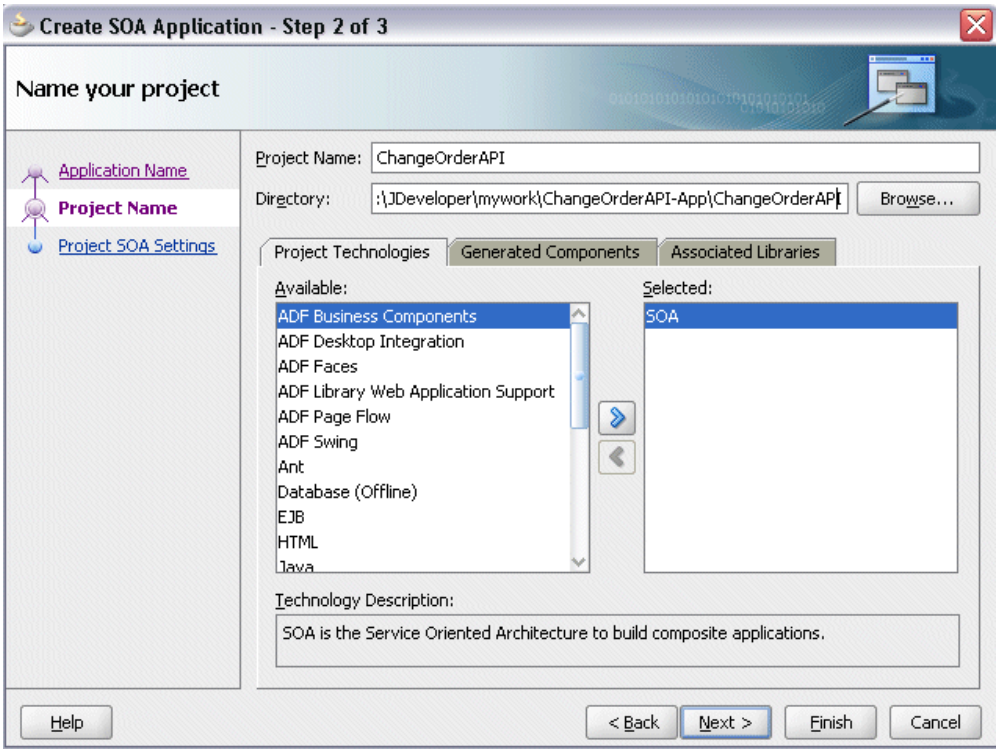


3. Enter an appropriate name for the application in the **Application Name** field and select **SOA Application** from the Application Template list.

   Click **Next**. The Create SOA Application - Name your project page is displayed.

4. Enter an appropriate name for the project in the **Project Name** field. For example, `ChangeOrderAPI`.

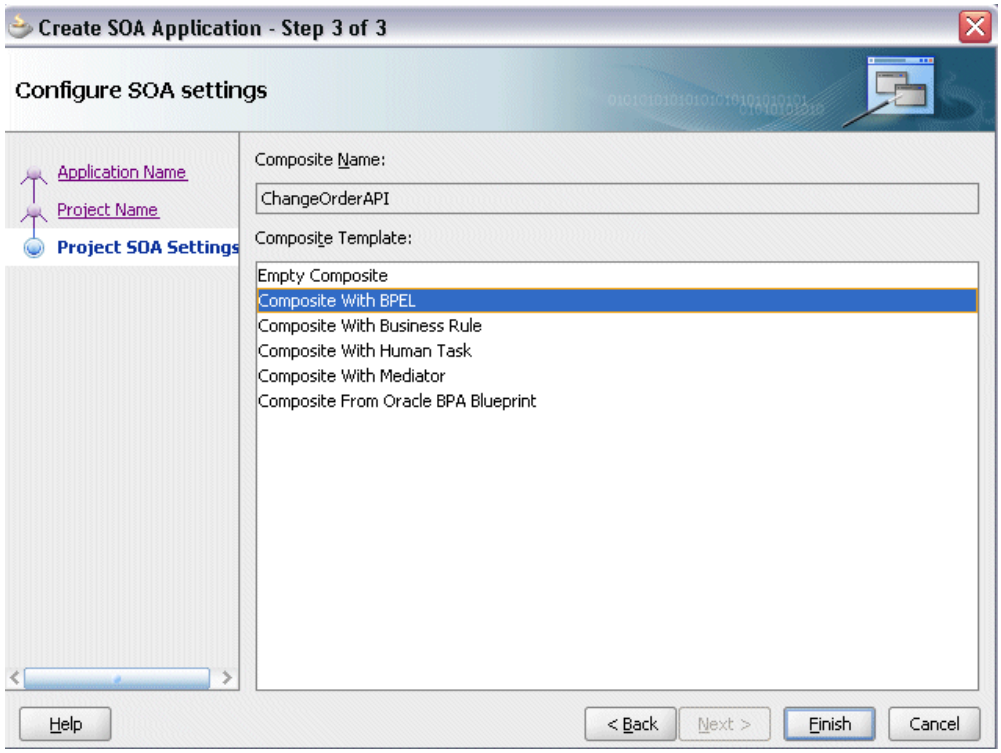
The Create SOA Application - Name your project Page



5. In the Project Technologies tab, ensure that **SOA** is selected from the Available technology list to the Selected technology list.

   Click **Next**. The Create SOA Application - Configure SOA settings page is displayed.

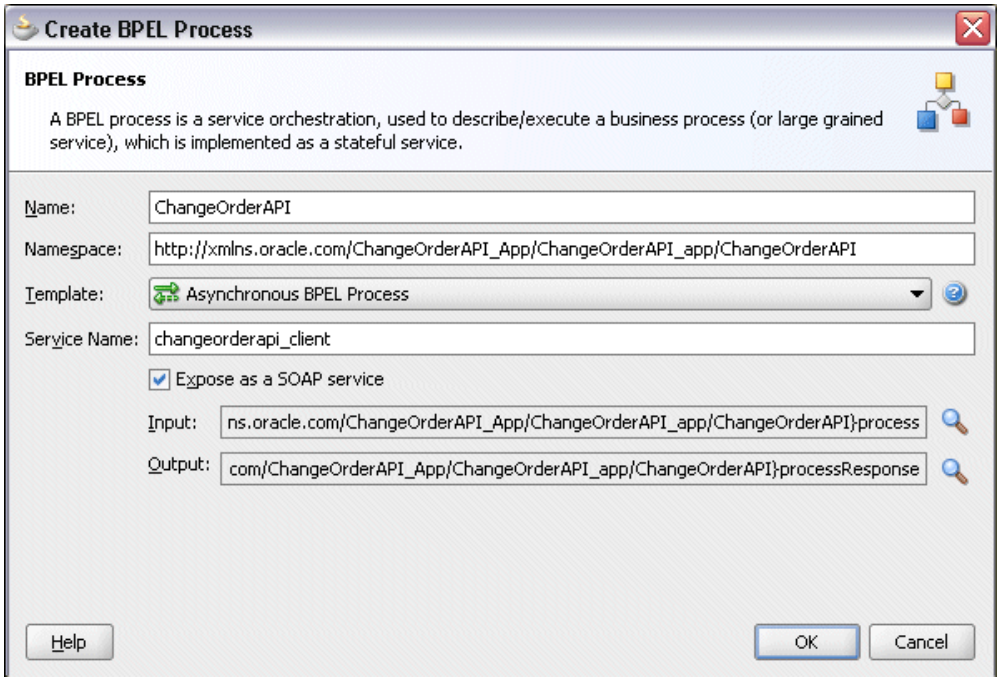
The Create SOA Application - Configure SOA settings Page



6. Select **Composite With BPEL** from the Composite Template list, and then click **Finish**. You have created a new application, and an SOA project. This automatically creates an SOA composite.

   The Create BPEL Process page is displayed.

The Create BPEL Process Page

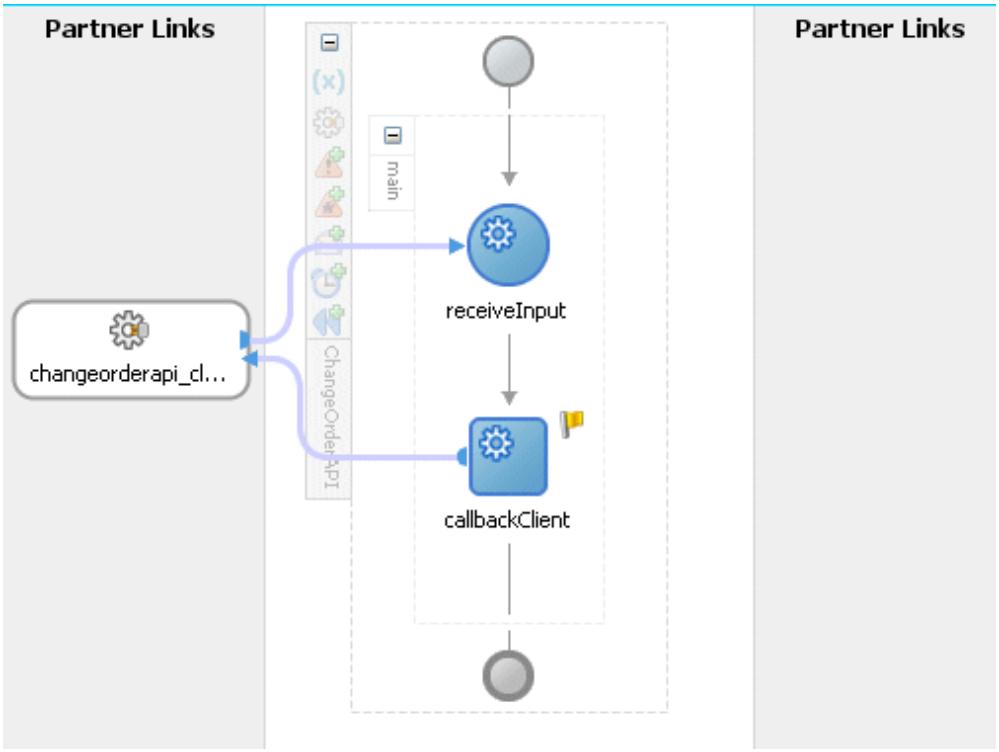


7. Enter an appropriate name for the BEPL process in the **Name** field. For example, `ChangeOrderAPI`.

   Select **Asynchronous BPEL Process** in the **Template** field. Click **OK**.

   An asynchronous BPEL process is created with the Receive and Reply activities. The required source files including bpel and wsdl, using the name you specified (for example, `ChangeOrderAPI.bpel` and `ChangeOrderAPI.wsdl`) and `composite.xml` are also generated.

New BPEL Process



Adding Partner Links

The next task is to add a partner link to the BPEL process. This section describes how to create an Oracle Applications 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.

Based on the BPEL process scenario discussed earlier, the following two partner links need to be configured:

- Add the first partner link `(initLineRec)` to initialize the purchase order
- Add the second partner link `(OrderManagement)` to update the existing purchase order

To add the first partner link:

1. Click **BPEL Services** in the Component palette.

   Drag and drop **Oracle Applications** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears. Click **Next**.

2. Enter a service name in the **Service Name** field. For example, `initLineRec`.

   Click **Next**. The Service Connection dialog appears.

3. You can perform either one of the following options for your database connection:

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

   - You can create a new database connection by clicking the **Create a New Database Connection** icon.

     How to define a new database connection, see Create a New Database Connection, page 4-13.

   - You can select an existing database connection that you have configured earlier from the **Connection** drop-down list.

     The Service Connection page will be displayed with the selected connection information. The JNDI (Java Naming and Directory Interface) name corresponding to the database connection appears automatically in the **Database Server JNDI Name** field. Alternatively, you can specify a JNDI name.

     > **Note:** When you specify a JNDI name, the deployment descriptor of the Oracle Applications adapter must associate this JNDI name with configuration properties required by the adapter to access the database.

   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.

   > **Note:** For more information about JNDI concepts, refer to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

4. Once you have completed creating a new connection for the service, you can add a PL/SQL API by browsing through the list of APIs available in Oracle Applications.

   Click **Next**.

   For Oracle E-Business Suite Release 12:

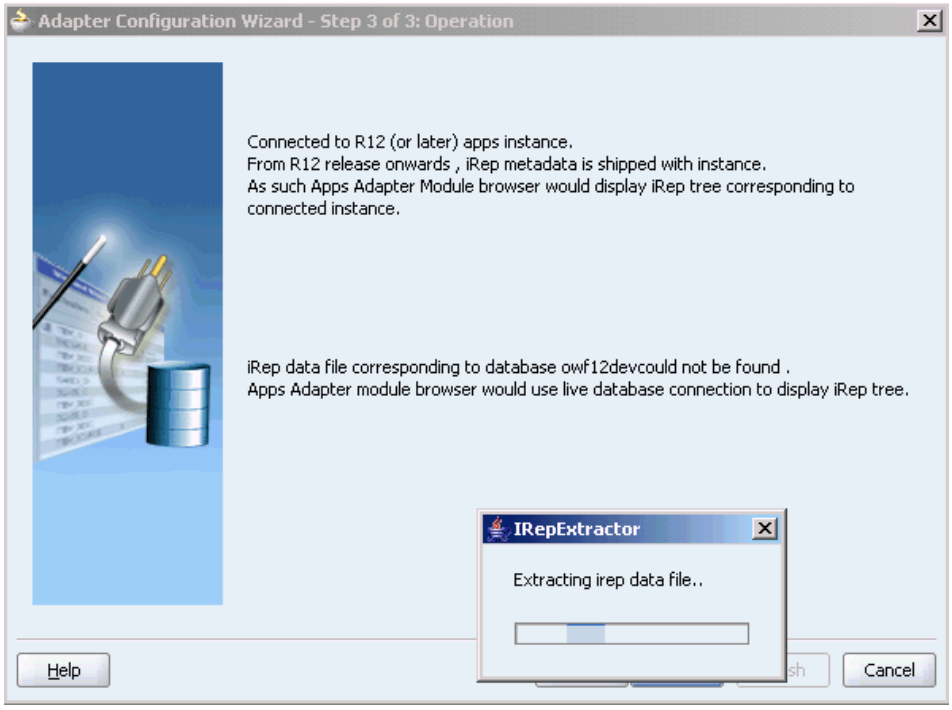
   If you are connecting to Oracle E-Business Suite Release 12, then the **IREP File not present** dialog appears indicating that Adapter could not find the Oracle Integration Repository data file corresponding to the database you are connecting to Oracle Applications in your workspace. Absence of the data file would make browsing or searching of Integration Repository tree considerably slow. You can

choose to extract the data file and create a local copy of the Integration Repository data file. Once it is created successfully, Adapter will pick it up automatically next time and retrieve data from your local Integration Repository.

You can select one of the following options:

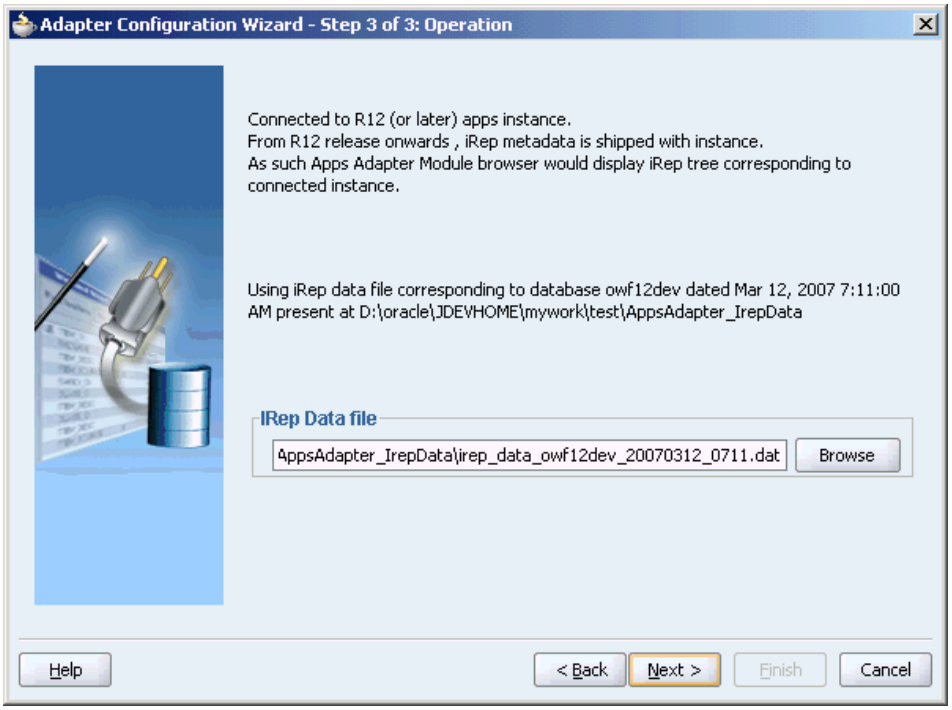
- Click **Yes** to extract the Integration Repository data file.

Extracting Integration Repository Data File



After the system successfully creates a local copy of the Integration Repository data file, next time when you connect to the database, you will find the **IRep Data File** field appears in the Operation dialog indicating where your local copy exists with the creation date and time as part of the file name.

Using the Local Integration Repository Data File



- Click **No** to query the Integration Repository data file from the live database you are connecting to display the Integration Repository tree.

  Note: It is highly recommended that you create a local copy of the Integration Repository data file so that Adapter will query the data next time from the local copy in your workspace to enhance the performance.

  Click **Next** in the Operation page to open the Oracle Applications Module Browser.

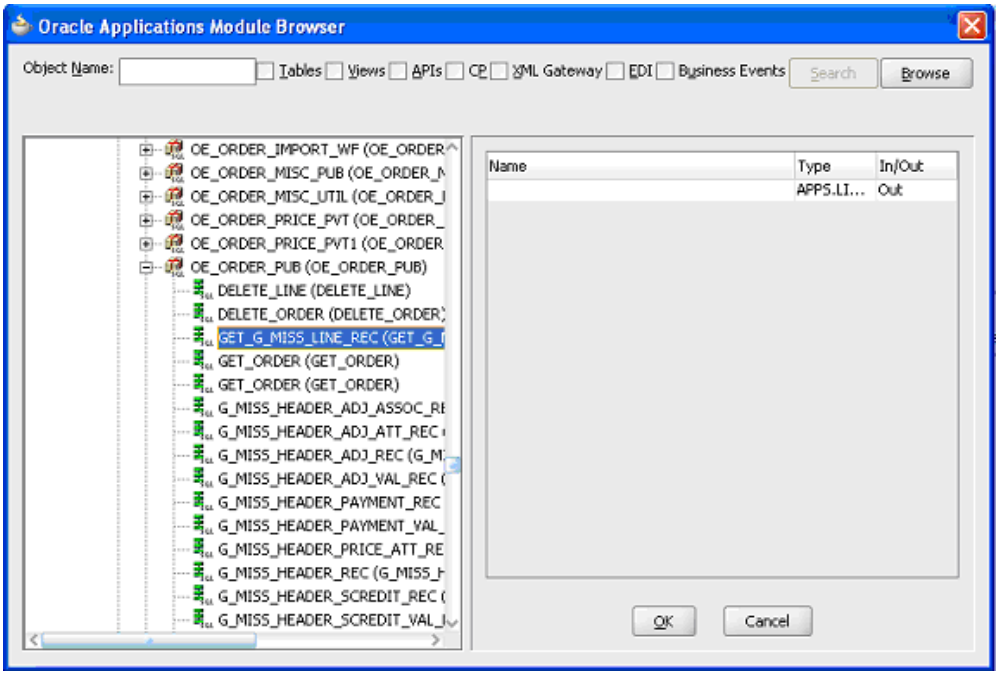
For Oracle E-Business Suite pre-Release 11.5.10:

If you are connecting to a pre-11.5.10 Oracle Applications instance, you must select the interface type in the Adapter Configuration Wizard. Select **Tables/Views/APIs/Concurrent Programs** to proceed.

Click **Get Object** in the Application Interface dialog to open the Oracle Applications Module Browser.

5. The Oracle Applications Module Browser combines interface data from Oracle Integration Repository with information about the additional interfaces supported by Oracle Application Adapter, organized in a tree hierarchy.

Specify an API from The Oracle Applications Module Browser



- The Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. Each product family contains the individual products. Each product contains the business entities associated with the product. Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide. PL/SQL APIs can be found under the PL/SQL category.

- You can specify whether customized APIs within an existing (Integration Repository) PL/SQL package are visible in the Module Browser tree. Right-click the name of a PL/SQL package, and click **Yes** to display all of its APIs (the default), or click **No** to hide the customized APIs.

  This selection applies separately to each PL/SQL package, and remains in effect only for the duration of the current JDeveloper session.

Navigate to *Other Interfaces* > *Custom Objects* >*PLSQL APIs* > *OE_ORDER_PUB* to select a custom stored procedure `GET_G_MISS_LINE_REC`.

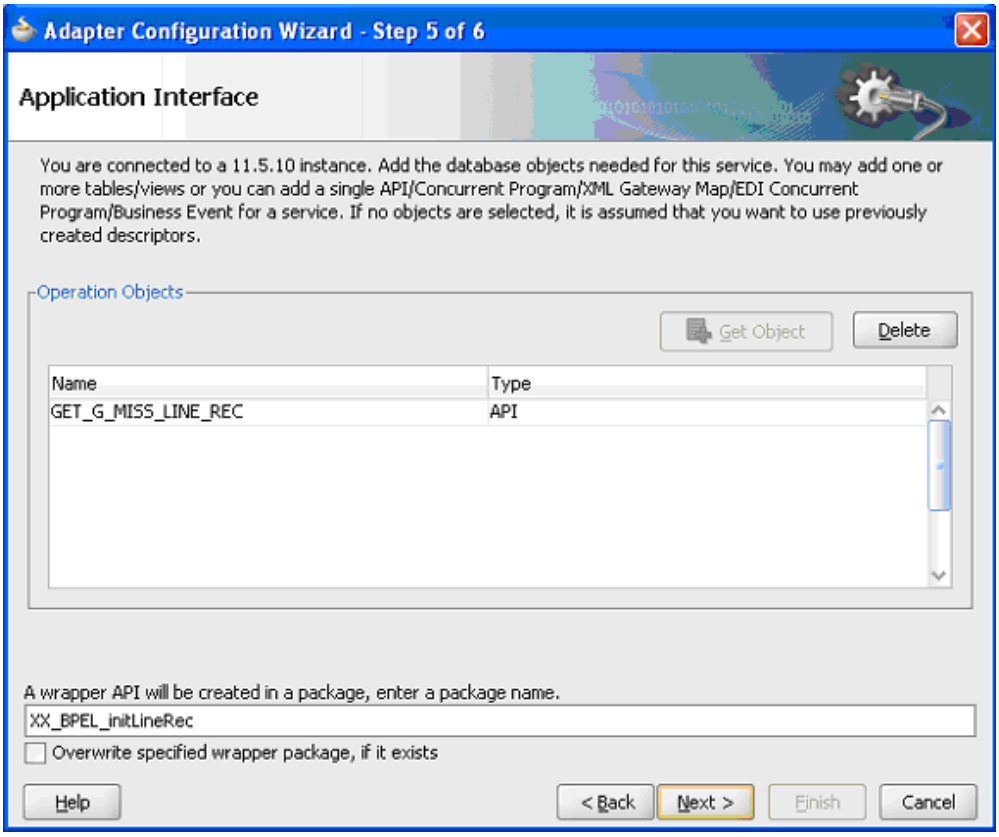
Note: The `GET_G_MISS_LINE_REC (GET_G_MISS_LINE_REC)` stored procedure does not take any input but has a complex data type (PL/SQL Table) as output. Adapter for Oracle Applications design-time automatically generates a wrapper stored procedure and loads it on the underlying database.

6. Click **OK**.

   Please note that the input and the output parameters of the Stored Procedure contains complex data types that are not readily mapped to JDBC types. The Adapter for Oracle Applications wizard provides a mechanism that detects when these types are used and then invokes Oracle JPublisher to generate the necessary wrappers automatically. Oracle JPublisher generates two SQL files, one to create schema objects, and another to drop them. The SQL that creates the schema objects is automatically executed from within the wizard to create the schema objects in the database schema before the XSD is generated.

   The Adapter Service points to the wrapper Stored Procedure instead of the Process Order Line Stored Procedure. The Adapter design-time automatically loads the wrapper procedure onto the underlying database.

Application Interface Page



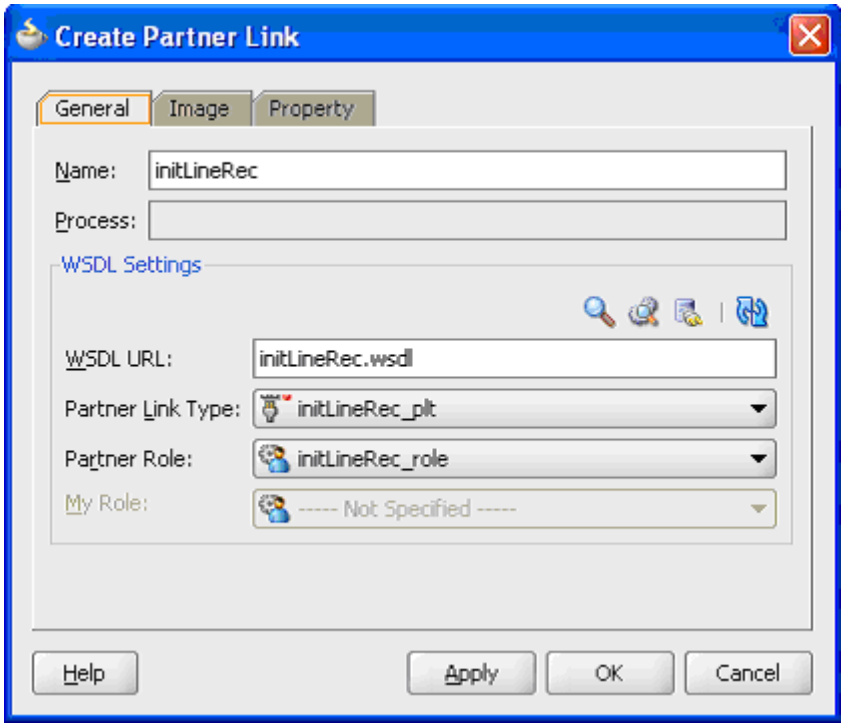
7. Click **Next**, then click **Finish** to complete the process of configuring Adapter for Oracle Applications.

   The wizard generates the WSDL file corresponding to the XML schema. This WSDL file is now available for the partner link.

8. Click **Apply** and then **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.

Partner Link Information



To add the second partner link:

1. Click **BPEL Services** in the Component palette.

   Drag and drop **Oracle Applications** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears. Click **Next**.

2. Enter a service name in the **Service Name** field. For example, `OrderManagement`.

   Click **Next**. The Service Connection dialog appears.

3. You can perform either one of the following options for your database connection:

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

   - You can create a new database connection by clicking **Create a New Database Connection** icon.

     How to define a new database connection, see Create a New Database Connection, page 4-13.

- You can select an existing database connection that you have configured earlier from the **Connection** drop-down list.

  The Service Connection page will be displayed with the selected connection information. The JNDI (Java Naming and Directory Interface) name corresponding to the database connection appears automatically in the **Database Server JNDI Name** field. Alternatively, you can specify a JNDI name.

  > **Note:** When you specify a JNDI name, the deployment descriptor of the Oracle Applications adapter must associate this JNDI name with configuration properties required by the adapter to access the database.

  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.

  > **Note:** For more information about JNDI concepts, refer to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

4. Once you have completed creating a new connection for the service, you can add a PL/SQL API by browsing through the list of APIs available in Oracle Applications.

   Click **Next**.

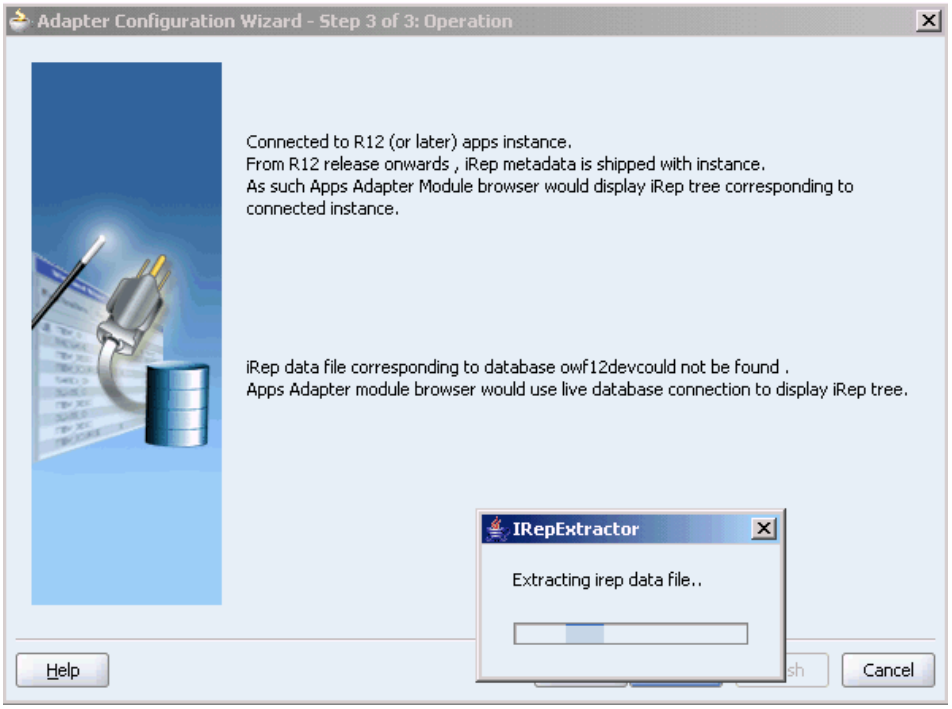
   For Oracle E-Business Suite Release 12:

   If you are connecting to Oracle E-Business Suite Release 12, then the **IREP File not present** dialog appears indicating that Adapter could not find the Oracle Integration Repository data file corresponding to the database you are connecting in your workspace. Absence of the data file would make browsing or searching of Integration Repository tree considerably slow. You can choose to extract the data file and create a local copy of the Integration Repository data file. Once it is created successfully, Adapter will pick it up automatically next time and retrieve data from your local Integration Repository.

   You can select one of the following options:

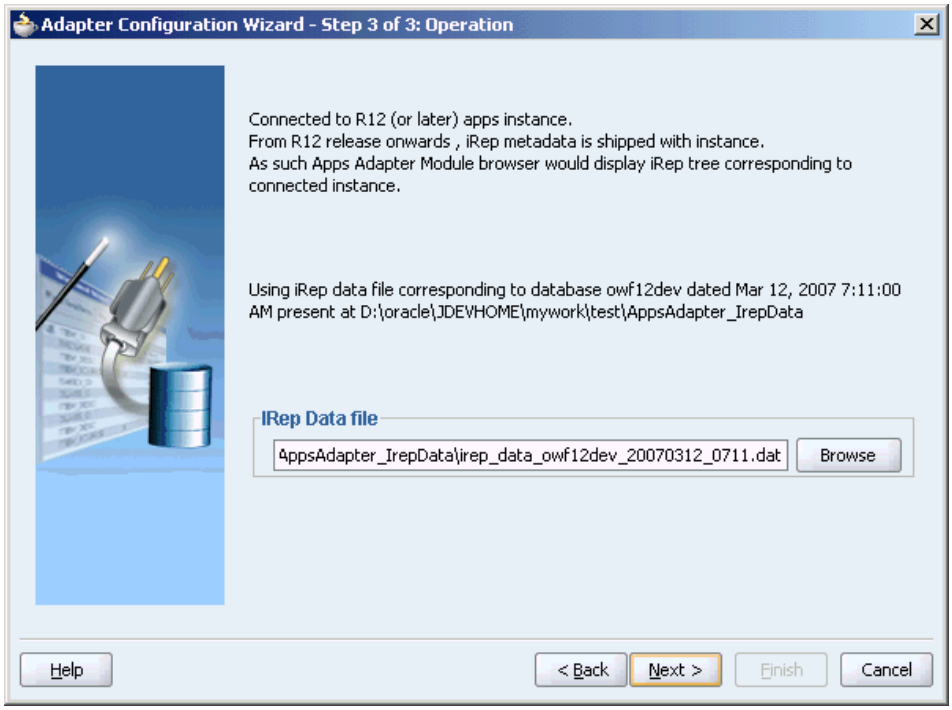
   - Click **Yes** to extract the Integration Repository data file.

Extracting Integration Repository Data File



After the system successfully creates a local copy of the Integration Repository data file, next time when you connect to the database, you will find the **IRep Data File** field appears in the Operation dialog indicating where your local copy exists with the creation date and time as part of the file name.

Using the Local Integration Repository Data File



- Click **No** to query the Integration Repository data file from the live database you are connecting to display the Integration Repository tree.

  Note: It is highly recommended that you create a local copy of the Integration Repository data file so that Adapter will query the data next time from the local copy in your workspace to enhance the performance.

  Click **Next** in the Operation page to open the Oracle Applications Module Browser.

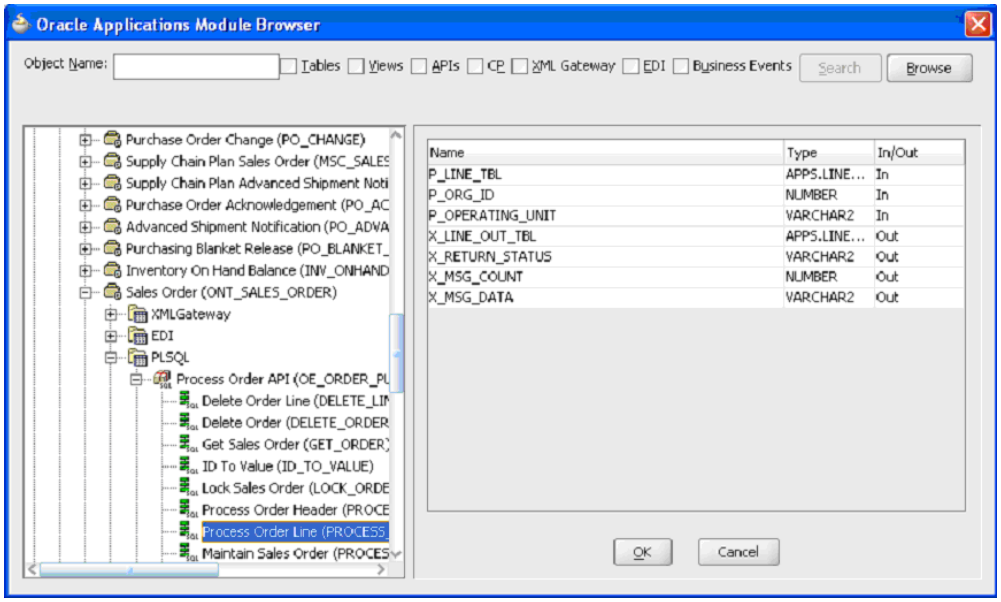
  For Oracle E-Business Suite pre-Release 11.5.10:

  If you are connecting to a pre-11.5.10 Oracle Applications instance, you must select the interface type in the Adapter Configuration Wizard. Select **Tables/Views/APIs/Concurrent Programs** to proceed.

  Click **Get Object** in the Application Interface dialog to open the Oracle Applications Module Browser.

5. The Oracle Applications Module Browser combines interface data from Oracle Integration Repository with information about the additional interfaces supported by Oracle Application Adapter, organized in a tree hierarchy.

Specify an API from The Oracle Applications Module Browser



- The Oracle Applications Module Browser includes the various product families that are available in Oracle Applications. Each product family contains the individual products. Each product contains the business entities associated with the product. Business entities contain the various application modules that are exposed for integration. These modules are grouped according to the interface they provide. PL/SQL APIs can be found under the PL/SQL category.

- You can specify whether customized APIs within an existing (Integration Repository) PL/SQL package are visible in the Module Browser tree. Right-click the name of a PL/SQL package, and click **Yes** to display all of its APIs (the default), or click **No** to hide the customized APIs.

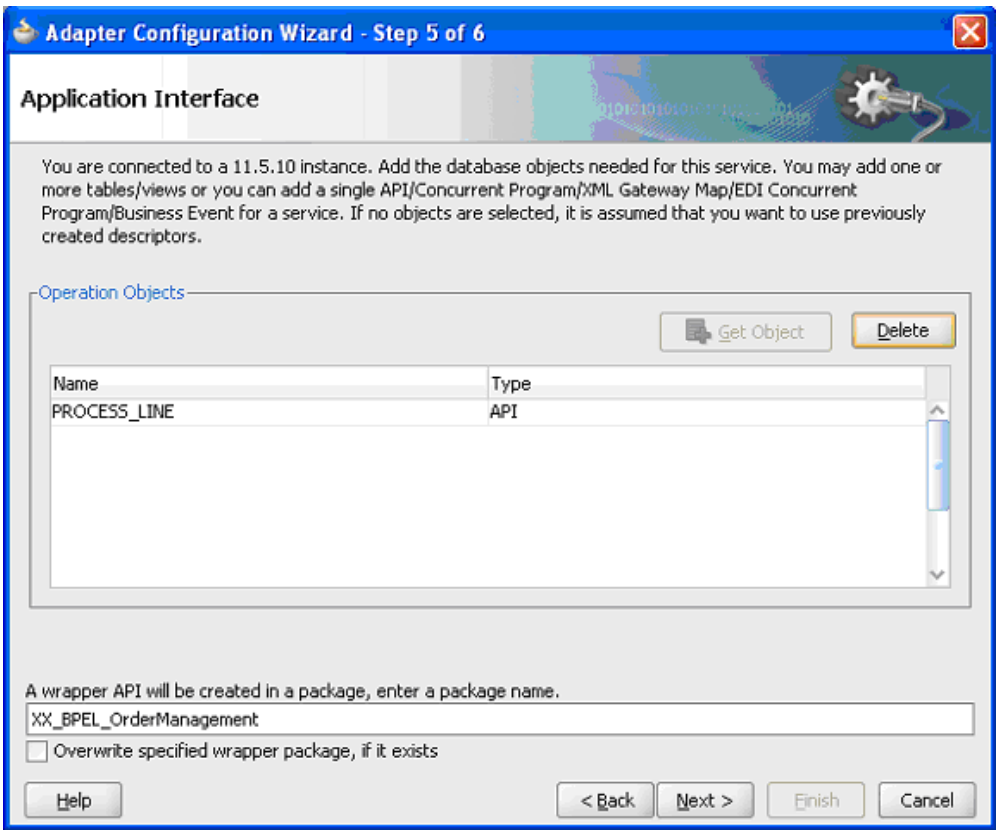
  This selection applies separately to each PL/SQL package, and remains in effect only for the duration of the current JDeveloper session.

Select the required PL/SQL API. For example, select `Process Order Line (PROCESS_LINE)` API from *Product Families > Supply Chain Management (SCM_PF)> Supply Chain Trading Connector(CLN) > Sales Order (ONT_SALES_ORDER) > PLSQL > Process Order API(OE_ORDER_PUB) > Process Order Line (PROCESS_LINE)*.

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 a PL/SQL 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 PL/SQL API while searching.

6. Click **OK**.

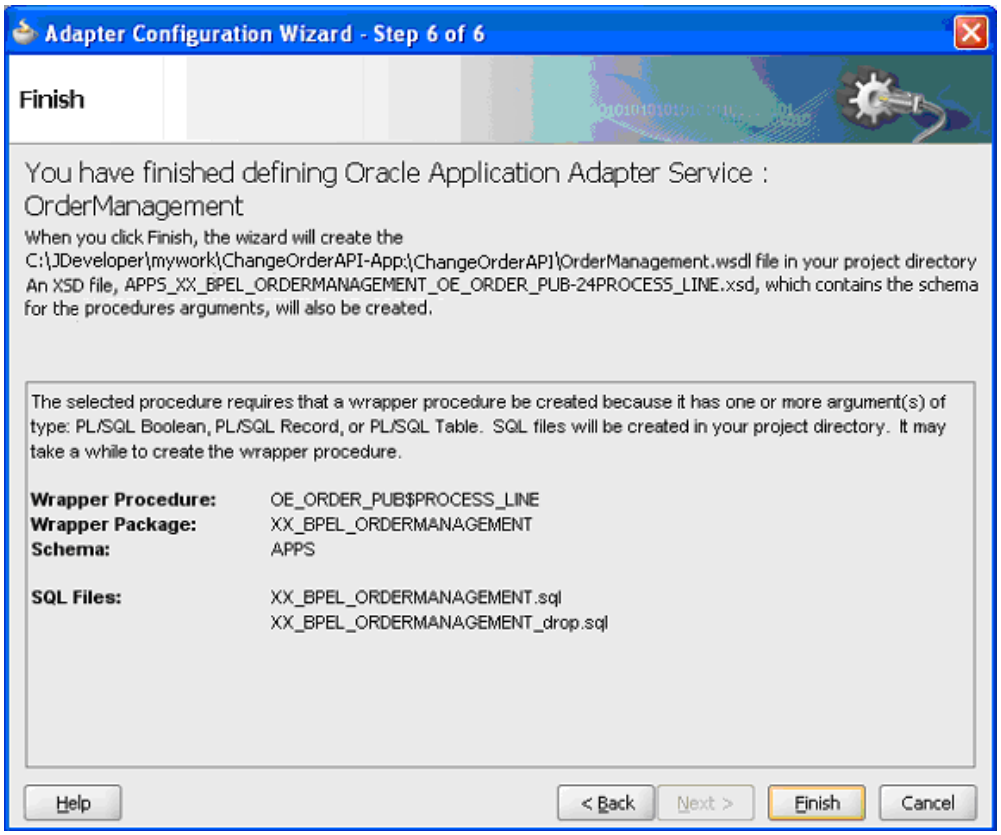
Adapter Configuration Wizard - Application Interface Page



Please note that the input and the output parameters of the Stored Procedure contains complex data types that are not readily mapped to JDBC types. The Adapter for Oracle Applications wizard provides a mechanism that detects when these types are used and then invokes Oracle JPublisher to generate the necessary wrappers automatically. Oracle JPublisher generates two SQL files, one to create schema objects, and another to drop them. The SQL that creates the schema objects is automatically executed from within the wizard to create the schema objects in the database schema before the XSD is generated.

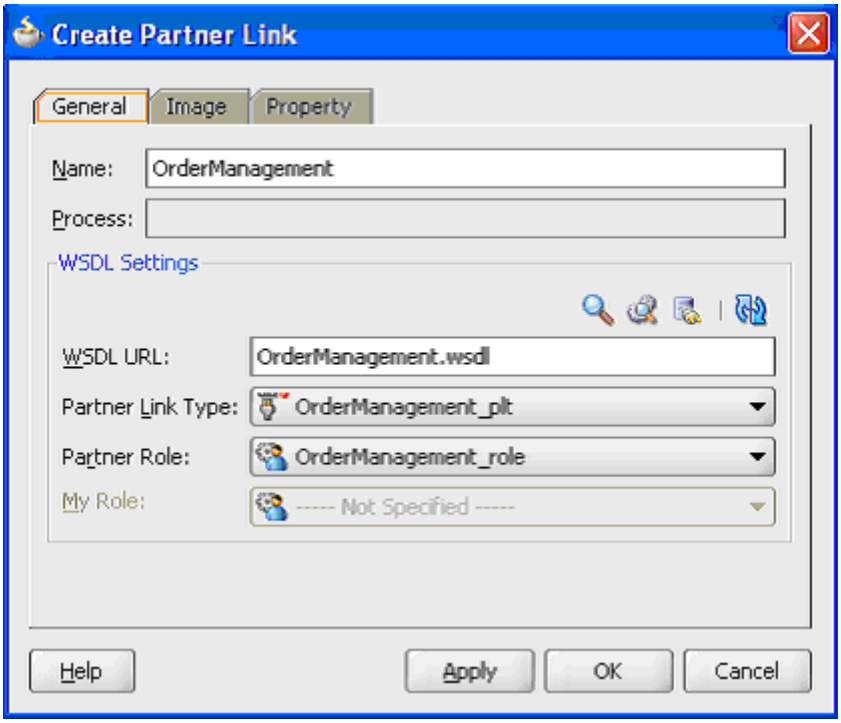
The Adapter Service points to the wrapper Stored Procedure instead of the Process Order Line Stored Procedure. The Adapter design-time automatically loads the wrapper procedure onto the underlying database.

7. Click **Next**, then click **Finish** to complete the process of configuring 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.

Partner Link Information



8. Click **Apply** and then **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.

Adding a Partner Link for File Adapter

Use this step to configure a BPEL process by synchronously reading an existing purchase order to obtain the order details.

To add a Partner Link for File Adapter to read order details:

1. In JDeveloper BPEL Designer, click **BPEL Services** in the Component palette.

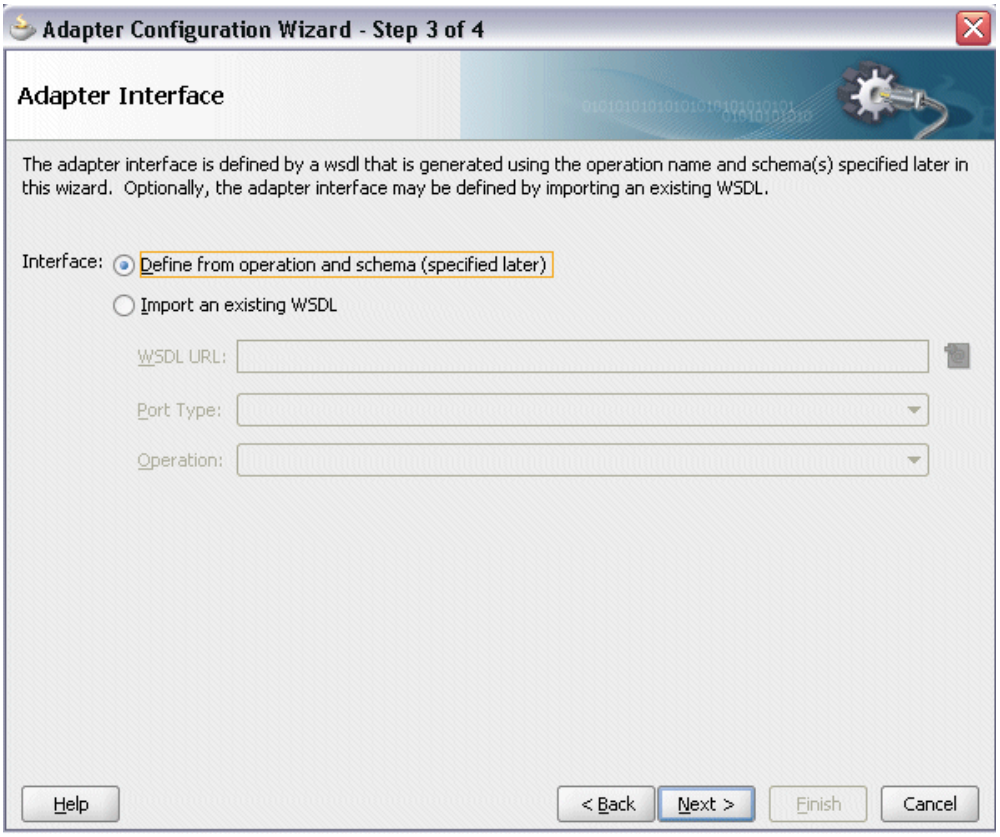
   Drag and drop **File Adapter** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears.

   Click **Next**.

2. In the Service Name page, enter a name for the file adapter service, such as `getOrderDetails`.

3. Click **Next**. The Adapter Interface page appears.

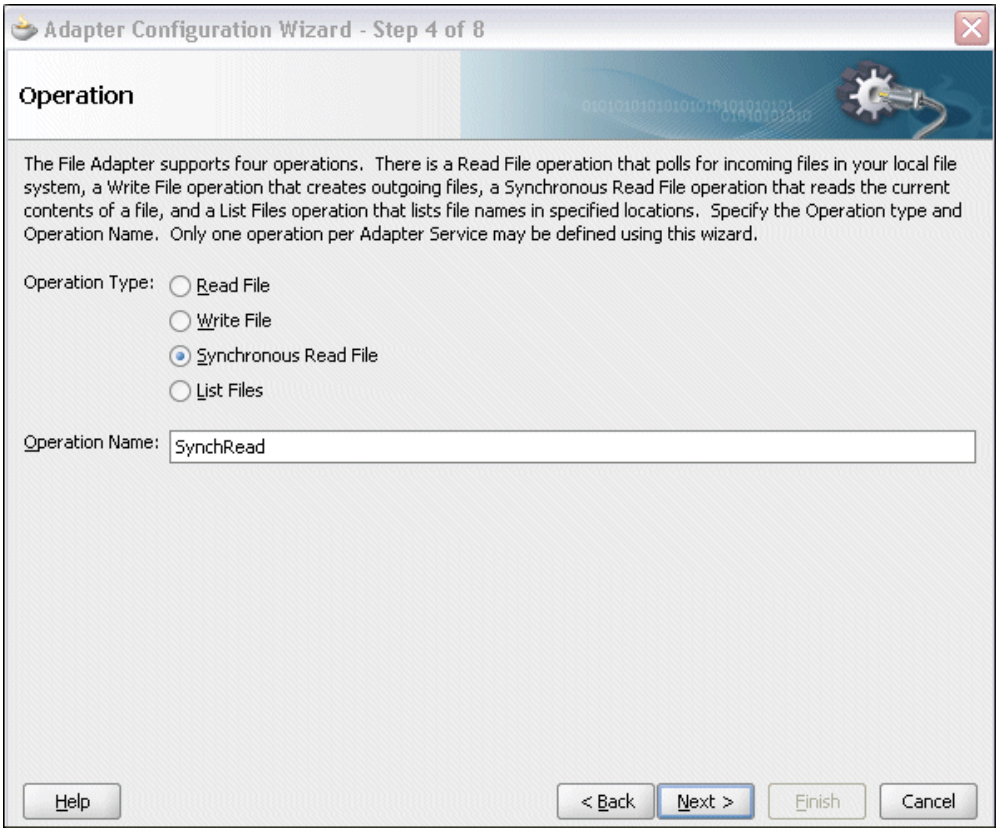
Specifying the Adapter Interface



Select the **Define from operation and schema (specified later)** radio button and click **Next**.

4. In the Operation page, specify the operation type. For example, select the **Synchronous Read File** radio button. This automatically populates the **Operation Name** field.

Specifying the Operation

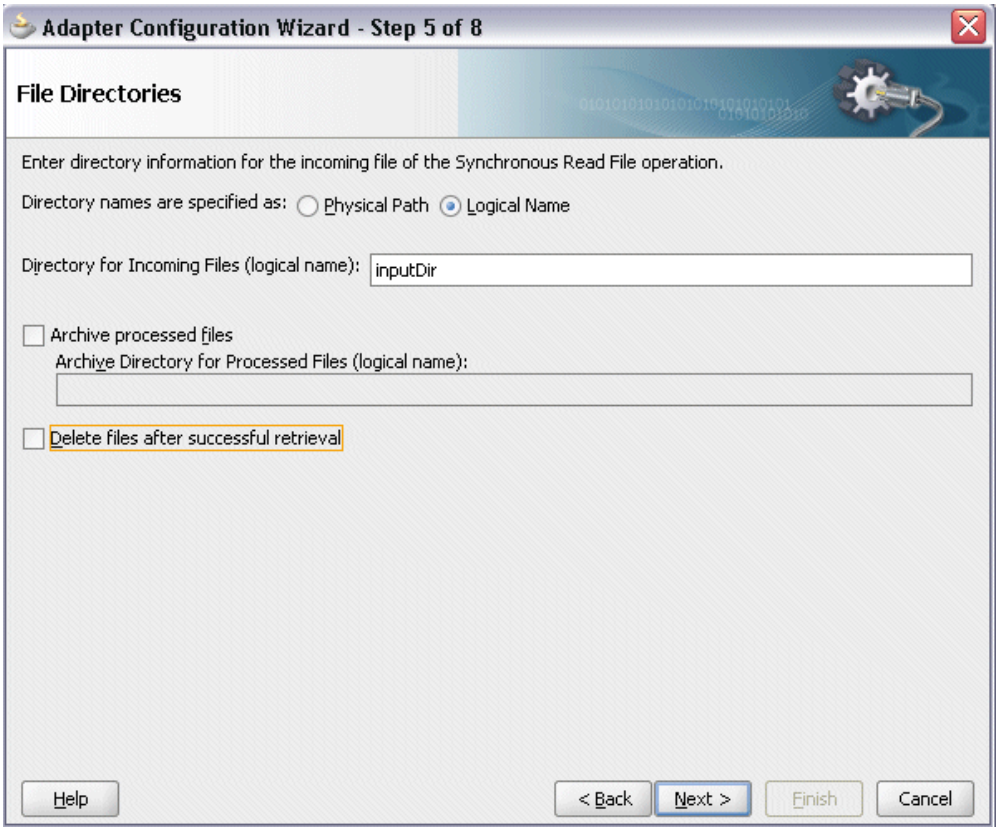


Click **Next** to access the File Directories page.

5. Select the **Logical Name** radio button and specify directory for incoming files, such as `inputDir`.

   Ensure the **Delete files after successful retrieval** check box is not selected.

Configuring the Input File



Click **Next** to open the File Name page.

6. Enter the name of the file for the synchronous read file operation. For example, enter `changeorder_data.xml`. Click **Next** to open the Messages page.

7. Select the 'browse for schema file' icon on the top right corner to open the Type Chooser window.

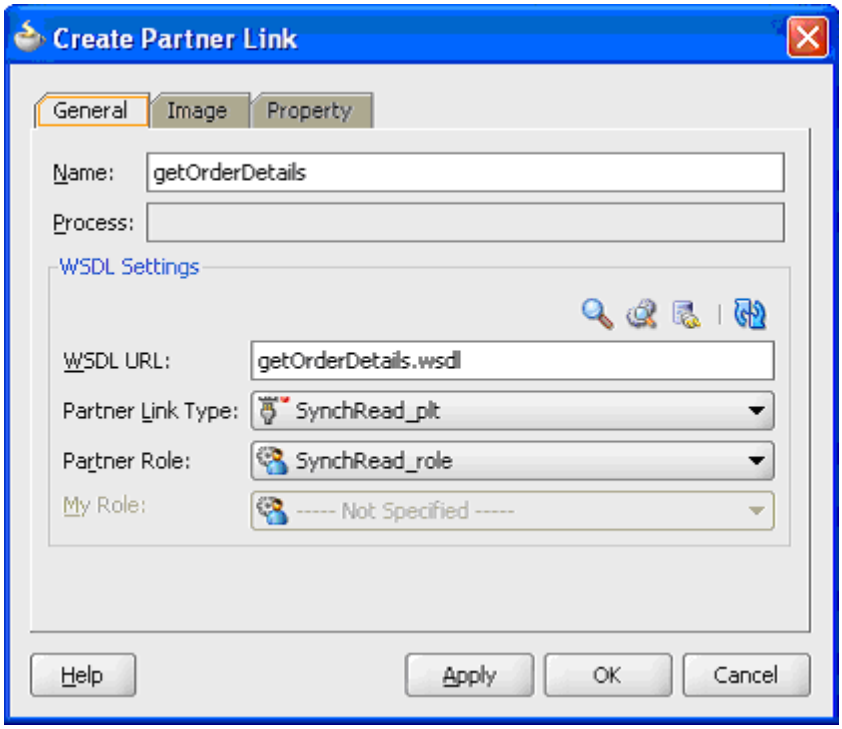
   Click Type Explorer and select *Project WSDL Files* > *OrderManagement.wsdl* > *Inline Schemas* > *schema* > *InputParameters*.

   The selected schema information will be automatically populated in the URL and Schema Element fields.

Specifying Message Schema

8. Click **Next** and then **Finish**. The wizard generates the WSDL file corresponding to the partner link. The main Create Partner Link dialog box appears, specifying the new WSDL file `getOrderDetails.wsdl`.

Completing the Partner Link Configuration



Click **Apply** and **OK** to complete the configuration and create the partner link with the required WSDL settings for the File Adapter service.

The `getOrderDetails` Partner Link appears in the BPEL process diagram.

Defining Wrapper APIs

The Adapter Configuration wizard generates a wrapper API when a PL/SQL API has arguments of data types, such as 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.

Following is a sample code of a PL/SQL 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 PL/SQL 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 PL/SQL 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

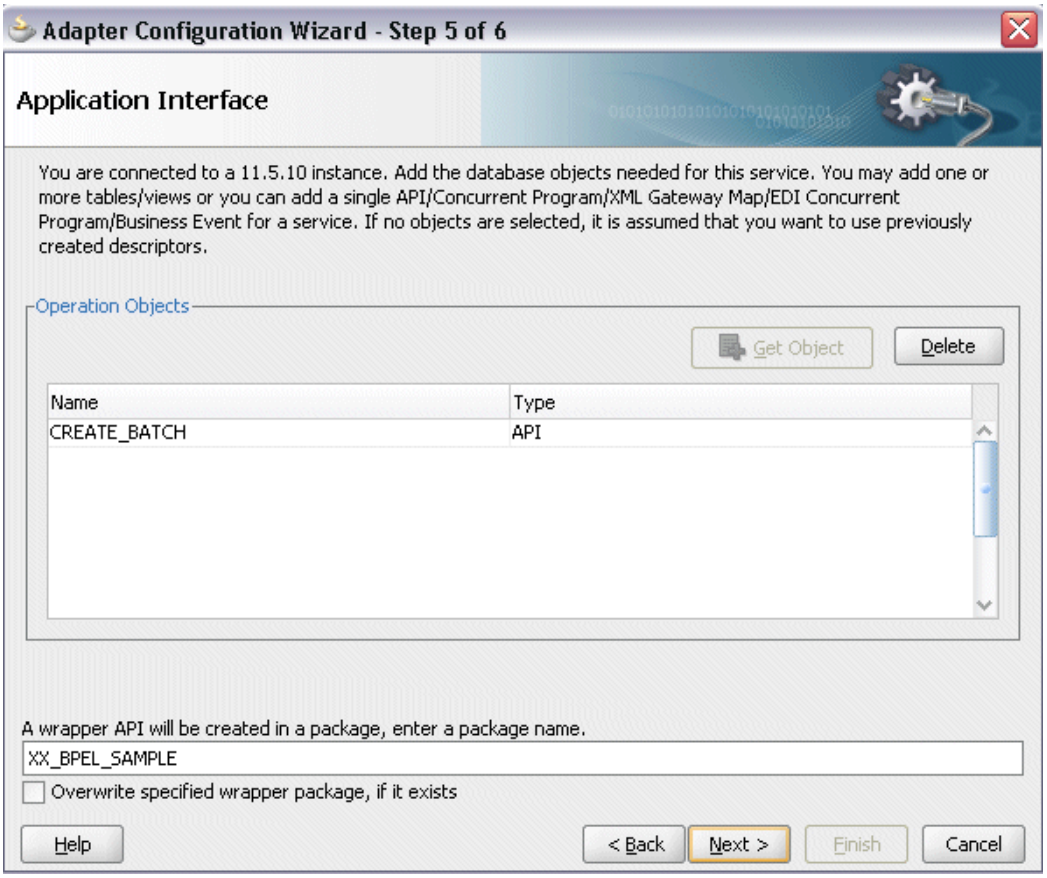
The generated SQL file that creates the wrapper API also creates the required schema objects. The types of the wrapper API's parameters will be those of the new schema object types. The wrapper package will contain conversion APIs to convert between the base PL/SQL type and the new schema object types.

> **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_*servicename.servicename* is the name of the service that you entered in Step 2 of Adding a Partner Link, page 8-9. 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 PL/SQL 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.

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 or the wrapper API name is longer than the maximum limit, it will be truncated accordingly.

The name of the wrapper API depends on whether the PL/SQL API that was originally selected is in a package or not. If the original PL/SQL 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 PL/SQL API is in a package, then the name of the wrapper API will be *original_package_name*$*original_api_name.*

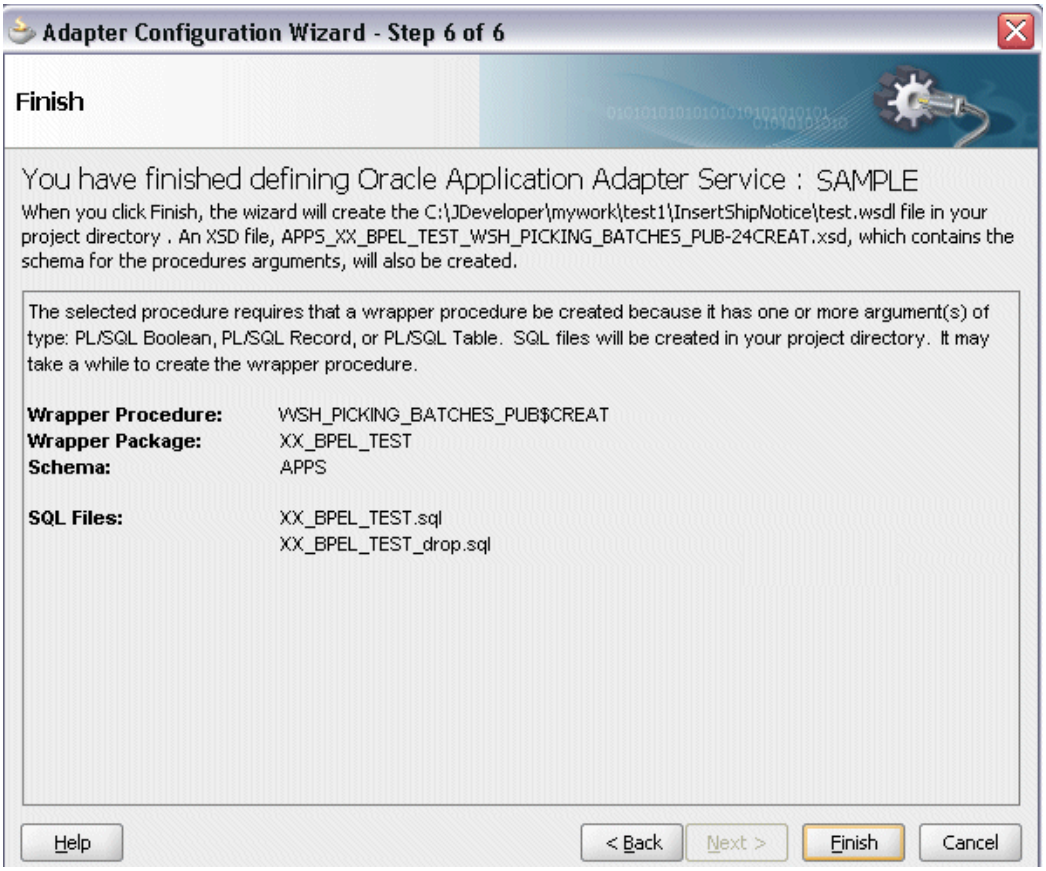
Wrapper APIs follow the naming convention of Oracle JPublisher. For example, if the original PL/SQL API was called `CREATE_EMPLOYEE` and was a root-level API, then the wrapper API would be named `TOPLEVEL$CREATE_EMPLOYEE`. If the original PL/SQL API is in a package called `EMPLOYEE`, then the wrapper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

The Finish page of the wizard is different when a wrapper API needs to be created. 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 PL/SQL APIs in the same package would be much shorter.

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 a PL/SQL API which takes `REF CURSOR` type, see the section "Support for REF CURSOR" in *Oracle BPEL Process Manager Developer's Guide* on OTN.

Overloaded APIs are not supported in release 11.5.10.

Declaring Parameters with a DEFAULT Clause

You can declare parameters of a stored procedure with a DEFAULT clause, so that when you invoke the procedure without that parameter, BPEL Process Manager 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 ways:

- `addEmployee ('John Smith') // country => 'US'`
- `addEmployee ('John Smith', 'France') // country => 'France'`

Omitting Parameters With a DEFAULT Clause

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 of input and runtime invocation.

Input

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

Runtime Invocation

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

If the input includes a value for the defaulted parameter, the value in the input will be used, rather than the default, as follows:

Input

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

Runtime Invocation

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

Omitting Parameters Without a DEFAULT Clause

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. Following is an example of a declaration where neither parameter has a DEFAULT clause:

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

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

In the current BPEL Process Manager 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`.

DEFAULT Clause Handling in Wrapper Procedures:

If a procedure contains a special type requiring a wrapper to be generated, 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, 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" .../>
```

Configuring the Invoke Activities

This step is to configure three Invoke activities:

1. To get the order details that is received from the Receive activity by invoking the `getOrderDetails` partner link in an XML file.

2. To initialize the existing purchase order by invoking `initLineRec` partner link.

3. To update the purchase order quantity information to Oracle Applications by invoking `OrderManagement` partner link.

To add the first Invoke activity for a partner link to get order details:

1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the first **Invoke** activity into the center swim lane of the process diagram, between the **receiveInput** and **callbackClient** activities.

2. Link the Invoke activity to the `getOrderDetails` service. The Edit Invoke dialog appears.

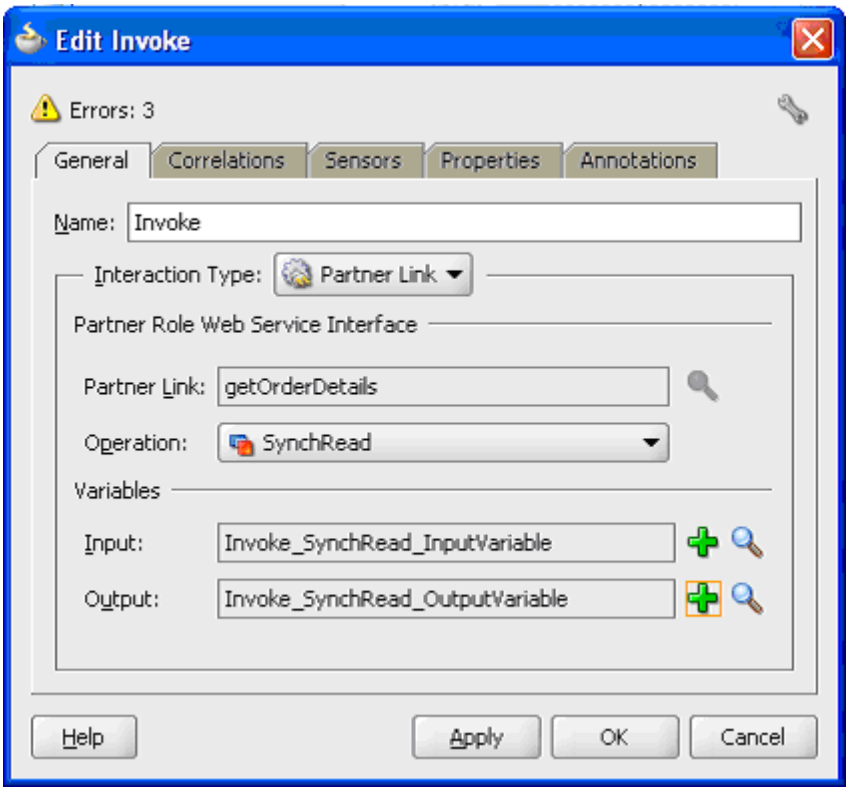
3. Enter a name for the Invoke activity, then click the **Create** icon next to the **Input Variable** field to create a new variable. The Create Variable dialog box appears.

4. Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK**.

5. In the Edit Invoke dialog, click the **Create** icon next to the **Output Variable** field to

create a new variable. The Create Variable dialog box appears.

Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK**.



Click **Apply** and then **OK** in the Edit Invoke dialog to finish configuring the Invoke activity.

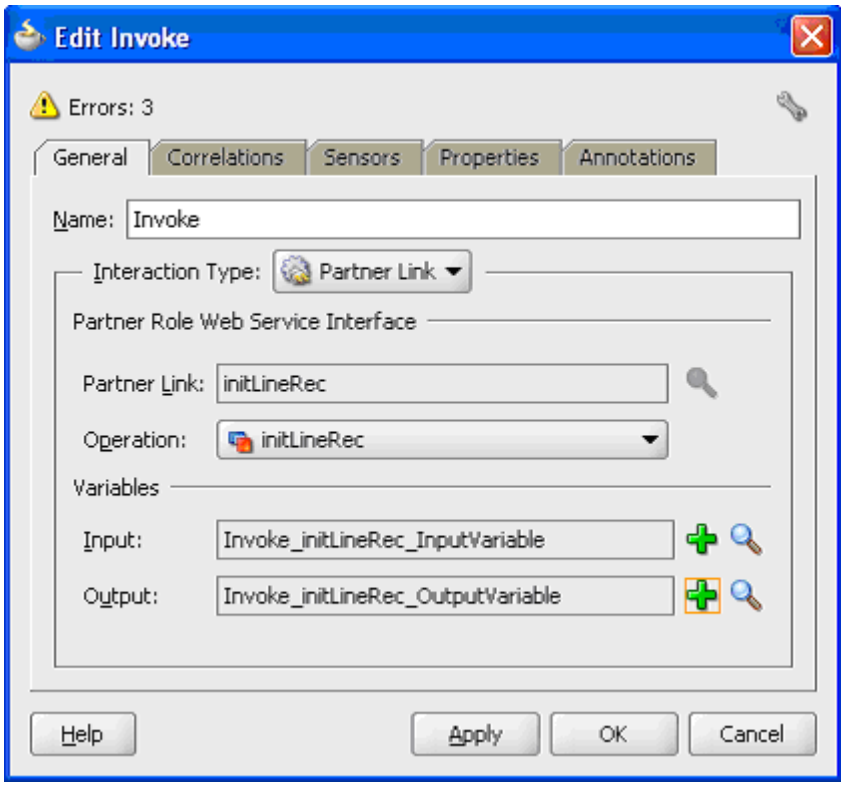
The Invoke activity appears in the process diagram.

To add the second Invoke activity for a partner link to initialize the existing purchase order:

1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the second **Invoke** activity into the center swim lane of the process diagram, between the first **Invoke** and **callbackClient** activities.

2. Link the Invoke activity to the `initLineRec` service. The Edit Invoke dialog box appears.

3. Repeat Step 3 to Step 5 described in the first Invoke activity procedure to complete the second Invoke activity.

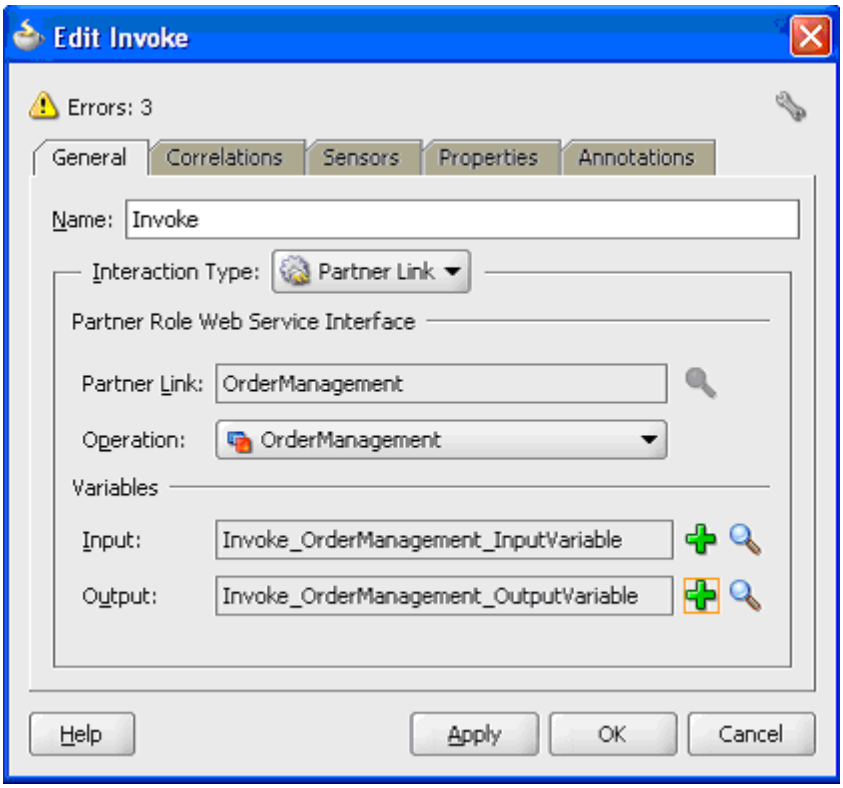


To add the third Invoke activity for a partner link to update the order quantity information to Oracle Applications:

1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the third **Invoke** activity into the center swim lane of the process diagram, between the second **Invoke** and **callbackClient** activities.

2. Link the Invoke activity to the `OrderManagement` service. The Edit Invoke dialog box appears.

3. Repeat Step 3 to Step 5 described in the first Invoke activity procedure.



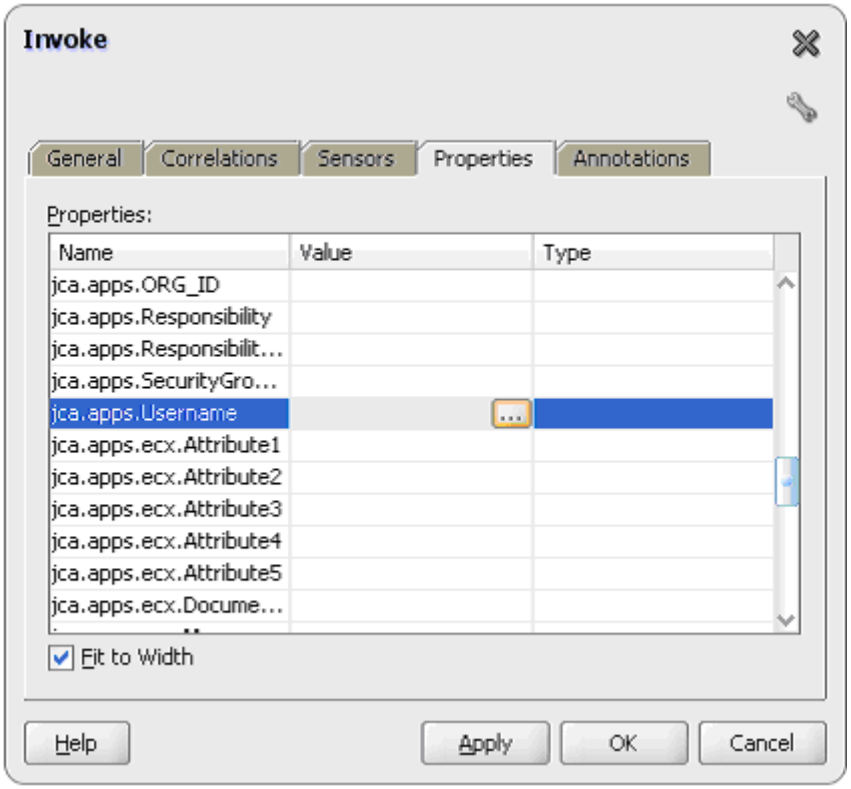
4. **Setting Header Properties for Application Context**

   Use the following steps to set the header message properties required to pass application context required to complete the BPEL process:

   1. Select the Properties tab in the Edit Invoke dialog box.

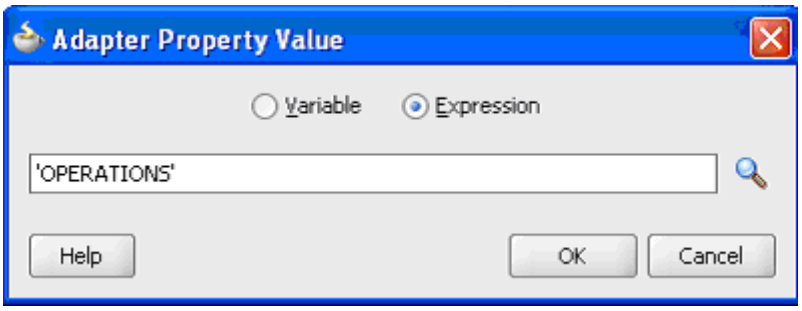
   2. Scroll down to locate the `jca.apps.Username` property from the list and double click the associated value field to enable the **Adapter Property Value** icon.

Setting Header Message Properties



3. Click the icon to open the Adapter Property Value dialog for the selected `jca.apps.Username` property.

Entering the Header Message Property Value



4. Select the **Expression** radio button and enter `'OPERATIONS'` as the property value.

   Click **OK**.

5. Repeat Step 2 to Step 4 to assign `'Order Management Super User, Vision Operations (USA)'` for `jca.apps.Responsibility`.

5. Click **Apply** and then **OK** to complete the Invoke activity.

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.

To configure the Transform activity:

Before add a Transform activity, use the following steps to add variables:

1. From the **structure** panel, select `ChangeOrderAPI.bpel` > Variables > Process > Variables folder. Right click on Variables folder and select **Create Variable** from the drop-down menu.

2. In the Create Variable dialog, enter `temp_miss_line` in the variable Name field.

3. Select the **Message Type** radio button and click the **Browse Message Type** icon to open the Type Chooser.

   Select `args_in_msg` from *Message Types* > *Project WSDL Files* > *OrderManagement.wsdl* > *Message Types*. Click OK to close the Type Chooser.

   The selected information will be populated in the Message Type field in the Create Variable dialog.

Use the following step to configure the Transform activity:

1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop **Transform** into center swim lane of the process map window. The **Transform** activity should be placed in between the second and third**Invoke** activities.

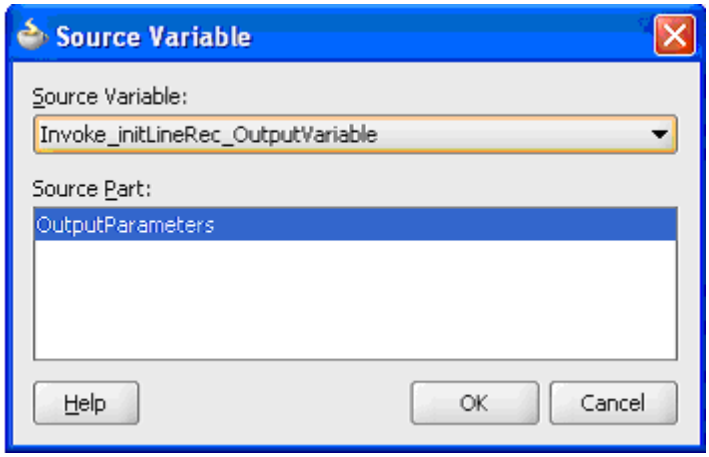
   This Transform activity is used to set the output of `initLineRec` service to the input of the `OrderManagement` service.

2. Double-click **Transform** in the process map to open the Transform dialog. The Transformation tab is selected by default.

   Click the General tab to name this Transform activity as `set_Miss_Line`.

3. In the Transformation tab, click the **Create** icon to open the Source Variable dialog.

Source Variable Dialog



Select the `Invoke_initLineRec_OutputVariable` from the Source Variable drop-down list, and select `OutputParameters` element as the Source Part value.

Click **OK**.

4. Select the `Invoke_OrderManagement_InputVariable` from the Target Variable drop-down list. The Target Part value of the variable is also selected.

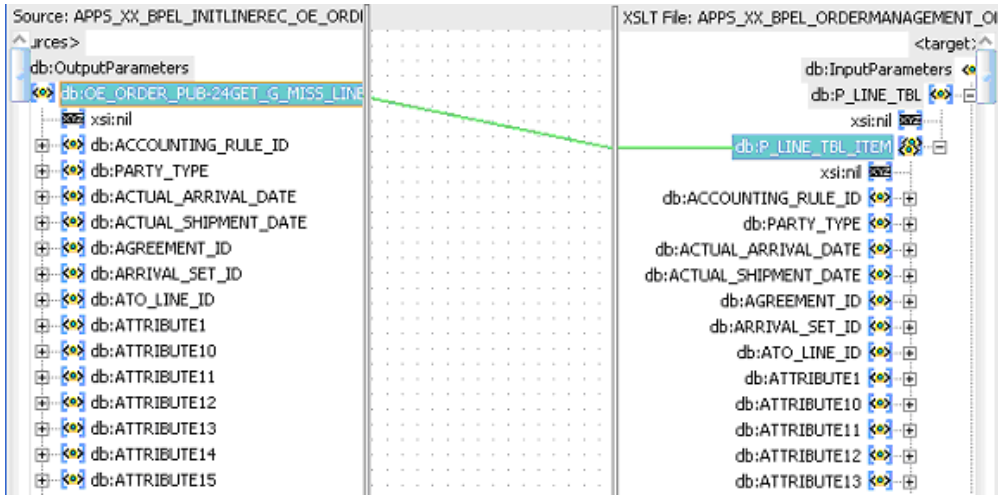
5. Click the **Create** icon next to the Mapper File field to create a new transformation mapping file. Mapper File specifies the file in which you create the mappings using the XSLT Mapper Transformation tool.

6. The transformation mapping file appears. The **Design** view appears by default.

7. 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.

   For example, link `db:OE_ORDER_PUB-24GET_G_MISS_LINE` from the source variable to `db:P_LINE_TBL_ITEM` as the target variable.

Variable Mapping



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.

8. Click **OK** to create the mapping from source variable to target variable.

Configuring an Assign Activity

Use the Assign activity to pass the output of `getOrderDetails` service and `initLineRec` service as an input to the `OrderManagement` service.

To add an Assign activity:

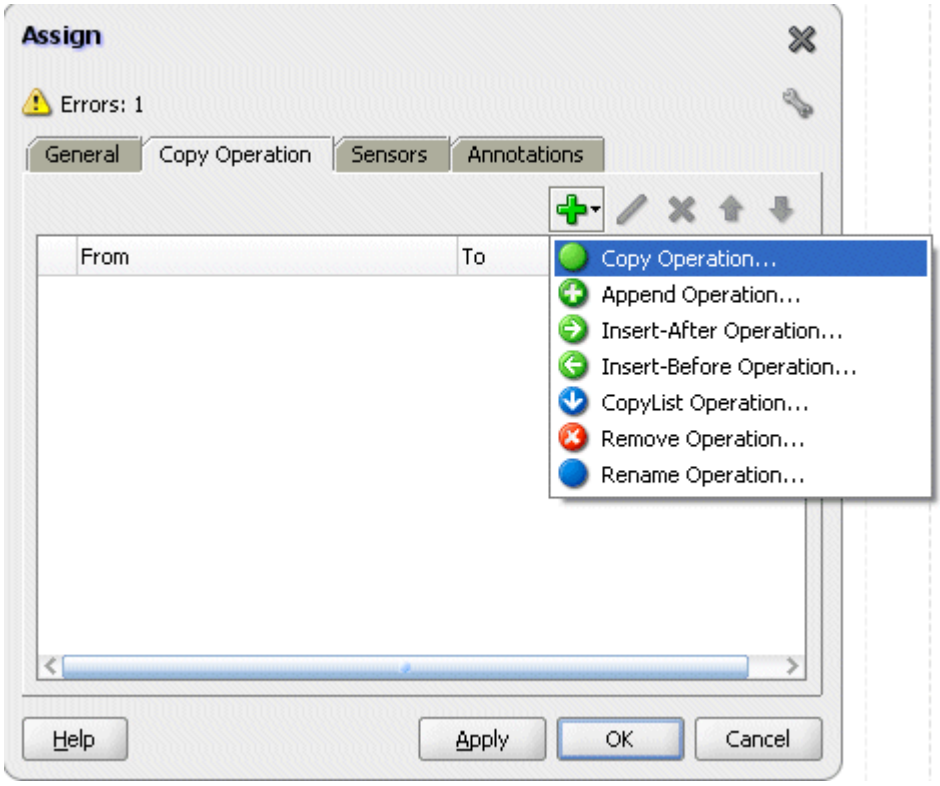
1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the **Assign** activity into the center swim lane of the process diagram right after the **Transform** activity that you just created earlier.

2. Double-click the **Assign** activity to access the Edit Assign dialog.

   Click the General tab to enter a name for the Assign activity. For example, `setOrderChanges`.

3. Select the Copy Operation tab, click the 'Plus' sign icon and select **Copy Operation** from the menu. The Create Copy Operation window appears.

Specifying a Copy Operation Action

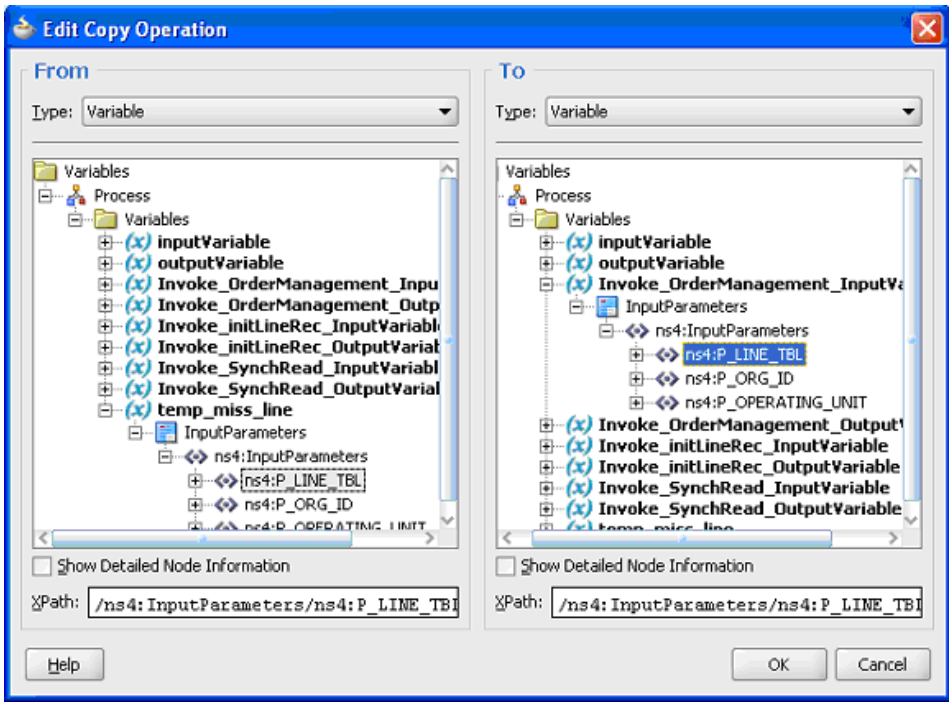


4. Enter the first pair of parameters:

   - In the From navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Temp_miss_line >Input Parameters** and select **ns4:P_LINE_TBL**.

     The XPath field should contain your selected entry.

   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_OrderManagement_InputVariable > Input Parameters** and select **ns4:P_LINE_TBL**. The XPath field should contain your selected entry.



- Click **OK**. The Edit Assign dialog box appears.

5. Repeat Step 3 to 4 to enter the second pair of parameters:
   - In the From navigation tree, select type Expression. Enter `'UPDATE'` in the Expression field.
   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_OrderManagement_InputVariable > Input Parameters > P_LINE_TBL > P_LINE_TBL_ITEM** and select **ns4:OPERATION**. The XPath field should contain your selected entry.
   - Click **OK**. The Edit Assign dialog box appears.

6. Repeat Step 3 to 4 to enter the third pair of parameters:
   - In the From navigation tree, select type Expression. Enter `'Update Order Quantity'` in the Expression field.
   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_OrderManagement_InputVariable > P_LINE_TBL > P_LINE_TBL_ITEM** and select **ns4:CHANGE_REASON**. The XPath field should contain your selected entry.
   - Click **OK**. The Edit Assign dialog box appears.

7. Repeat Step 3 to 4 to enter the fourth pair of parameters:

   - In the From navigation tree, select type Variable. Navigate to **Variable > Process > Variables > Invoke_SyncRead_OutputVariable>Input Parameters > P_LINE_TBL >P_LINE_TBL_ITEM** and select **ns4:LINE_ID**. The XPath field should contain your selected.

   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_OrderManagement_InputVariable > P_LINE_TBL > P_LINE_TBL_ITEM** and select **ns4:LINE_ID**. The XPath field should contain your selected entry.

   - Click **OK**. The Edit Assign dialog box appears.

8. Repeat Step 3 to 4 to enter the fifth pair of parameters:

   - In the From navigation tree, select type Variable. Navigate to **Variable > Process > Variables > Invoke_SyncRead_OutputVariable>Input Parameters > P_LINE_TBL >P_LINE_TBL_ITEM** and select **ns4:ORDERED_QUANTITY**. The XPath field should contain your selected.

   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_OrderManagement_InputVariable > P_LINE_TBL > P_LINE_TBL_ITEM** and select **ns4:ORDERED_QUANTITY**. The XPath field should contain your selected entry.

   - Click **OK**. The Edit Assign dialog box appears.

9. Click **Apply** and then **OK** to complete the configuration of the Assign activity.

Run-Time Tasks for PL/SQL APIs

After designing the BPEL process, the next step is to deploy, run and monitor it.

1. Deploy the BPEL Process., page 8-43

2. Test the BPEL Process., page 8-45

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 application server (Oracle WebLogic Server) that you have established the connection.

Prerequisites

- You must have established the connectivity between the deign-time environment

and an application server.

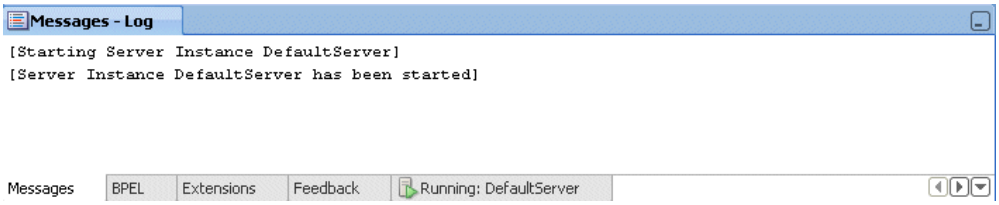
For more information, see Configuring the Data Source in Oracle WebLogic Server, page A-3 and Creating an Application Server Connection, page A-8.

- Oracle WebLogic Server has been started.

  Before deploying the BPEL process, you need to start the Oracle WebLogic Server that you have established the connection.

  If a local instance of the WebLogic Server is used, start the WebLogic Server by selecting **Run > Start Server Instance** from Oracle JDeveloper.

  Once the WebLogic Admin Server "DefaultServer" instance is successfully started, you should find that <Server started in Running mode> and DefaultServer started message in the Running:DefaultServer and Messages logs.

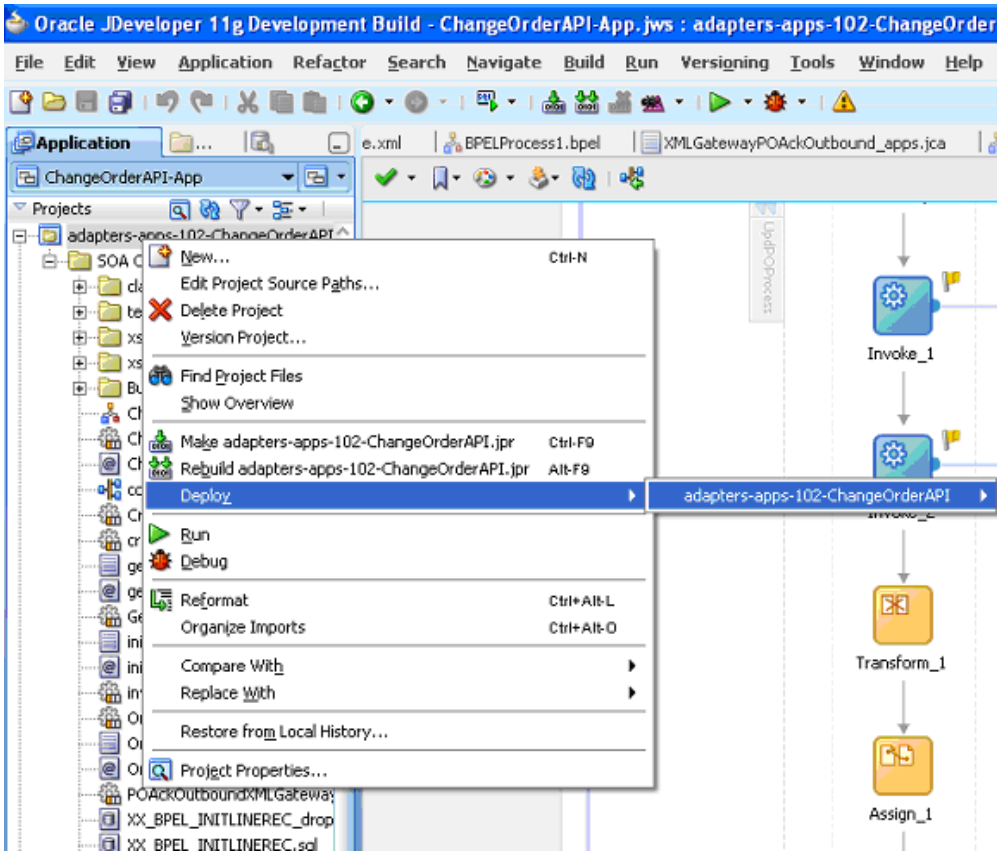


To deploy the BPEL process:

1. Select the BPEL project in the Applications window.

2. Right-click the project name, and then select **Deploy > [project name] > [serverConnection]** from the menu that appears.

Deploying the BPEL Process



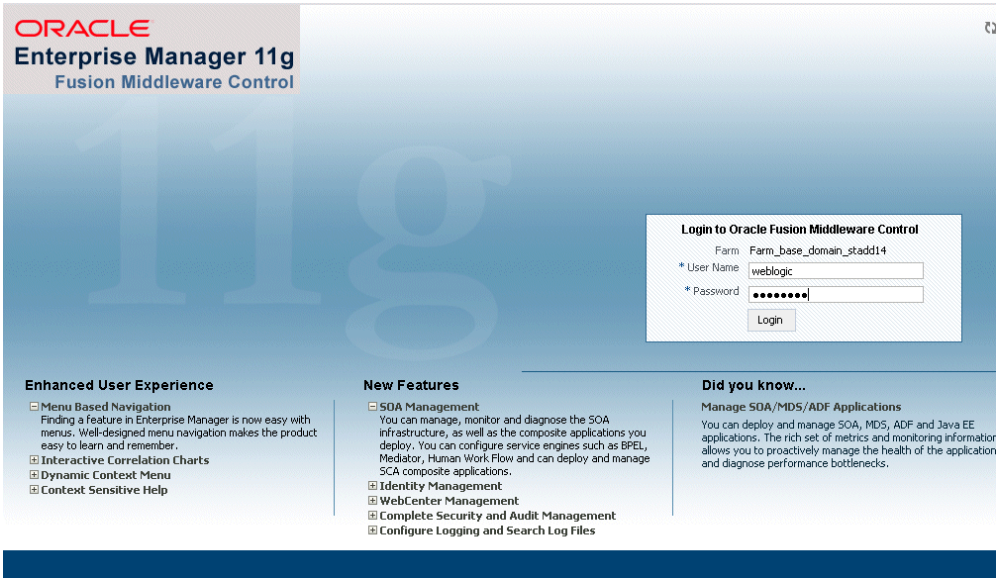
3. The BPEL process is compiled and deployed. You can check the progress in the Messages window.

Testing the BPEL Process

Once the BPEL process is deployed, you can manage and monitor the process from the Oracle Enterprise Manager Fusion Middleware Control Console. You can also test the process and the integration interface by manually initiating the process.

To test the BPEL process:

1. Navigate to Oracle Enterprise Manager Fusion Middleware Control Console (`http://<servername>:<portnumber>/em`). The composite you deployed is displayed in the Applications Navigation tree.



2. Enter username (such as `weblogic`) and password and click **Login** to log in to a farm.

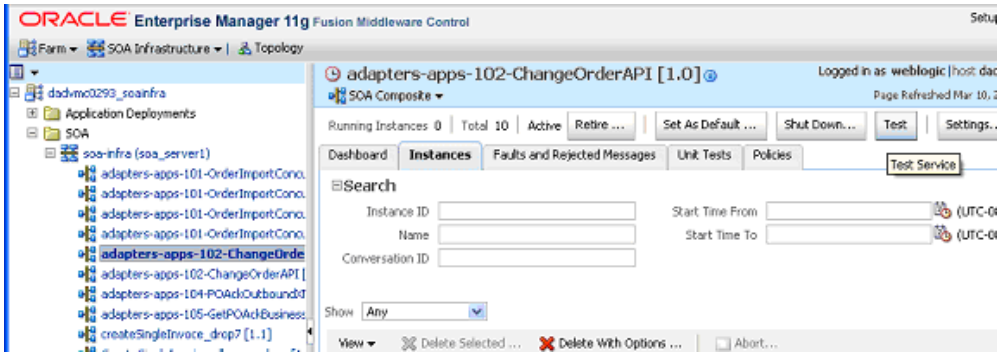
   You may need to select an appropriate target instance farm if there are multiple target Oracle Enterprise Manager Fusion Middleware Control Console farms.

3. From the Farm base domain, expand the **SOA >soa-infra** to navigate through the SOA Infrastructure home page and menu to access your deployed SOA composite applications running in the SOA Infrastructure for that managed server.

   > **Note:** The Farm menu always displays at the top of the navigator. As you expand the SOA folder in the navigator and click the links displayed beneath it, the SOA Infrastructure menu becomes available at the top of the page.

   Click the SOA composite application that you want to initiate (such as 'ChangeOrderAPI') from the SOA Infrastructure.

Viewing Deployed SOA Composites

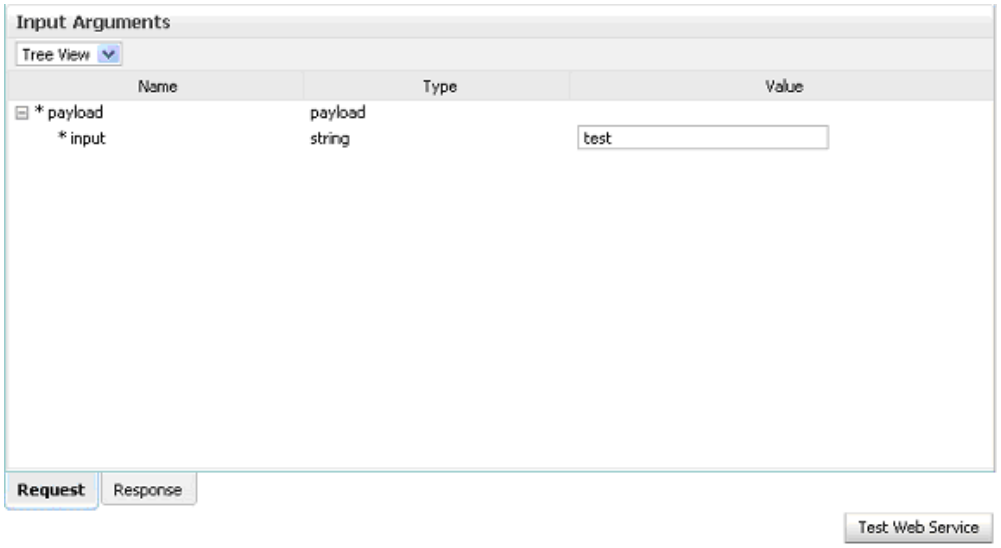


Click **Test** at the top of the page.

4. The Test Web Service page for initiating an instance appears. You can specify the XML payload data to use in the Input Arguments section.

   Enter the input string required by the process and click **Test Web Service** to initiate the process.

Testing Web Service



The test results appear in the Response tab upon completion.

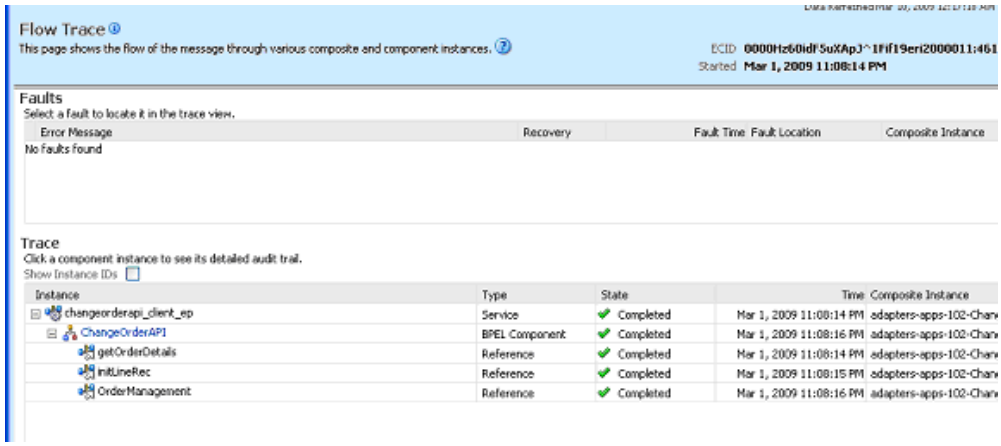
5. Click on the BPEL process name and then click the Instances tab. The SOA composite application instance ID, name, conversation ID, most recent known state of each instance since the last data refresh of the page are displayed.

   In the Instance ID column, click a specific instance ID to show the message flow

through the various service components and binding components. The Flow Trace page is displayed.

In the Trace section, you should find the sequence of the message flow for the service binding component (`changeorderapi_client_ep`), BPEL component (`ChangeOrderAPI`), and reference binding components (`getOrderDetails`, `initLineRec`, and `OrderManagement`). All involved components have successfully received and processed messages.

Flow Trace Page



Click your BPEL service component instance link (such as `ChangeOrderAPI`) to display the Instances page where you can view execution details for the BPEL activities in the Audit Trail tab.

Click the Flow tab to check the BPEL process flow diagram. Click an activity of the process diagram to view the activity details and flow of the payload through the process.

Validating Records Using SQL

To confirm that the records have been written into the Oracle Applications, you can write SQL `select l.ordered_quantity from oe_order_lines_all l,oe_order_headers_all h where h.orig_sys_document_ref='<ORDER_ID> and h.header_id=l.header_id;` statements and fetch the results showing the `ordered_quantity` should be the same as the value provided in the `changeorder_data.xml` document.

Verifying Records in Oracle Applications

Alternatively, you can go to the specific module in Oracle Applications with Oracle Management Super User, Vision Operation (USA) responsibility.

To validate it in Oracle Order Management:

1. Log on to the Forms-based Oracle Applications with the Order Management,

Super User responsibility.

2. Select Order Returns > Sales Order. Sales Order Forms would open up.

3. Search for an order by entering the order number in the Customer PO field. This would bring up the details of a newly created order.

4. Select the Line Items tab to check the quantity of the order. It should be the same as it is set in `changeorder_data.xml` file.

Troubleshooting and Debugging

If you experience problems with your PL/SQL API integration, you can take the following troubleshooting steps:

- If you designed your PL/SQL API integration in one instance of your Oracle Applications environment, and plan to run it in another instance, be sure to rerun the SQL scripts in the second instance to create the necessary PL/SQL wrappers.

- Ensure that the JNDI location in **weblogic-ra.xml** is properly configured.

If you still experience problems with your integration, you can enable debugging.

Enabling Debugging

You can enable debugging for PL/SQL APIs using the BPEL Process Manager.

To enable debugging:

1. Log into your BPEL Process Manager domain.

2. Select *yourdomain*`.collaxa.cube.ws`

3. Select **Debug.**

Debugging information is output to the log file for your domain. To examine the log file in the BPEL Process Manager, navigate to **Home > BPEL Domains >*yourdomain* > Logs.** The log file is ***yourdomain*.log**.

9

Using e-Commerce Gateway

This chapter covers the following topics:

- Overview of e-Commerce Gateway Integration
- Design-Time Tasks for e-Commerce Gateway
- Creating a New BPEL Project
- Adding a Partner Link
- Adding a Partner Link for File Adapter
- Configuring the Invoke Activities
- Configuring the Assign Activity
- Run-Time Tasks for e-Commerce Gateway
- Deploying the BPEL Process
- Testing the BPEL Process
- Verifying Records in Oracle Applications

Overview of e-Commerce Gateway Integration

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.

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.

> **Note:** For detailed information about Oracle e-Commerce Gateway, see *Oracle e-Commerce 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

Adapter for Oracle 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.

Design-Time Tasks for e-Commerce Gateway

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 Adapter for Oracle Applications to use e-Commerce Gateway. It describes the tasks required to configure Adapter for Oracle Applications using the Adapter Configuration Wizard in Oracle JDeveloper.

BPEL Process Scenario

This example uses IN: Ship Notice/Manifest (ECASNI) EDI concurrent program exposed as a Web service to support the inbound Ship Notice/Manifest transaction.

When a shipment request is received, the shipping details will be retrieved and then imported to the Purchasing system to create an electronic shipment notice as a pre-receipt.

If the BPEL process is successfully executed after deployment, you should be able to validate if the shipment notice has been inserted in Oracle Applications with the same shipment identifier number.

Prerequisites to Configure e-Commerce Gateway

Populating Application Context Header Variables

You need to populate certain variables in the BPEL PM in order to provide context information for Oracle Applications. The context information is required for an EDI transaction in order for an Oracle Applications user that has sufficient privileges to run the program.

The context is set taking into account the values passed for the header properties including *Username*, *Responsibility*, *Responsibility Application*, *Security Group*, and *NLS Language*. If the values for the new header properties *Responsibility Application*, *Security Group*, and *NLS Language* are not passed, context information will be determined based on *Username* and *Responsibility*.

The default Username is `SYSADMIN`, the default Responsibility is `SYSTEM ADMINISTRATOR`, the default Security Group Key is `Standard`, and the default NLS Language is `US`.

You can change the default values specified in the generated WSDL. 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 values. The context information can be specified by configuring an Invoke activity.

For more information about applications context, see Supporting for Normalized Message Properties, page 3-8.

Based on the scenario described earlier, the following design-time tasks are discussed in this chapter.

1. Create a new BPEL project, page 9-3
2. Add a partner link, page 9-8
3. Add a partner link for File Adapter, page 9-15
4. Configure the Invoke activities, page 9-20
5. Configure the Assign activity, page 9-25

Creating a New BPEL Project

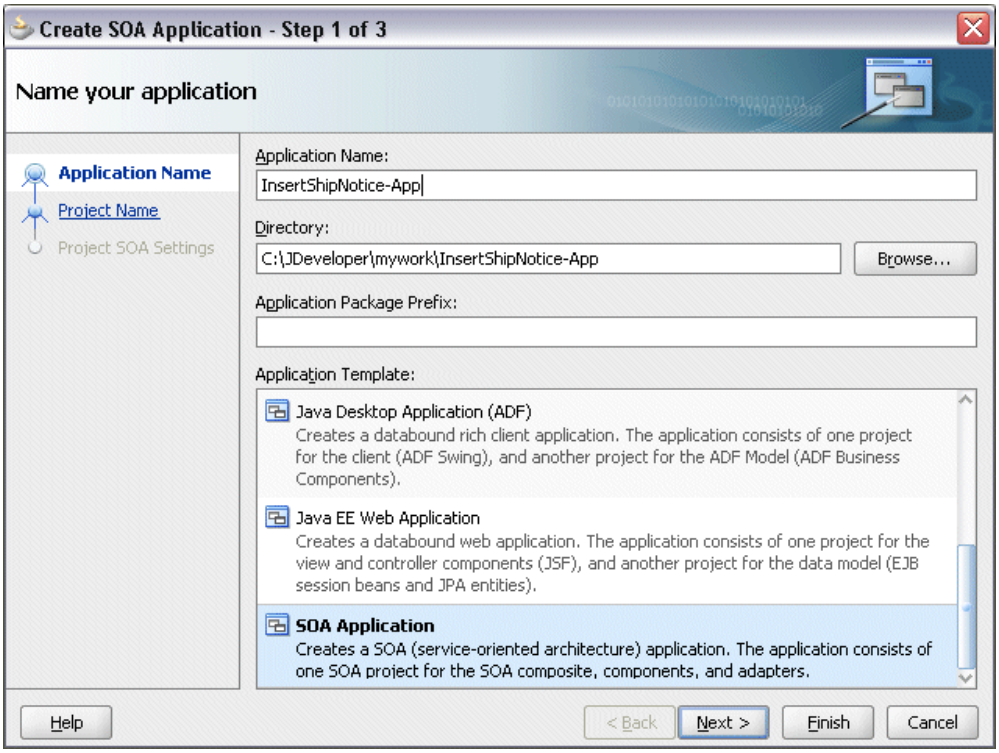
The first configuration task is to create a new BPEL project.

To create a new BPEL project:

1. Launch Oracle JDeveloper.
2. Click **New Application** in the Application Navigator.

The Create SOA Application - Name your application page is displayed.

The Create SOA Application - Name your application Page

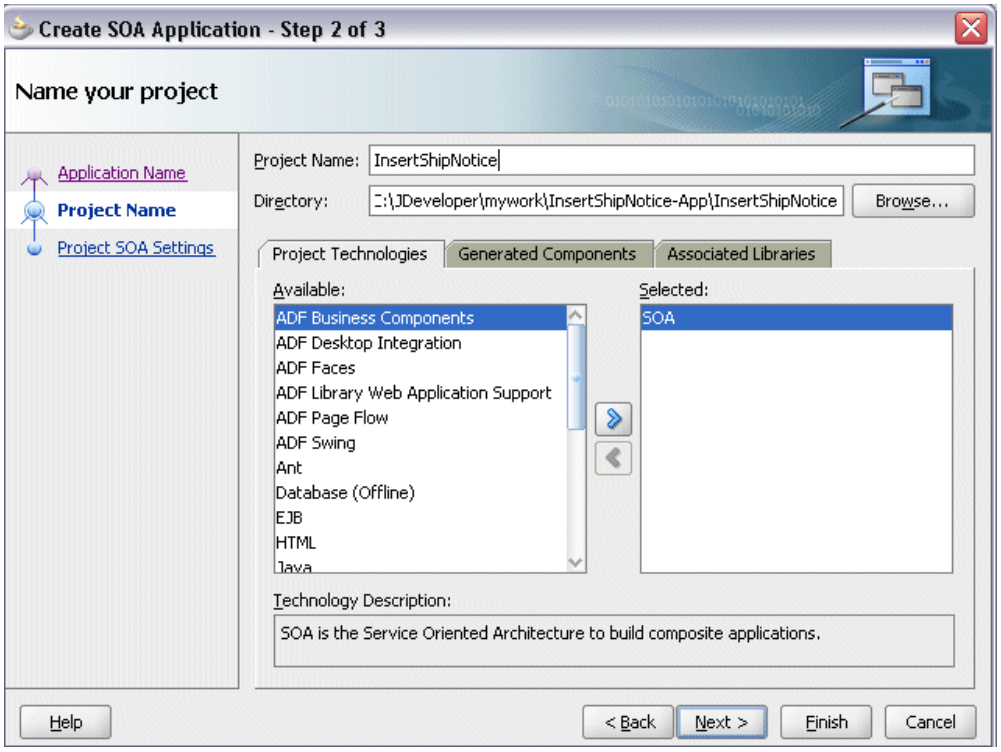


3. Enter an appropriate name for the application in the **Application Name** field and select **SOA Application** from the Application Template list.

   Click **Next**. The Create SOA Application - Name your project page is displayed.

4. Enter an appropriate name for the project in the **Project Name** field. For example, `InsertShipNotice`.

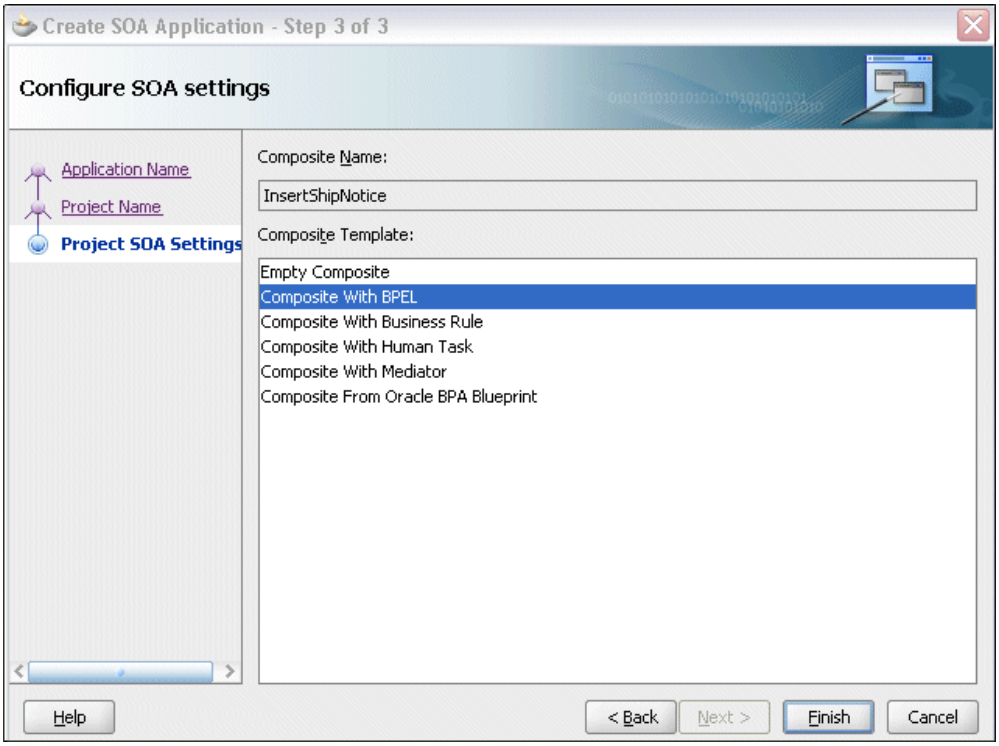
The Create SOA Application - Name your project Page



5. In the Project Technologies tab, ensure that **SOA** is selected from the Available technology list to the Selected technology list.

   Click **Next**. The Create SOA Application - Configure SOA settings page is displayed.

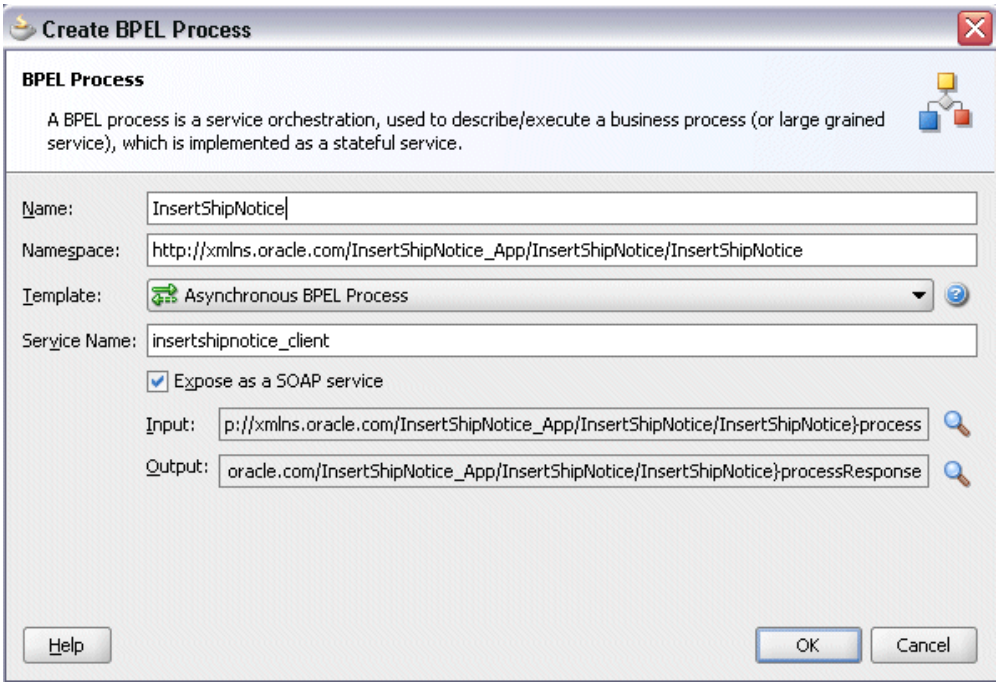
The Create SOA Application - Configure SOA settings Page



6. Select **Composite With BPEL** from the Composite Template list, and then click **Finish**. You have created a new application, and an SOA project. This automatically creates an SOA composite.

   The Create BPEL Process page is displayed.

The Create BPEL Process Page

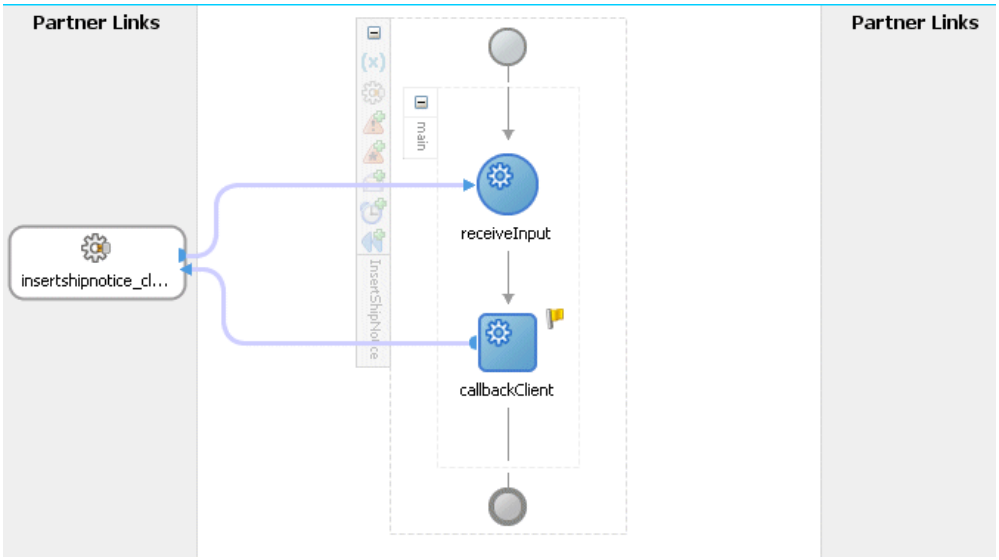


7. Enter an appropriate name for the BEPL process in the **Name** field. For example, `InsertShipNotice`.

   Select **Asynchronous BPEL Process** in the **Template** field. Click **OK**.

   An asynchronous BPEL process is created with the Receive and Reply activities. The required source files including bpel and wsdl, using the name you specified (for example, `InsertShipNotice.bpel` and `InsertShipNotice.wsdl`) and `composite.xml` are also generated.

New BPEL Process



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.

Based on the BPEL process scenario, you need to create a partner link to create a ship notice.

To add a partner link:

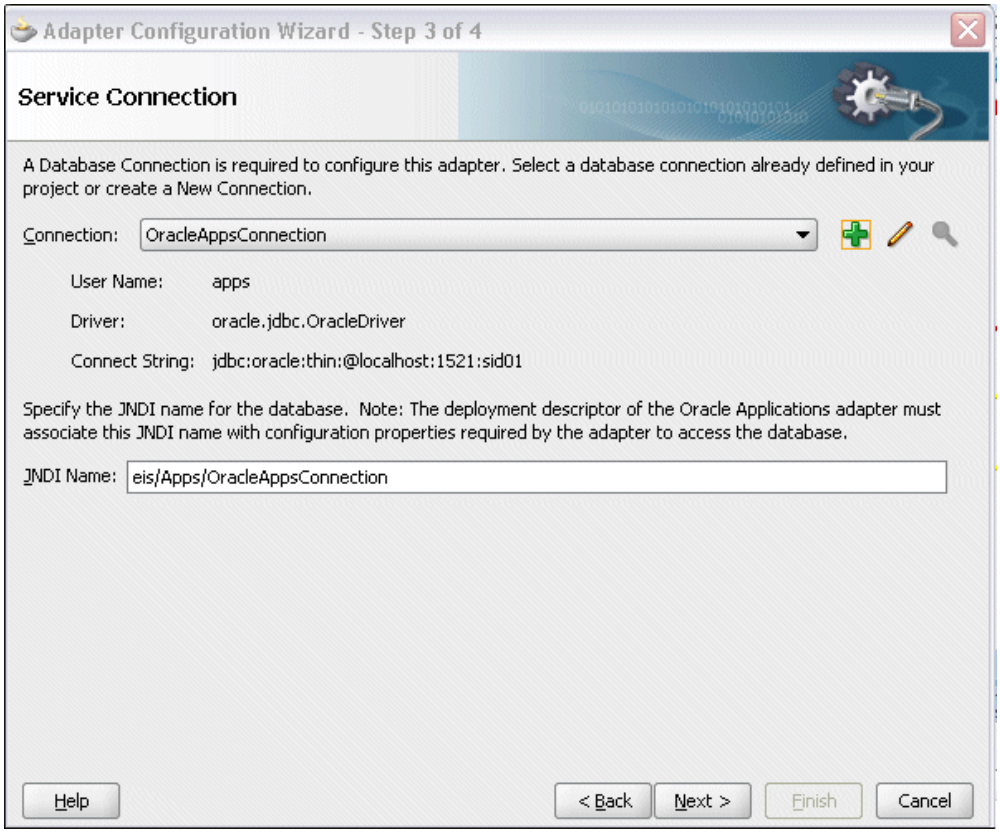
1. Click **BPEL Services** in the Component palette.

   Drag and drop **Oracle Applications** from the BPEL Services list into the right Partner Link swim lane of the process diagram. The Adapter Configuration Wizard Welcome page appears. Click **Next**.

2. Enter a service name in the **Service Name** field. For example, `InsertShipment`.

   Click **Next**. The Service Connection page appears.

Specifying a Database Service Connection



3. You can perform either one of the following options for your database connection:

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

   - You can create a new database connection by clicking the **Create a New Database Connection** icon.

     How to define a new database connection, see Create a New Database Connection, page 4-13.

   - You can select an existing database connection that you have configured earlier from the **Connection** drop-down list.

     The Service Connection page will be displayed with the selected connection information. The JNDI (Java Naming and Directory Interface) name corresponding to the database connection appears automatically in the **Database Server JNDI Name** field. Alternatively, you can specify a JNDI name.

Note: When you specify a JNDI name, the deployment descriptor of the Oracle Applications adapter must associate this JNDI name with configuration properties required by the adapter to access the database.

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.

Note: For more information about JNDI concepts, refer to *Oracle Fusion Middleware User's Guide for Technology Adapters.*

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

   Click **Next**.

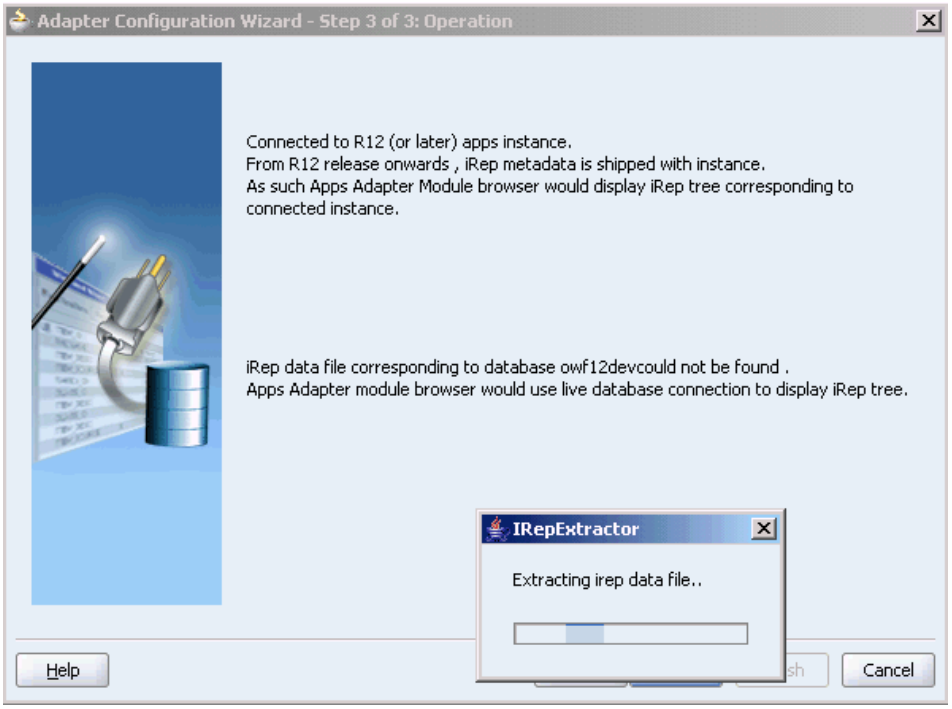
   For Oracle E-Business Suite Release 12:

   If you are connecting to Oracle E-Business Suite Release 12, then the **IREP File not present** dialog appears indicating that Adapter could not find the Oracle Integration Repository data file corresponding to the database you are connecting in your workspace. Absence of the data file would make browsing or searching of Integration Repository tree considerably slow. You can choose to extract the data file and create a local copy of the Integration Repository data file. Once it is created successfully, Adapter will pick it up automatically next time and retrieve data from your local Integration Repository.

   You can select one of the following options:

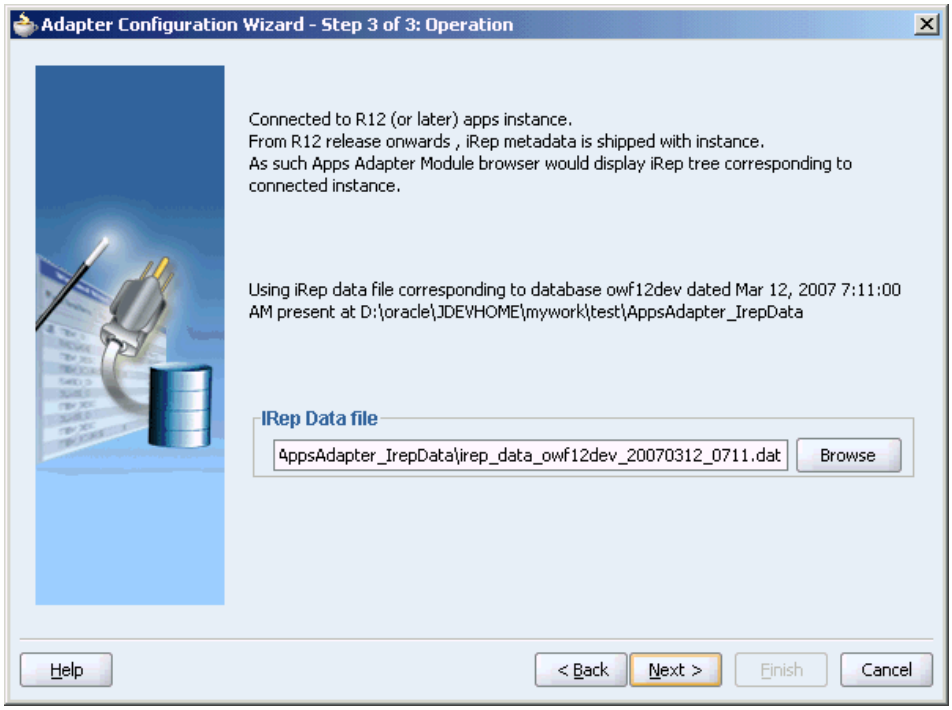
   - Click **Yes** to extract the Integration Repository data file.

Extracting Integration Repository Data File



After the system successfully creates a local copy of the Integration Repository data file, next time when you connect to the database, you will find the **IRep Data File** field appears in the Operation dialog indicating where your local copy exists with the creation date and time as part of the file name.

Using the Local Integration Repository Data File



- Click **No** to query the Integration Repository data file from the live database you are connecting to display the Integration Repository tree.

  Note: It is highly recommended that you create a local copy of the Integration Repository data file so that Adapter will query the data next time from the local copy in your workspace to enhance the performance.

  Click **Next** in the Operation page to open the Oracle Applications Module Browser.

For Oracle E-Business Suite pre-Release 11.5.10:

If you are connecting to a pre-11.5.10 Oracle Applications instance, you must select the interface type in the Adapter Configuration Wizard. Select **EDI Gateway** to proceed.

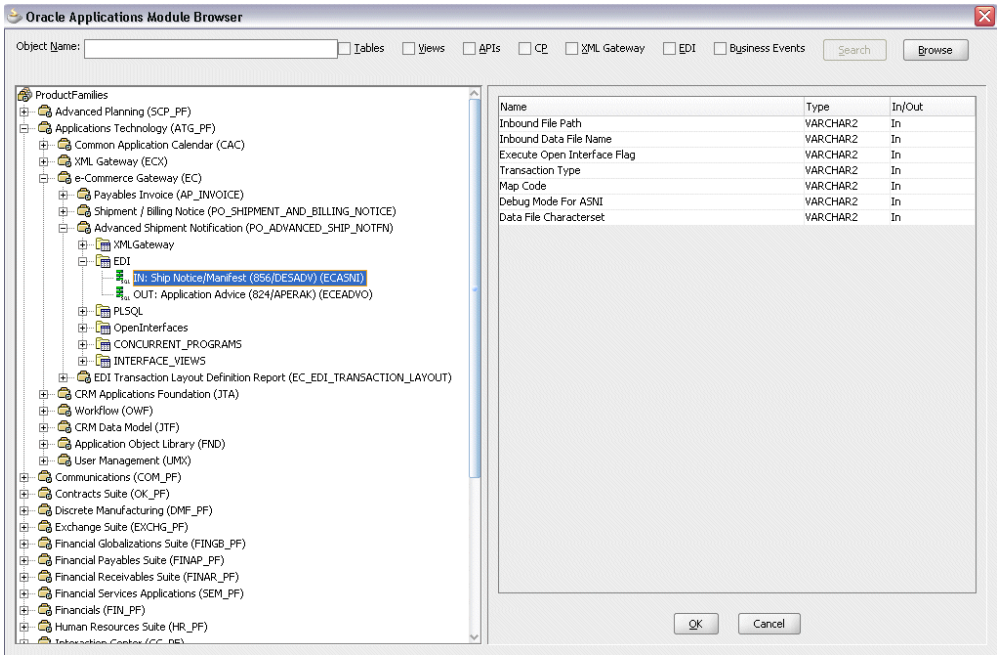
Click **Get Object** in the Application Interface dialog to open the Oracle Applications Module Browser.

5. Once you have identified an existing database connection or completed a new connection for the service, you can select an e-Commerce Gateway interface through Oracle Applications Module Browser.

Note: 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.

Specify an Interface from The Oracle Applications Module Browser



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

For example, navigate to *Product Families* > *Applications Technology* > *e-Commerce Gateway* > *Advanced Shipment Notification* > *EDI* to select `EDI > IN: Ship Notice/Manifest (856/DESADV) (ECASNI)` EDI concurrent program.

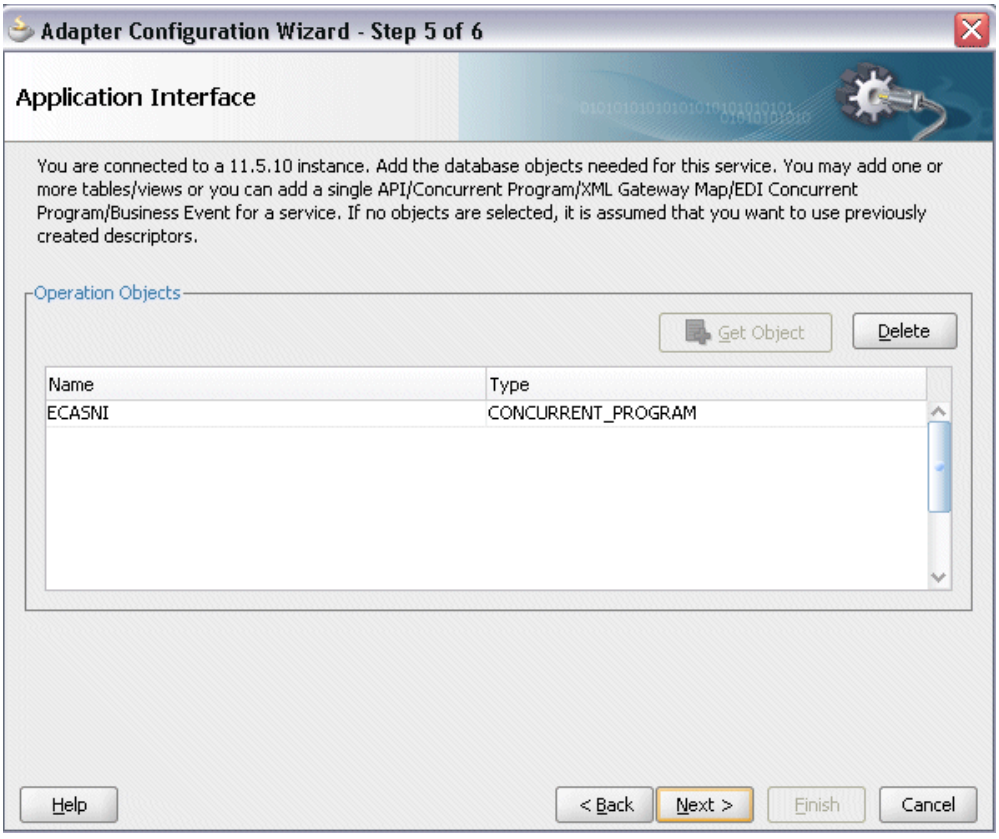
Note: You can also search for an EDI program by entering the name

of the program in the **Object Name** field. Select the **EDI** check box and click **Search**.

6. Click **OK**.

   The selected EDI concurrent program is added to Operation Objects.

Adapter Configuration Wizard - Application Interface Page



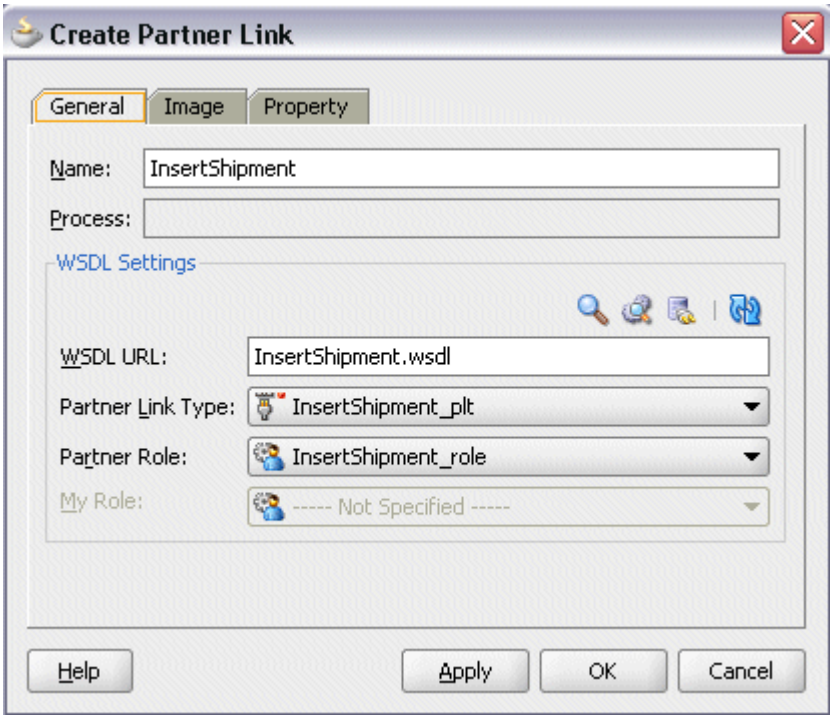
   Click **Next**.

7. Click **Finish** to complete the process of configuring Adapter for Oracle Applications.

   The wizard generates the WSDL file corresponding to the selected interface. This WSDL file is now available for the partner link.

   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 same wrapper.

Completing the Partner Link Configuration



8. Click **Apply** and then **OK**. The partner link is created with the required WSDL settings.

Adding a Partner Link for File Adapter

Use this step to configure a BPEL process by synchronously reading an existing order to obtain the shipping details.

To add a Partner Link for File Adapter to read shipping details:

1. In JDeveloper BPEL Designer, click **BPEL Services** in the Component palette.

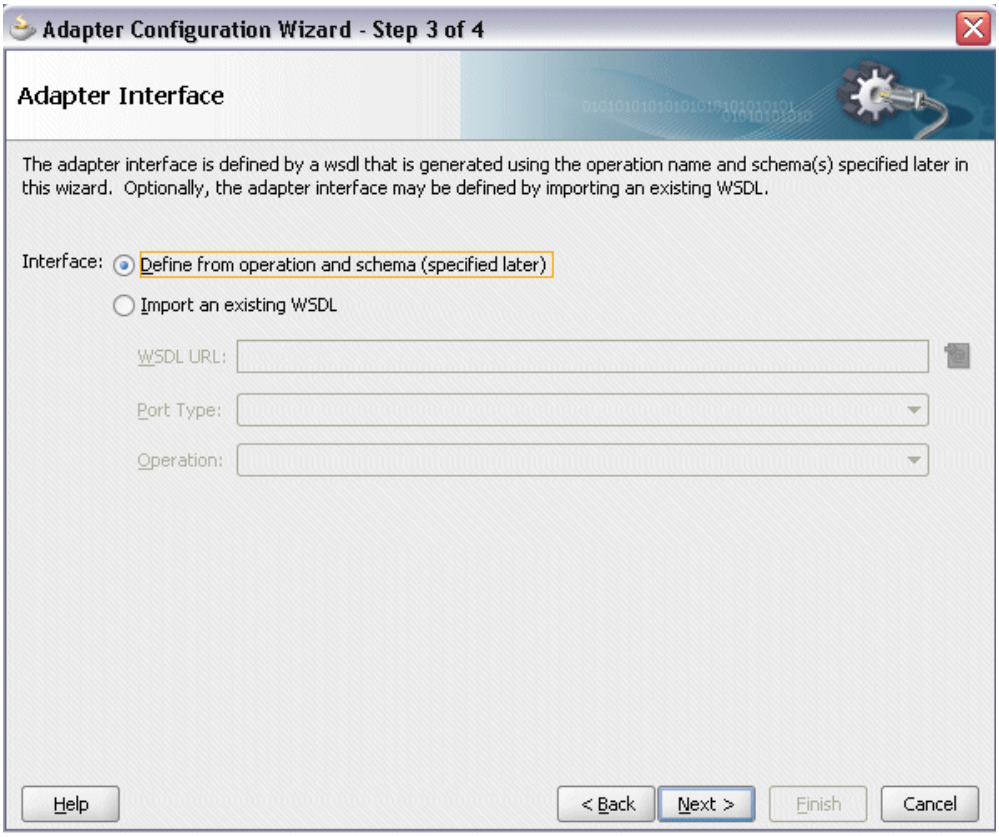
   Drag and drop **File Adapter** from the BPEL Services list into the right Partner Link swim lane of the process diagram before the partner link `InsertShipment`. The Adapter Configuration Wizard Welcome page appears.

   Click **Next**.

2. In the Service Name dialog, enter a name for the file adapter service, such as `getShippingDetails`.

3. Click **Next**. The Adapter Interface dialog appears.

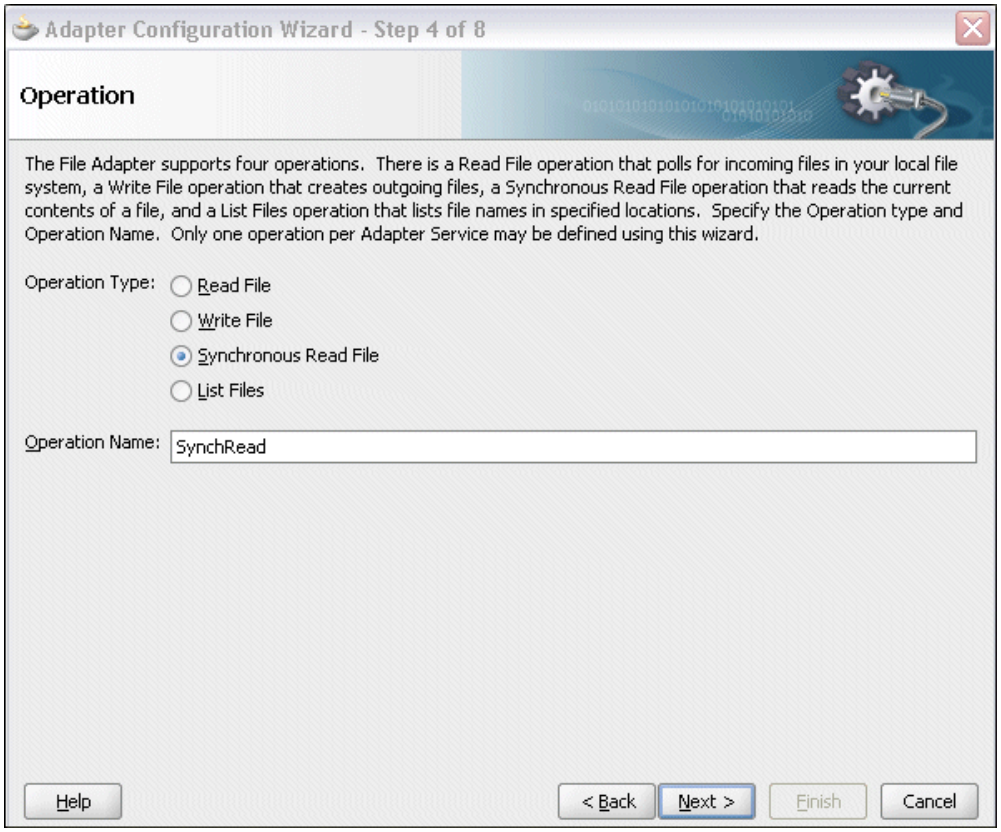
Specifying the Adapter Interface



Select the **Define from operation and schema (specified later)** radio button and click **Next**.

4. In the Operation dialog, specify the operation type. For example, select the **Synchronous Read File** radio button. This automatically populates the **Operation Name** field.

Specifying the Operation

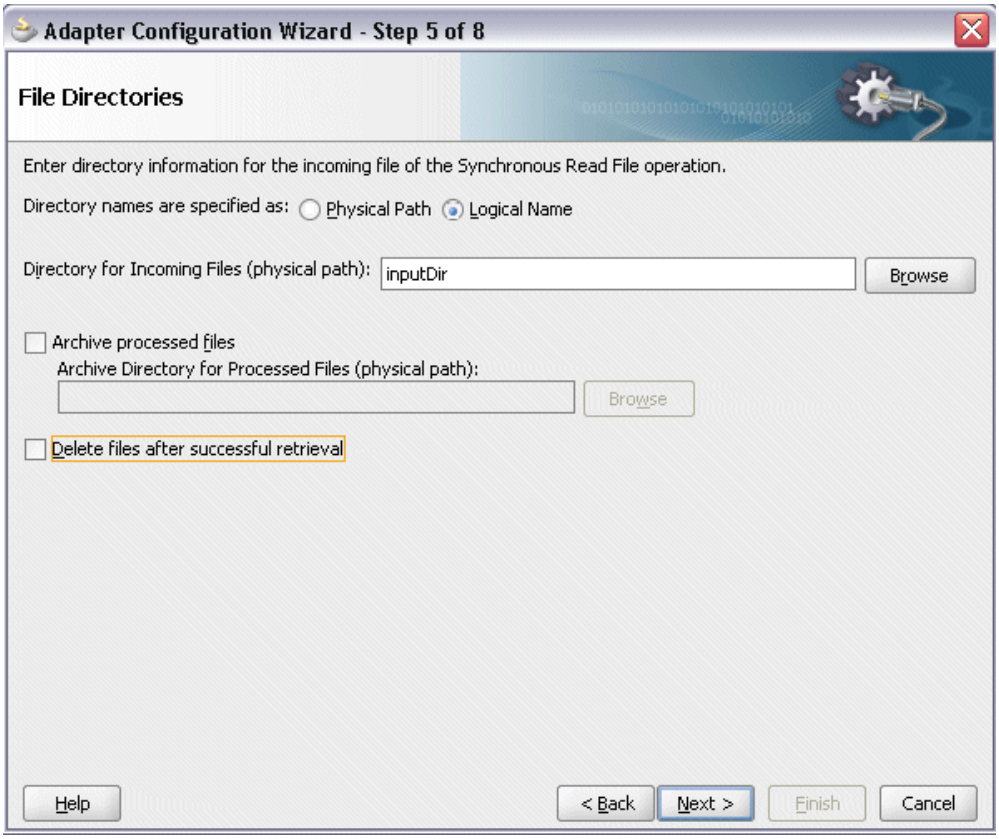


Click **Next** to access the File Directories dialog.

5. Select the **Logical Name** radio button and specify directory for incoming files, such as `inputDir`.

   Ensure the **Delete files after successful retrieval** check box is not selected.

Configuring the Input File



Click **Next** to open the File Name dialog.

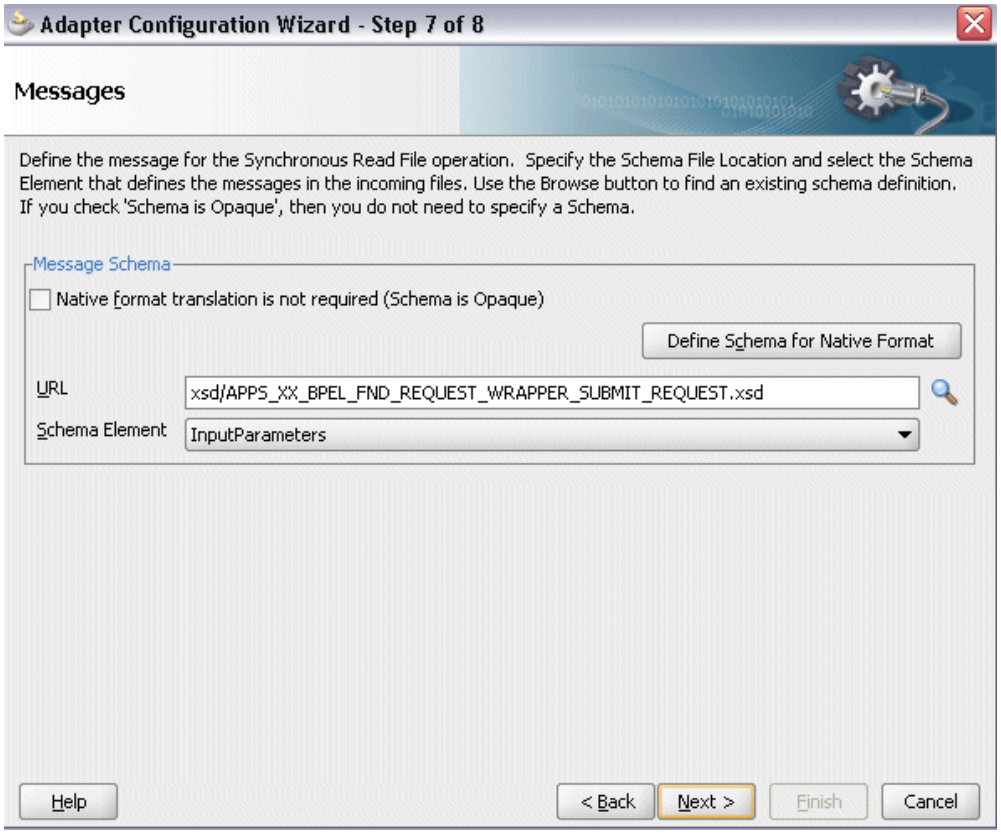
6. Enter the name of the file for the synchronous read file operation. For example, enter `order_data.xml`. Click **Next** to open the Messages dialog.

7. Select the 'browse for schema file' icon to open the Type Chooser.

   Click Type Explorer and select *Project Schema Files* > *APPS_XX_BPEL_FND_REQUEST_WRAPPER_SUBMIT_REQUEST.xsd* > *InputParameters*.

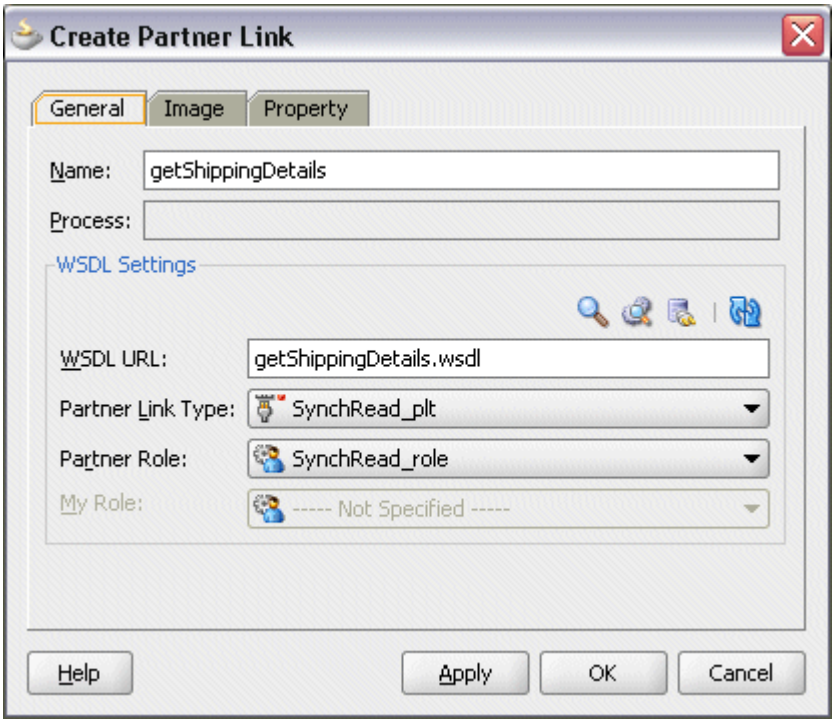
   The selected schema information will be automatically populated in the URL and Schema Element fields.

Specifying Message Schema



8. Click **Next** and then **Finish**. The wizard generates the WSDL file corresponding to the partner link. The main Create Partner Link dialog box appears, specifying the new WSDL file `getShippingDetails.wsdl`.

Completing the Partner Link Configuration



Click **Apply** and **OK** to complete the configuration and create the partner link with the required WSDL settings for the File Adapter service.

The `getShippingDetails` Partner Link appears in the BPEL process diagram.

Configuring the Invoke Activities

After adding and configuring the partner link, you can configure the following two **Invoke** process activities to invoke the EDI concurrent program and File Adapter partner link:

1. To get the shipping details that is received from the Receive activity by invoking the `getShippingDetails` partner link in an XML file.

2. To create a shipment notice by invoking `InsertShipment` partner link.

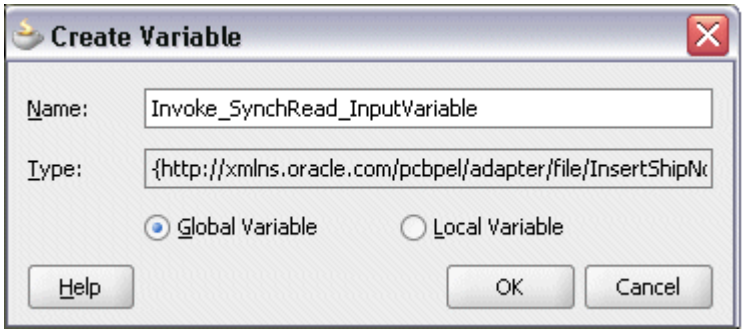
To add the first Invoke activity for a partner link to get shipping details:

1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the first **Invoke** activity into the center swim lane of the process diagram, between the **receiveInput** and **callbackClient** activities.

2. Link the Invoke activity to the `getShippingDetails` service. The Edit Invoke dialog appears.

3. Enter a name for the Invoke activity, then click the **Create** icon next to the **Input Variable** field to create a new variable. The Create Variable dialog box appears.

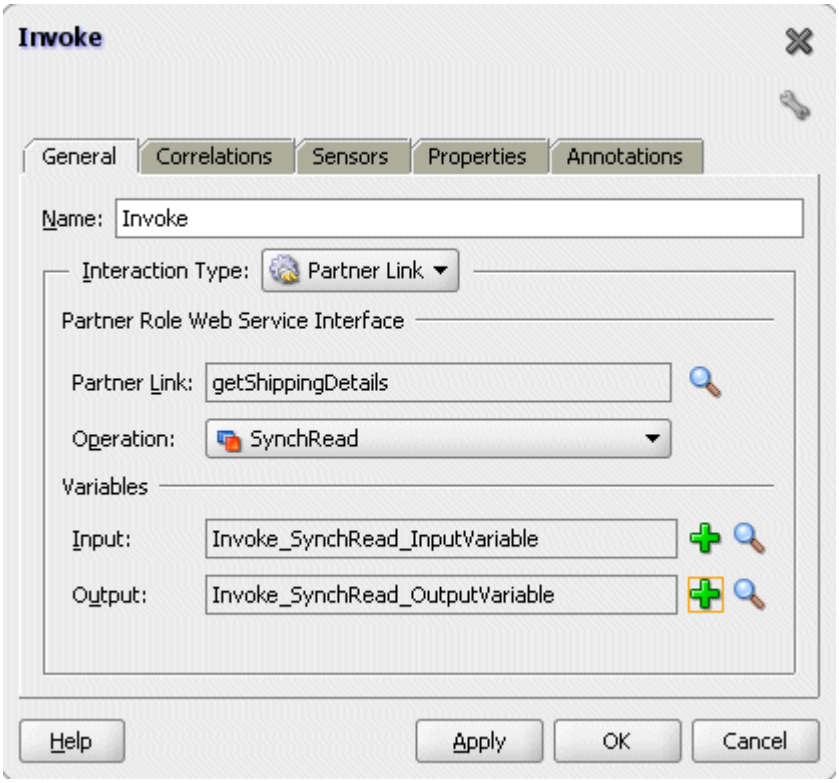
Create Input Variable



4. Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK**.

5. Click the **Create** icon next to the **Output Variable** field to create a new variable. The Create Variable dialog box appears.

6. Select **Global Variable**, then enter a name for the variable. You can also accept the default name. Click **OK**.



7. Click **Apply** and **OK** in the Edit Invoke dialog box to finish configuring the Invoke activity.

   The Invoke activity appears in the process diagram.

To add the second Invoke activity for a partner link to create a shipment notice:

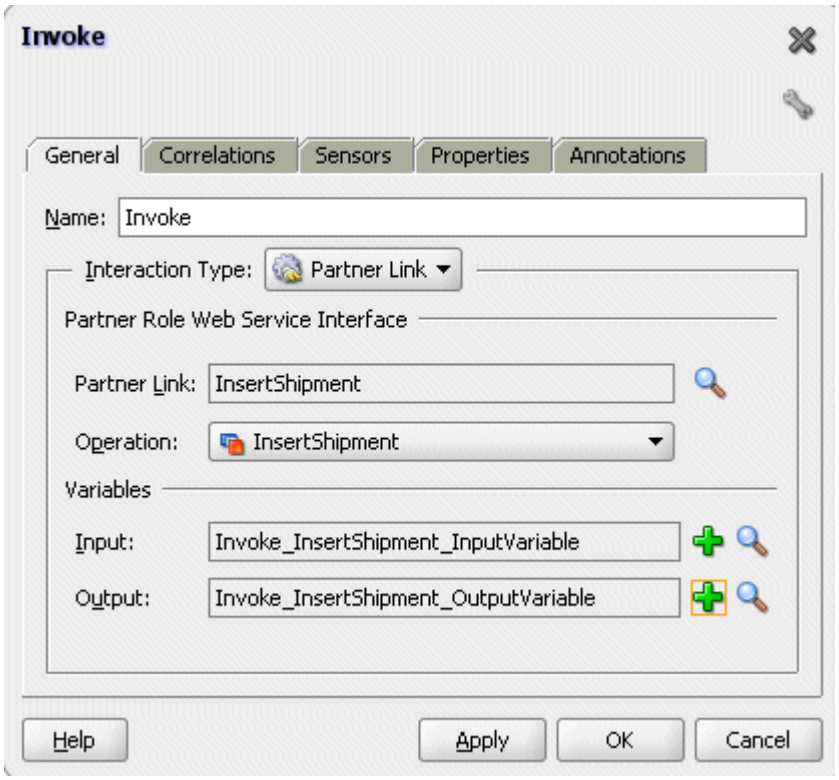
1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the third **Invoke** activity into the center swim lane of the process diagram, between the first **Invoke** and **callbackClient** activities.

2. Link the Invoke activity to the `InsertShipment` service. The Edit Invoke dialog box appears.

   The **Operation** is automatically selected, depending on the EDI concurrent program that you chose when configuring the partner link.

3. Repeat Step 3 to Step 6 described in the first Invoke activity procedure.

   Click **Apply**.

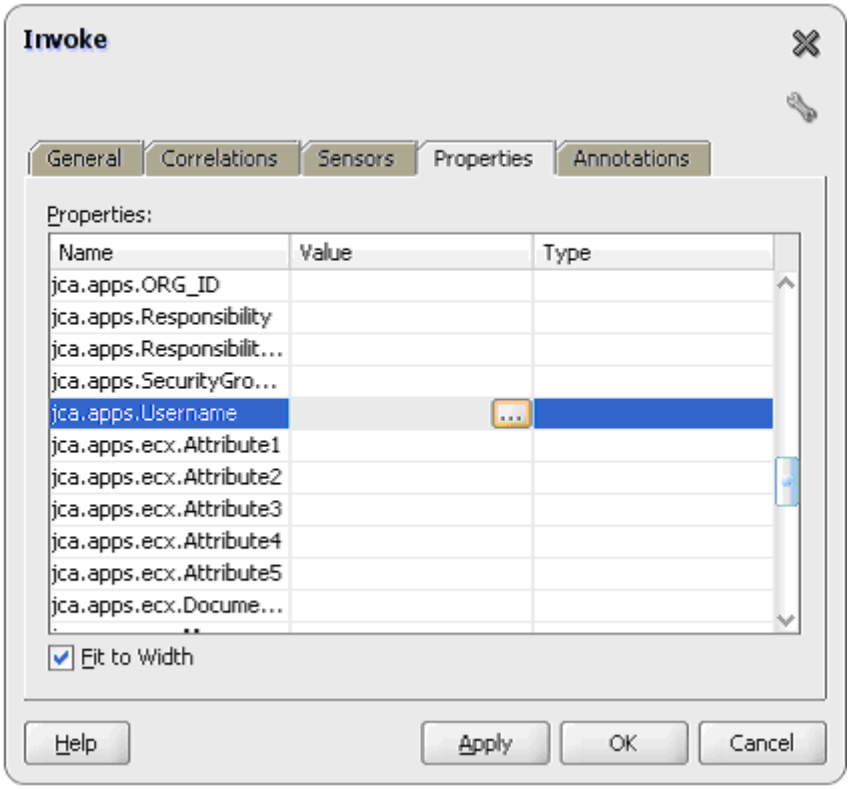


4. **Setting Header Properties for Application Context**

   Use the following steps to set the header message properties required to pass application context required to complete the BPEL process:

   1. Select the Properties tab to set the Header message properties required to pass application context required to complete the BPEL process.

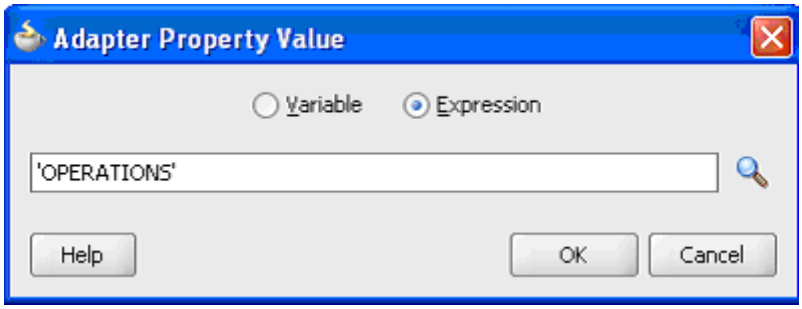
Setting Header Message Properties



2. Scroll down to locate the `jca.apps.Username` property from the list and double click the associated value field to enable the **Adapter Property Value** icon.

3. Click the icon to open the Adapter Property Value dialog for the selected `jca.apps.Username` property.

Entering the Header Message Property Value



4. Select the **Expression** radio button and enter `'OPERATIONS'` as the property

value.

Click **OK**.

5. Repeat Step 2 to Step 4 to assign `'Purchasing, Vision Operations (USA)'` for `jca.apps.Responsibility`.

5. Click **Apply** and then **OK** to complete the Invoke activity.

Configuring the Assign Activity

Use the Assign activity to pass the output of `getShippingDetails` service as an input to the `InsertShipment` service.

To add an Assign activity:

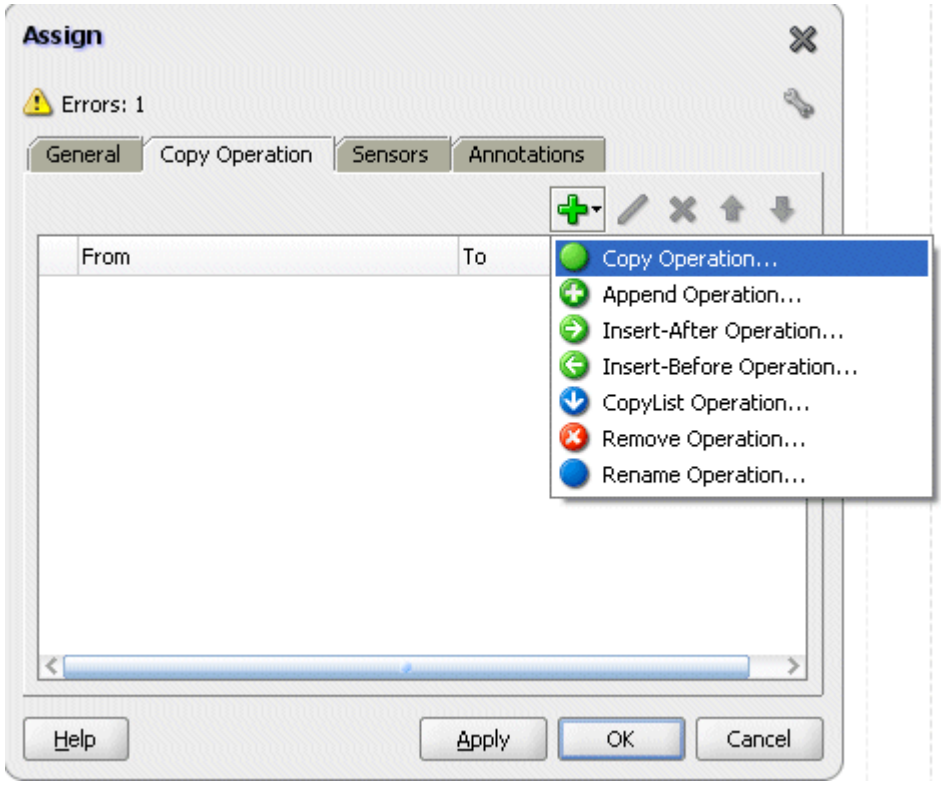
1. In JDeveloper BPEL Designer, select BPEL Activities and Components in the component palette. Drag and drop the **Assign** activity into the center swim lane of the process diagram between the two **Invoke** activities that you just created earlier.

2. Double-click the **Assign** activity to access the Edit Assign dialog.

   Click the General tab to enter a name for the Assign activity. For example, `setShipInfo`.

3. Select the Copy Operation tab, click the 'Plus' sign icon and select **Copy Operation** from the menu. The Create Copy Operation window appears.

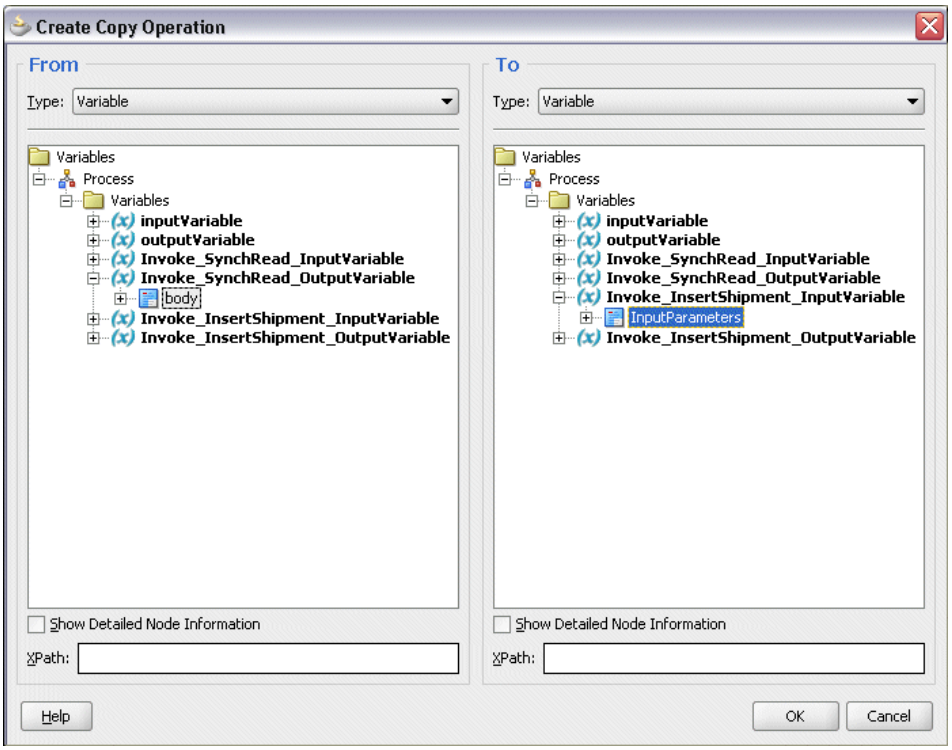
Specifying a Copy Operation Action



4. Enter the parameter information:

   - In the From navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_SynchRead_OutputVariable** and select **body**.

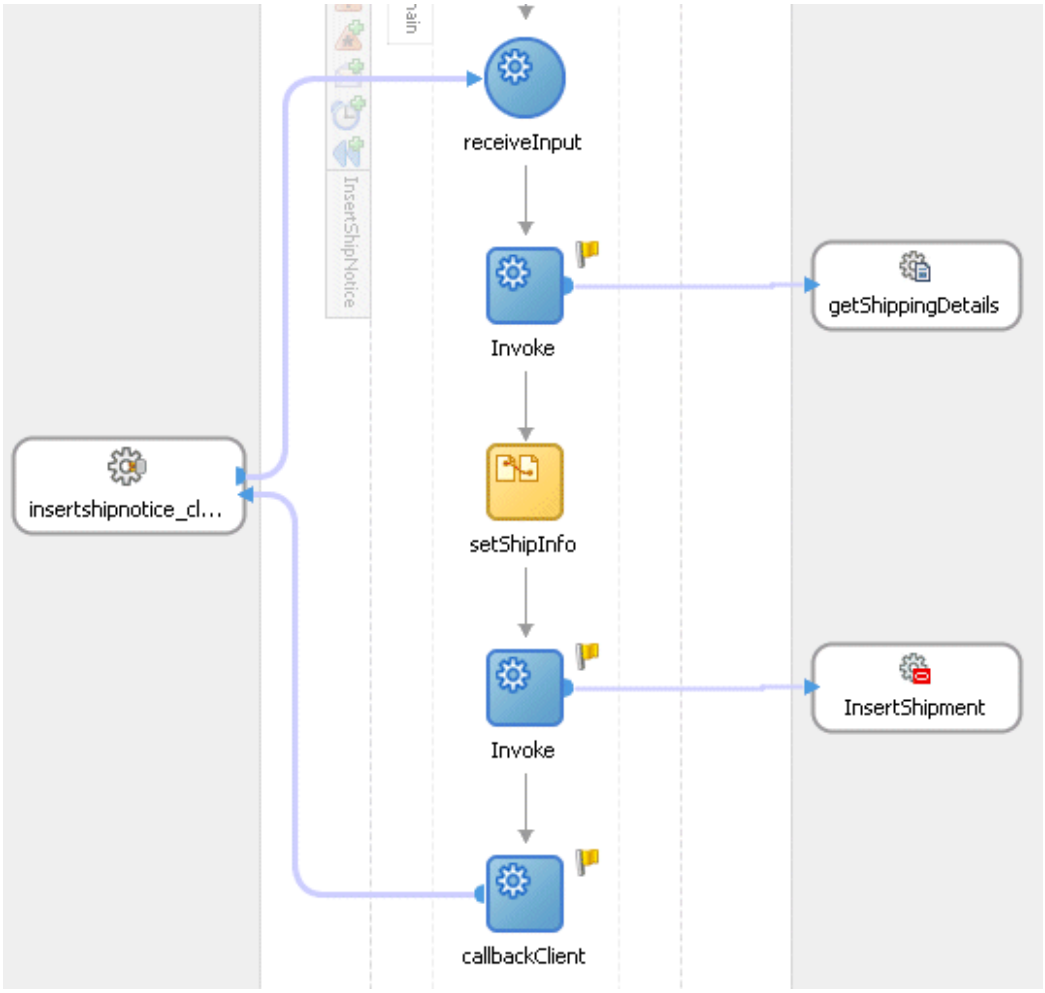
   - In the To navigation tree, select type Variable, then navigate to **Variable > Process > Variables > Invoke_InsertShipment_InputVariable** and select **Input Parameters**.



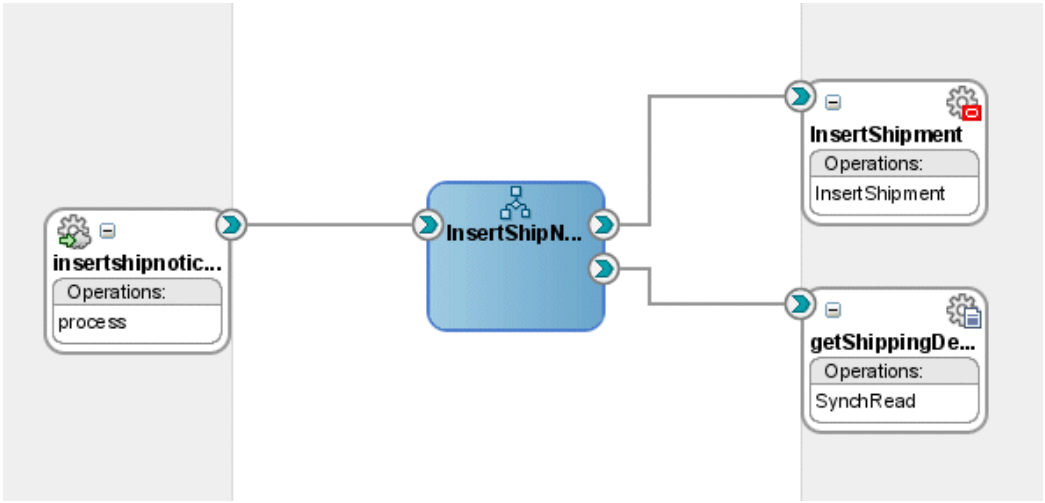
- Click **OK**. The Edit Assign dialog box appears.

5. Click **Apply** and **OK** to complete the configuration of the Assign activity.

Completed Concurrent Program BPEL Process Project



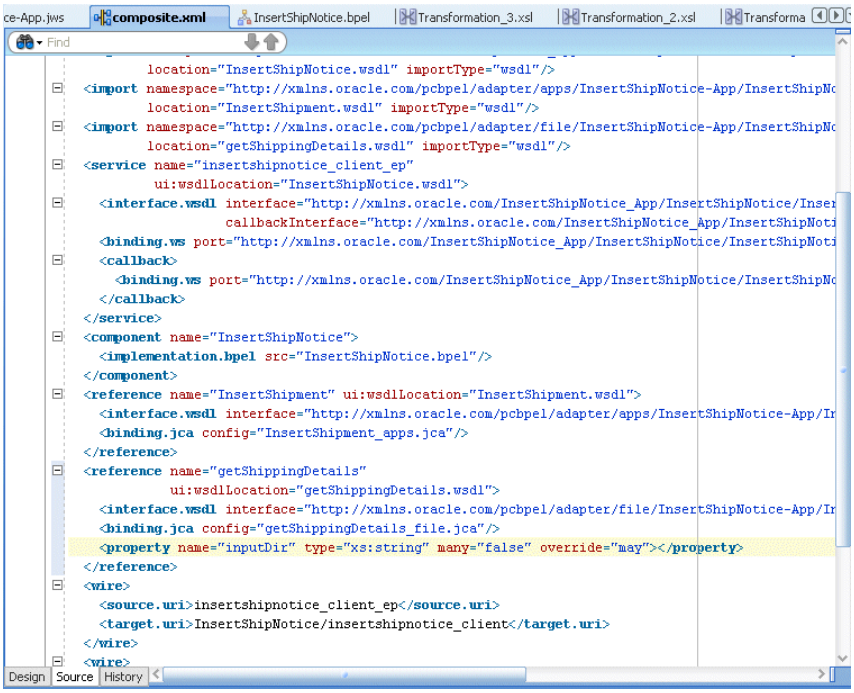
Click the `composite.xml` to display the Oracle JDeveloper composite diagram:



Note: Click the Source tab of `composite.xml` to enter a value for the physical directory `inputDir` for the reference `getShippingDetails` (such as `/usr/tmp`).

```
<property name="inputDir" type="xs:string"
many="false" override="may">/usr/tmp</property>
```

Specifying the Physical Directory for the Property



Run-Time Tasks for e-Commerce Gateway

After designing the BPEL process, the next step is to deploy, run and monitor it.

1. Deploy the BPEL process., page 9-30
2. Test the BPEL process., page 9-32
3. Verify records in Oracle Applications., page 9-34

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 application server (Oracle WebLogic Server) that you have established the connection.

Prerequisites

- You must have established the connectivity between the deign-time environment and an application server.

  For more information, see Configuring the Data Source in Oracle WebLogic Server,

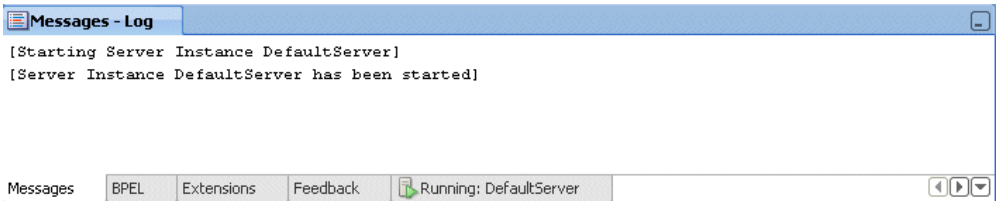
page A-3 and Creating an Application Server Connection, page A-8.

- Oracle WebLogic Server has been started.

  Before deploying the BPEL process, you need to start the Oracle WebLogic Server that you have established the connection.

  If a local instance of the WebLogic Server is used, start the WebLogic Server by selecting **Run > Start Server Instance** from Oracle JDeveloper.

  Once the WebLogic Admin Server "DefaultServer" instance is successfully started, you should find that <Server started in Running mode> and DefaultServer started message in the Running:DefaultServer and Messages logs.

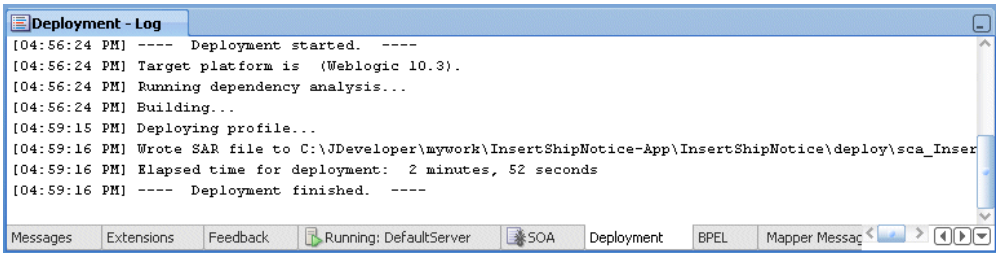


To deploy the BPEL process:

1. Select the BPEL project in the Applications Navigator.

2. Right-click the project name, and then select **Deploy > [project name] > [serverConnection]** from the menu that appears.

3. The BPEL process is compiled and deployed. You can check the progress in the Messages window.



Testing the BPEL Process

Once the BPEL process is deployed, you can manage and monitor the process from the Oracle Enterprise Manager Fusion Middleware Control Console. You can also test the process and the integration interface by manually initiating the process.

To test the BPEL process:

1. Navigate to Oracle Enterprise Manager Fusion Middleware Control Console (`http://<servername>:<portnumber>/em`). The composite you deployed is displayed in the Applications Navigation tree.

2. Enter username (such as `weblogic`) and password and click **Login** to log in to a farm.

   You may need to select an appropriate target instance farm if there are multiple target Oracle Enterprise Manager Fusion Middleware Control Console farms.

3. From the Farm base domain, expand the **SOA** >**soa-infra** to navigate through the SOA Infrastructure home page and menu to access your deployed SOA composite applications running in the SOA Infrastructure for that managed server.

   > **Note:** The Farm menu always displays at the top of the navigator. As you expand the SOA folder in the navigator and click the links displayed beneath it, the SOA Infrastructure menu becomes available at the top of the page.

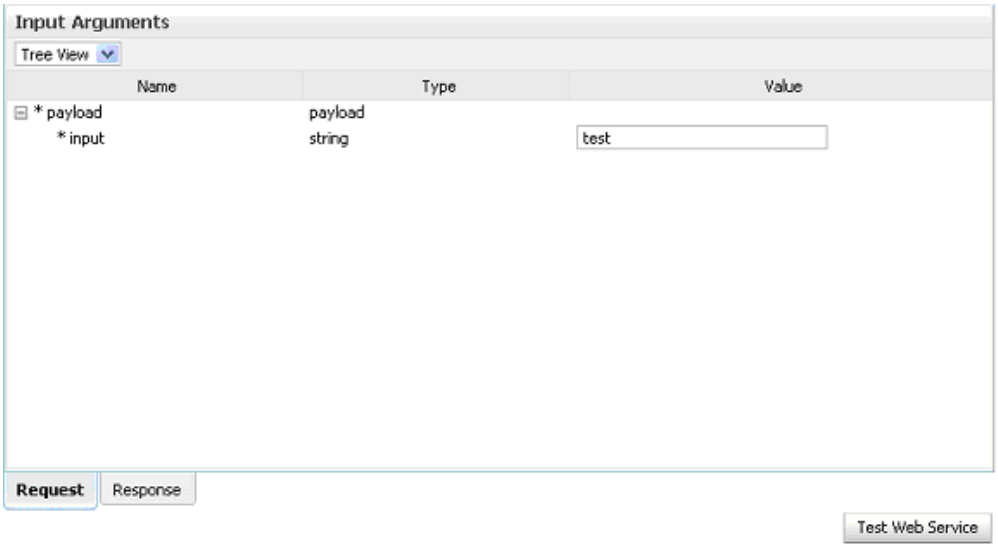
   Click the SOA composite application that you want to initiate (such as 'InsertShipment') from the SOA Infrastructure.

   Click **Test** at the top of the page.

4. The Test Web Service page for initiating an instance appears. You can specify the XML payload data to use in the Input Arguments section.

   Enter the input string required by the process and click **Test Web Service** to initiate the process.

Testing Web Service



The test results appear in the Response tab upon completion.

5. Click the Instances tab. The SOA composite application instance ID, name, conversation ID, most recent known state of each instance since the last data refresh of the page are displayed.

   In the Instance ID column, click a specific instance ID to show the message flow through the various service components and binding components. The Flow Trace page is displayed.

   In the Trace section, you should find the sequence of the message flow for the service binding component (`insertshipnotice_client_ep`), BPEL component (`InsertShipment`), and reference binding components (`getShippingDetails` and `InsertShipment`). All involved components have successfully received and processed messages.

   If any error occurred during the test, you should find it in the Faults section.

6. Click your BPEL service component instance link (such as `InsertShipment`) to display the Instances page where you can view execution details for the BPEL activities in the Audit Trail tab.

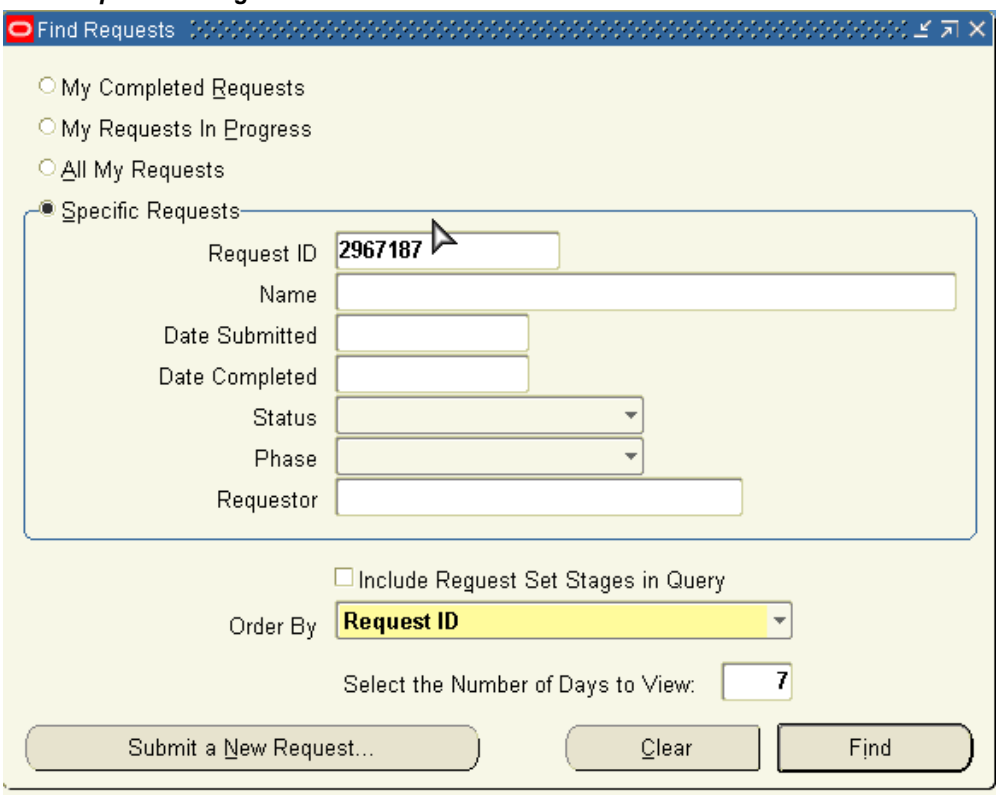
   Click the Flow tab to check the BPEL process flow diagram. Click an activity of the process diagram to view the activity details and flow of the payload through the process.

Verifying Records in Oracle Applications

To verify records in Oracle Applications:

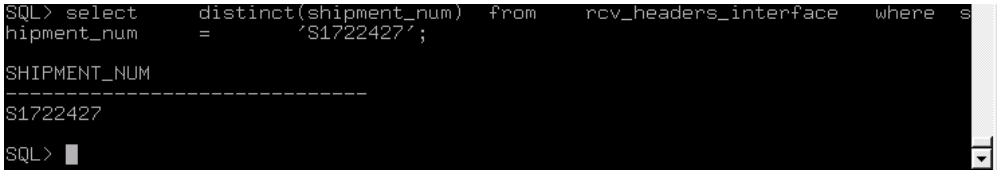
1. Log in to Oracle Applications as the System Administrator.
2. Select **Requests** from the **View** menu.
3. Search for the Request by entering the Request Id that you got from the audit trail, then click **Find.**

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`, you can also query the appropriate table in Oracle Applications to search for the relevant records that have been inserted.

Querying Oracle Applications for a Record



A

WSDL Definition File and Connection Information Details

This appendix covers the following topics:

- WSDL Definition File
- Configuring Connection Information

WSDL Definition File

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.

Example of a WSDL File

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="InsertShipNotice"
             targetNamespace="http://xmlns.oracle.com/InsertShipNotice_App/InsertShipNotice/InsertShipNotice"
             xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
             xmlns:client="http://xmlns.oracle.com/InsertShipNotice_App/InsertShipNotice/InsertShipNotice"
             xmlns:plnk="http://schemas.xmlsoap.org/ws/2003/05/partner-link/">

        <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        TYPE DEFINITION - List of services participating in this BPEL process
        The default output of the BPEL designer uses strings as input and
        output to the BPEL Process. But you can define or import any XML
        Schema type and use them as part of the message types.
        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
        <wsdl:types>
                <schema xmlns="http://www.w3.org/2001/XMLSchema">
                        <import namespace="http://xmlns.oracle.com/InsertShipNotice_App/InsertShipNotice/InsertShipNotice" schemaLocation="xsd/InsertShipNotice.xsd" />
                </schema>
        </wsdl:types>

        <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        MESSAGE TYPE DEFINITION - Definition of the message types used as
        part of the port type defintions
        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
        <wsdl:message name="InsertShipNoticeRequestMessage">
                <wsdl:part name="payload" element="client:process"/>
        </wsdl:message>

        <wsdl:message name="InsertShipNoticeResponseMessage">
                <wsdl:part name="payload" element="client:processResponse"/>
        </wsdl:message>


        <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        PORT TYPE DEFINITION - A port type groups a set of operations into
        a logical service unit.
        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
        <!-- portType implemented by the InsertShipNotice BPEL process -->
        <wsdl:portType name="InsertShipNotice">
                <wsdl:operation name="process">
                        <wsdl:input message="client:InsertShipNoticeRequestMessage"/>
                </wsdl:operation>
        </wsdl:portType>

        <!-- portType implemented by the requester of InsertShipNotice BPEL process
        for asynchronous callback purposes
        -->
        <wsdl:portType name="InsertShipNoticeCallback">
                <wsdl:operation name="processResponse">
                        <wsdl:input message="client:InsertShipNoticeResponseMessage"/>
                </wsdl:operation>
        </wsdl:portType>

        <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        PARTNER LINK TYPE DEFINITION
        the InsertShipNotice partnerLinkType binds the provider and
        requester portType into an asynchronous conversation.
        ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
        <plnk:partnerLinkType name="InsertShipNotice">
                <plnk:role name="InsertShipNoticeProvider">
                        <plnk:portType name="client:InsertShipNotice"/>
                </plnk:role>
                <plnk:role name="InsertShipNoticeRequester">
                        <plnk:portType name="client:InsertShipNoticeCallback"/>
                </plnk:role>
        </plnk:partnerLinkType>
</wsdl:definitions>
```

Configuring Connection Information

Oracle Adapter for Oracle Applications is deployed as J2CA 1.5 resource adapter within the Oracle WebLogic Server container.

Although Oracle Adapter for Oracle Applications is physically deployed as J2CA 1.5 resource adapter, the logical deployment of the Adapter involves creating the connection entries for the J2CA 1.0 resource adapter. This connection information is for tying up the database connection information provided when you create a partner link or service and is used by iAS at run time in the background.

The following example of connection configuration for Adapter for Oracle Applications can be used with the Oracle Applications adapter tutorials packaged with the product. For the logical deployment changes to take effect, the Oracle WebLogic Server container process must be restarted.

The following topics are included in this section:

- Configuring the Data Source in Oracle WebLogic Server, page A-3
- Creating the Application Server Connection, page A-8
- Creating the Oracle Applications Adapter Database Connection, page A-12

Configuring the Data Source in Oracle WebLogic Server

Oracle WebLogic Server consists of a J2CA container for hosting J2CA resource adapters. J2CA defines standard Java interfaces for simplifying the integration of a J2EE server with various back-end applications. All client applications run within the Oracle WebLogic Server environment.

The J2CA resource adapter is a WebLogic Server component contained in a Resource Adapter Archive (RAR) file within the applications/directory. An RAR file contains a correctly formatted deployment descriptor. In addition, it contains declarative information about the contract between the Oracle WebLogic Server and resource adapter.

The `weblogic-ra.xml` file is the deployment descriptor for a WebLogic resource adapter. It contains deployment configurations for deploying resource adapters to Oracle WebLogic Server, which includes the back-end application connection information as specified in the deployment descriptor of the resource adapter, Java Naming and Directory Interface (JNDI) name to be used, connection pooling parameters, security identities, Work Manager properties, logging, and resource principal mapping mechanism and configurations.

> **Note:** A WebLogic resource adapter also uses `ra.xml` descriptor that describes the resource adapter-related attributes type and its deployment properties using the standard XML schema specified by the J2CA 1.5 Specification.

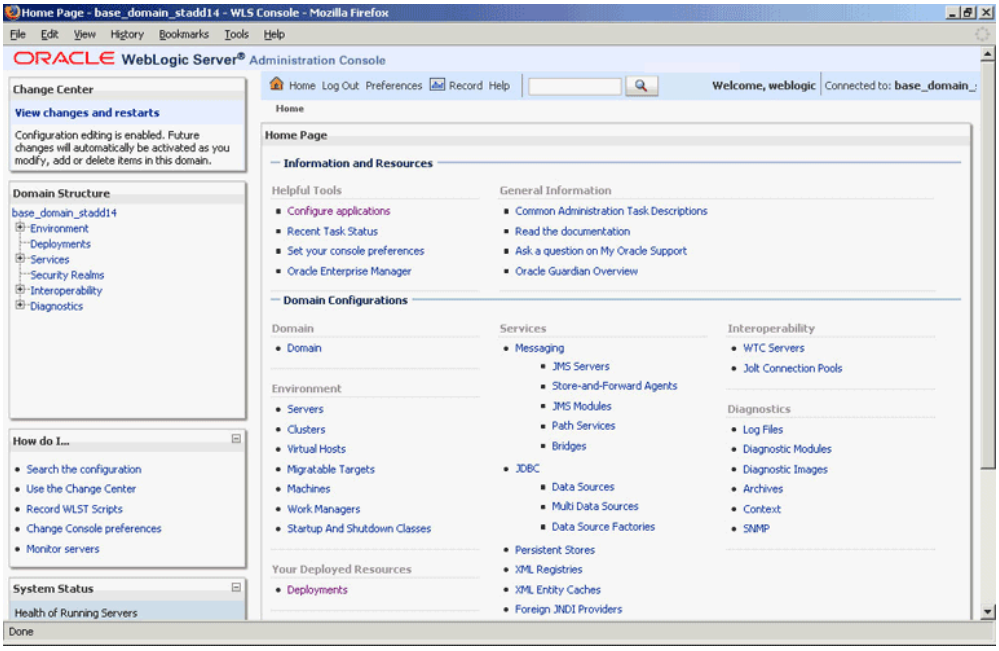
After successfully installing Oracle WebLogic Server, use the following steps to configure the data sources in the Oracle WebLogic Server Administration Console:

> **Note:** How to install Oracle WebLogic Server and create connection factory, see *Oracle WebLogic Server Installation Guide* for details.

1. Navigate to `http://servername:portnumber/console`.
2. Enter the server administrator's username and password to log on to the Oracle WebLogic Server Administration Console.

   The Home page of the Oracle WebLogic Server Administration Console appears.

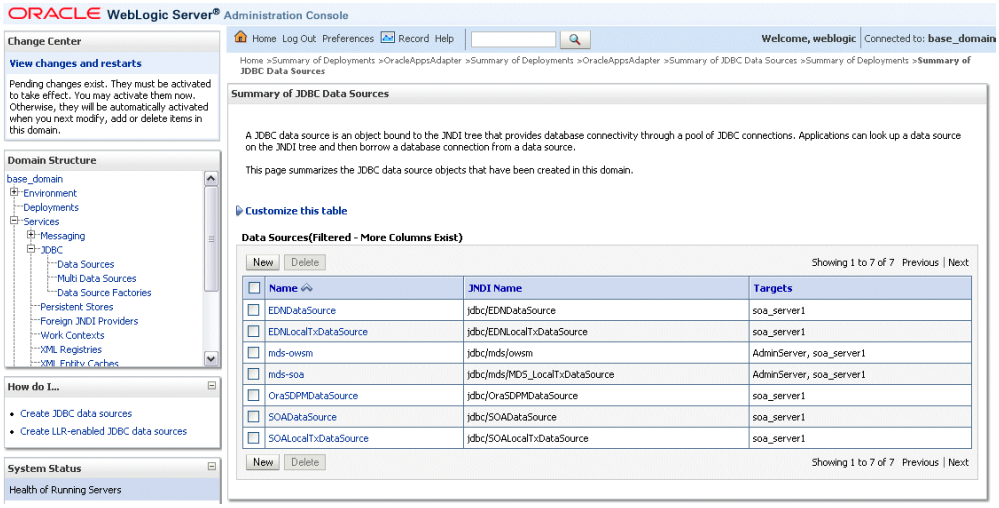
Oracle WebLogic Server Administration Console Home Page



3. Under the Domain Structure section, select **Services > JBDC > DataSources**.

   The Summary of JDBC Data Sources page is displayed.

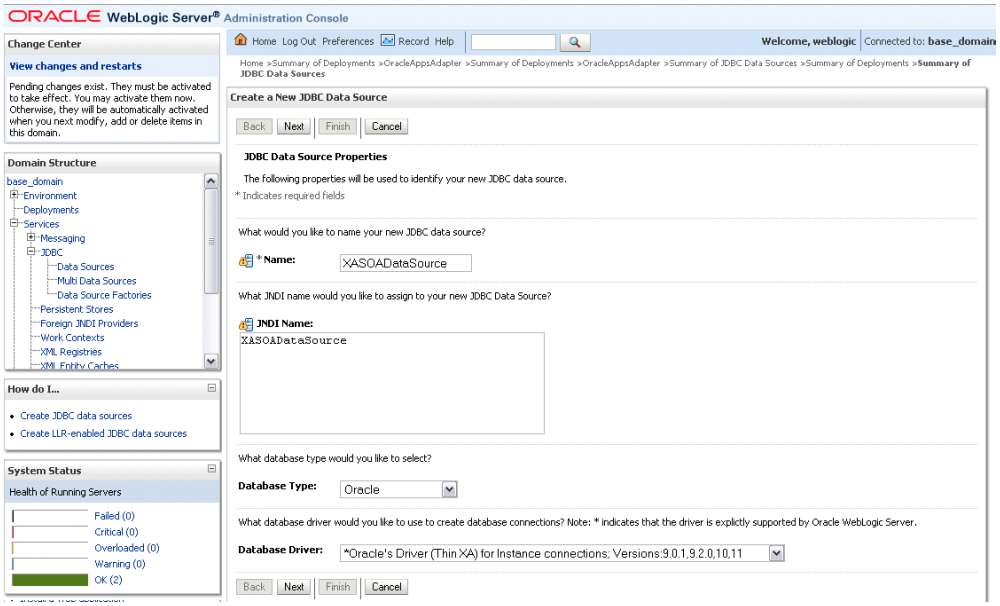
The Summary of JDBC Data Sources Page



4. In the Summary of JDBC Data Sources section, click **New** to create a new data source. The Create a New JDBC Data Source page is displayed.

5. Enter the values for the properties to be used to identify your new JDBC data source.

The Create a New JDBC Data Source - JDBC Data Source Properties Page



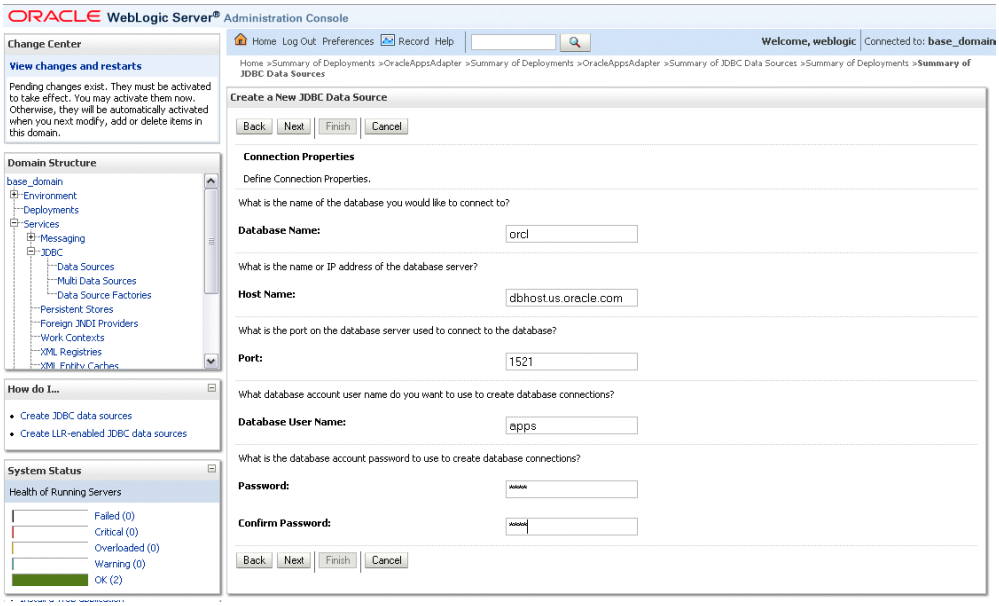
- Name: Enter a desired new JDBC data source name.
- JNDI Name: Enter a JNDI name that you like to assign to your new JDBC Data Source.
- Database Type: Select `Oracle` as database type from the drop-down list.
- Database Driver: Select `Oracle's Driver` (Oracle's Driver (Thin and ThinXA) is the only supported version).

  Note: A driver name with '*' indicates that the driver is explicitly supported by Oracle WebLogic Server.

6. Click **Next**. The Create a New JDBC Data Source - Transaction Options page is displayed.

7. Click **Next**. The Create a New JDBC Data Source - Connection Properties page is displayed.

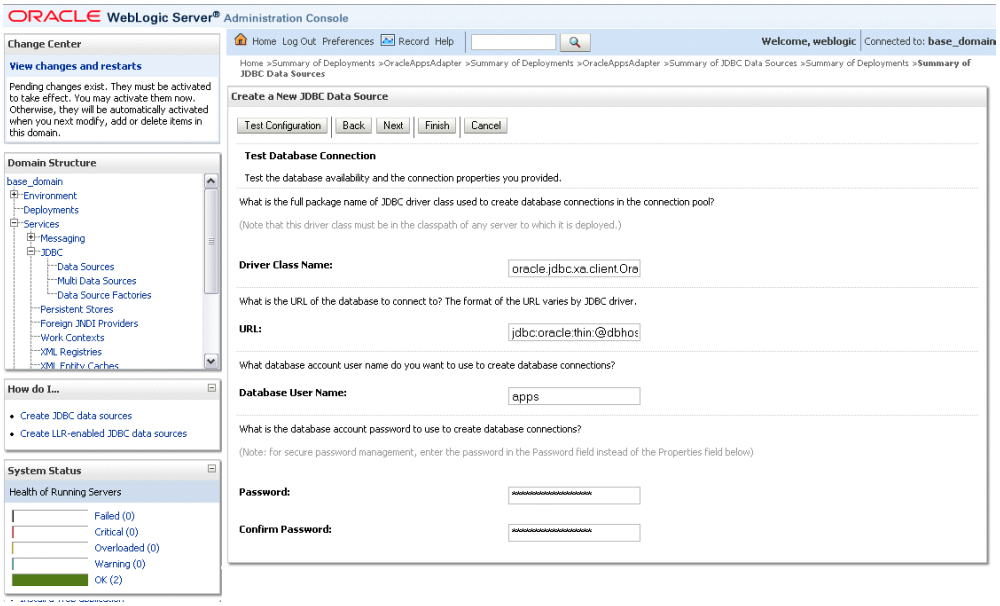
The Create a New JDBC Data Source - Connection Properties page



8. Enter the connection properties in the Connection Properties page.
   - Database Name: Enter a database name that you would like to connect.
   - Host Name: Enter a host name or IP address of the database server.
   - Port: Enter a database port on the database server used to connect to the database.
   - Database User Name: Enter the database account username that you want to use to create the database connections.
   - Password: Enter the database account password that you want to use to create the database connections.
   - Confirm Password: Enter the same password for confirmation.

9. Click **Next**. The Create a New JDBC Data Source - Test Database Connection page is displayed.

The Create a New JDBC Data Source - Test Database Connection Page

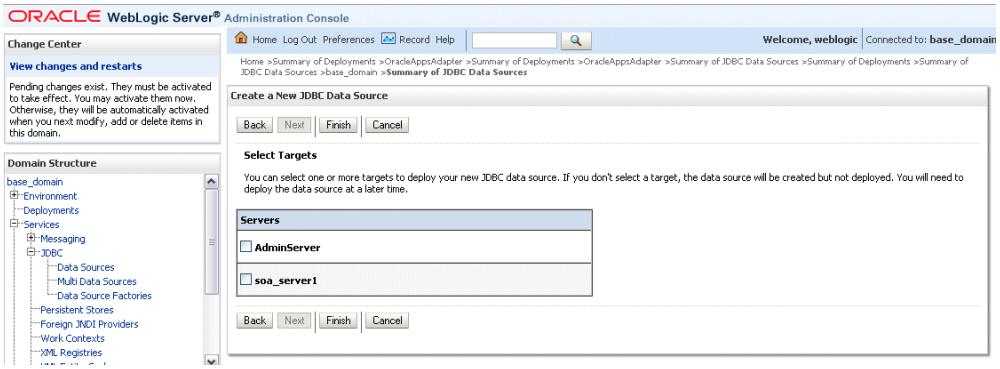


10. Click **Test Configuration** to test the database availability and the connection properties you provided.

    A 'Connection test succeeded' message appears at the top of the Create a New JDBC Data Source - Test Database Connection page.

11. Click **Next**. The Create a New JDBC Data Source - Select Targets page is displayed.

The Create a New JDBC Data Source - Select Targets Page



12. Select the 'AdminServer' check box, and then click **Finish**.

    The Summary of JDBC Data Sources page is displayed. This page summarizes the JDBC data source objects that have been created in this domain. The Data Source that you created appears in this list.

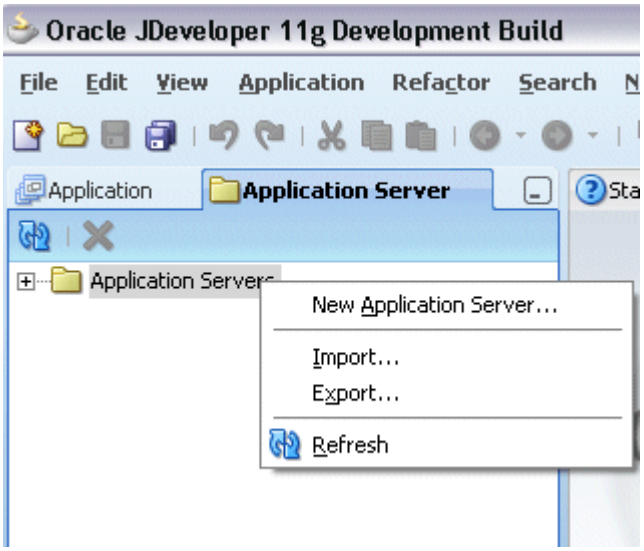
Creating the Application Server Connection

You must establish a connectivity between the design-time environment and the server you want to deploy it to. In order to establish such a connectivity, you must create the application server connection.

Use the following steps to create the application server connection:

1. From Oracle JDeveloper, select **View** >**Application Server Navigator** to open the Application Server tab.

Create a New Application Server Connection

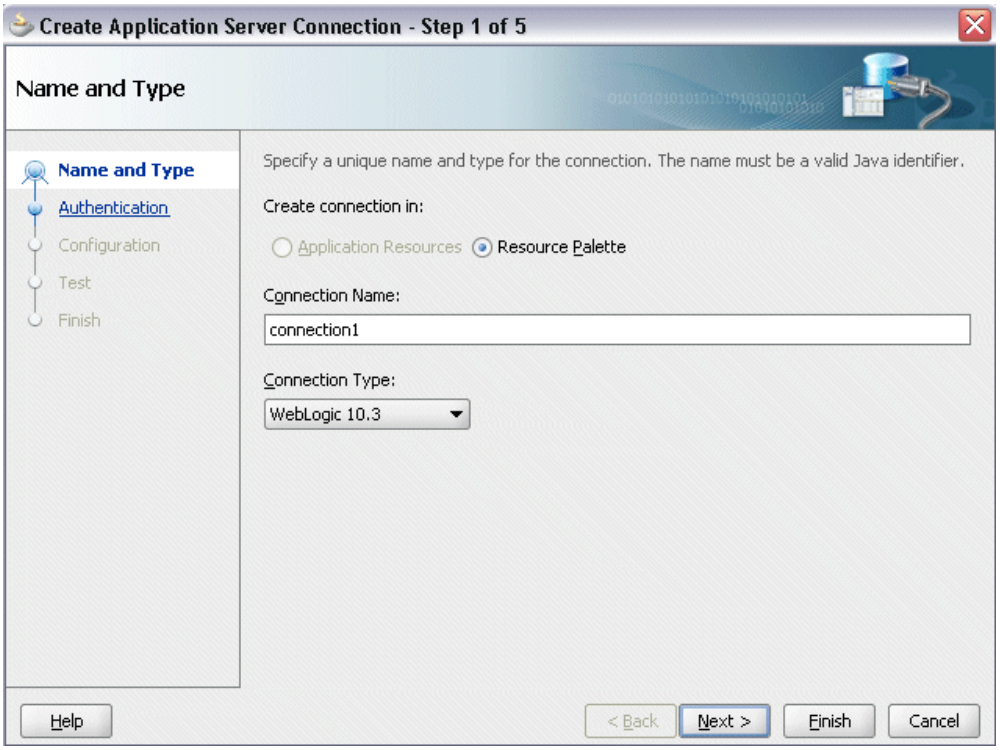


2. Right-click on the Application Server and select **New Application Server** to open the Create Application Server Connection wizard.

3. Enter the connection name and select **WebLogic 10.3** as the connection type.

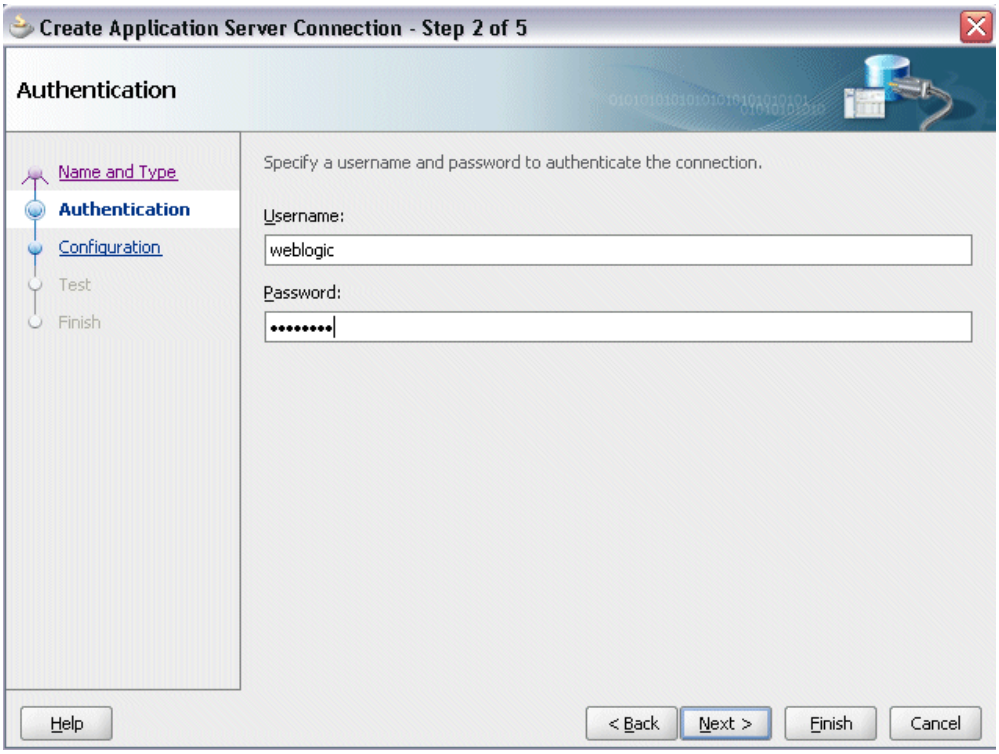
   Click **Next**.

Select the Server Name and Type



4. Enter a valid username (such as `weblogic`) and the password information (such as `weblogic`) and click **Next**.

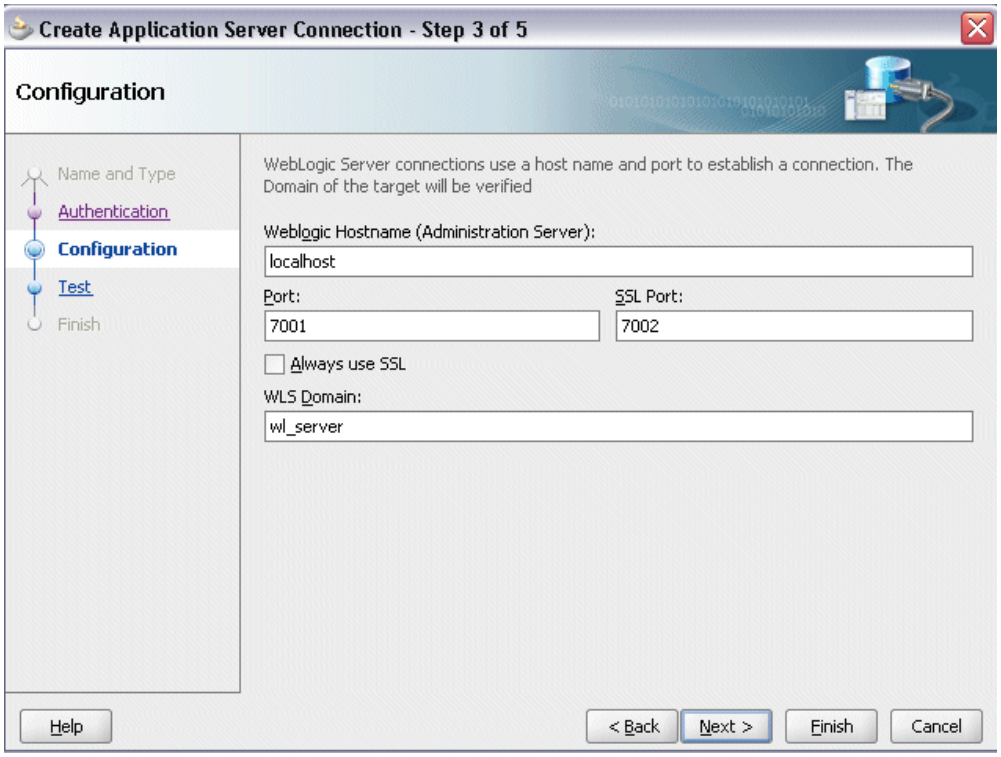
Enter Server Authentication Information



5. Enter the WebLogic Server connection host name and port information. Leave the default domain name unchanged in the WLS Domain field.

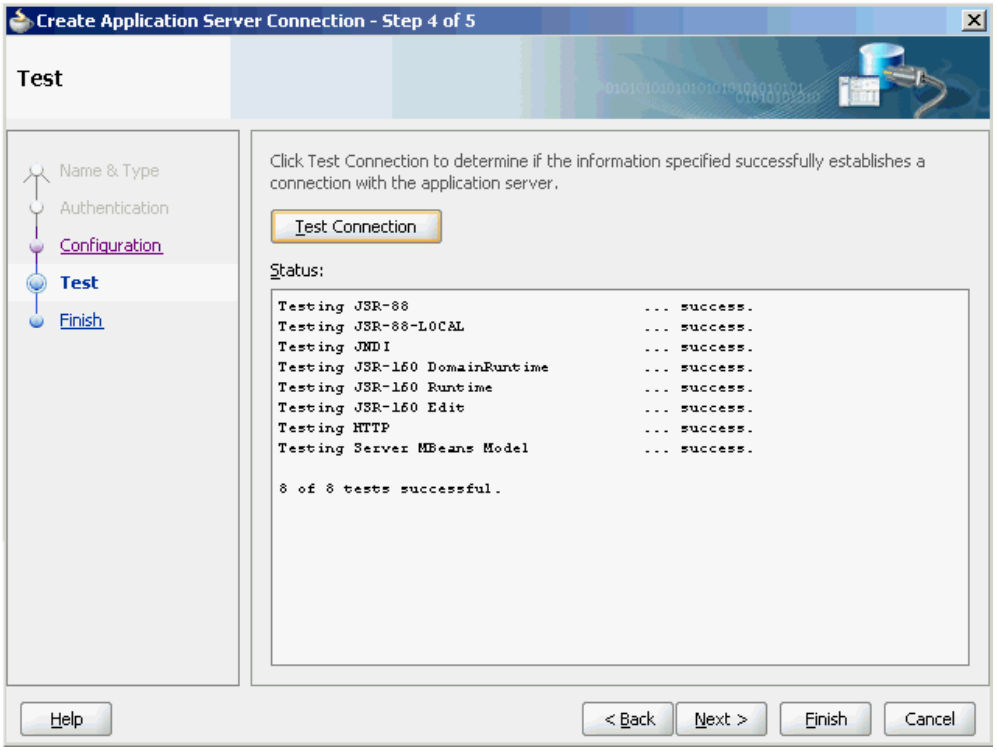
   Click **Next**.

Specifying Server Configuration Information



6. Click **Test Connection** to validate your server configuration. You should find success messages populated in the Status window.

Validating the Server Configuration



Click **Finish**.

Creating the Oracle Applications Adapter Database Connection

The deployment descriptor of the Adapter for Oracle Applications must associate with the Java Naming and Directory Interface (JNDI) name with configuration properties required by the adapter to access the database.

Use the following steps to configure the JNDI and additional settings for Adapter for Oracle Applications through the Oracle WebLogic Server Administration Console:

1. Navigate to `http://servername:portnumber/console`.

2. Enter the server administrator's username and password to log on to the Oracle WebLogic Server Administration Console.

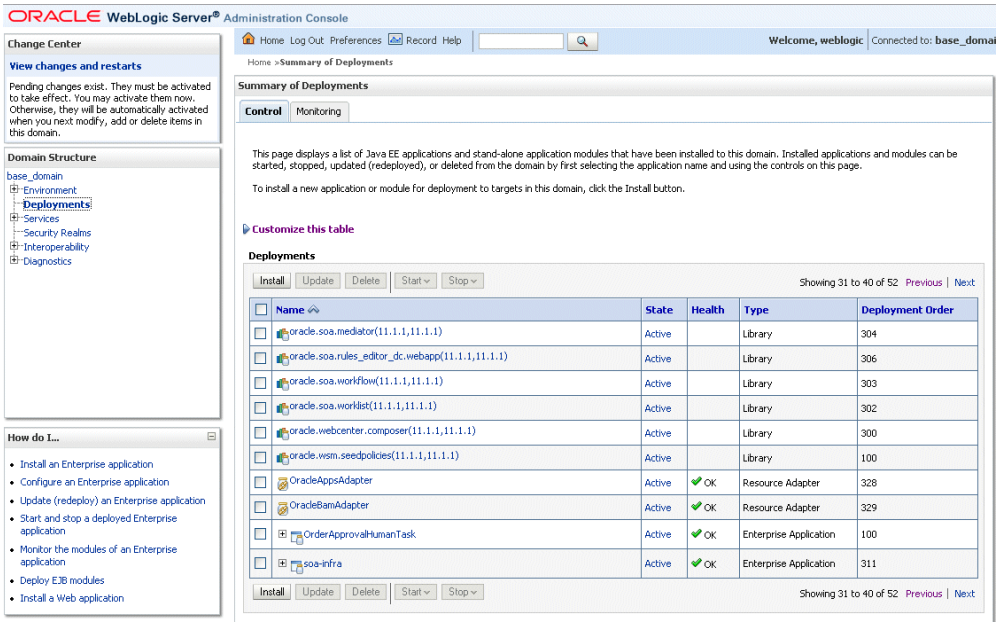
   The Home page of the Oracle WebLogic Server Administration Console appears.

3. Under the Domain Structure section, select **Deployment**.

   The Summary of Deployments page appears where you can find a list of Java EE applications and standalone application modules that have been installed to this domain.

4. Locate an appropriate link for Oracle Adapter for Oracle Applications from the summary of deployment table first and then click the link.

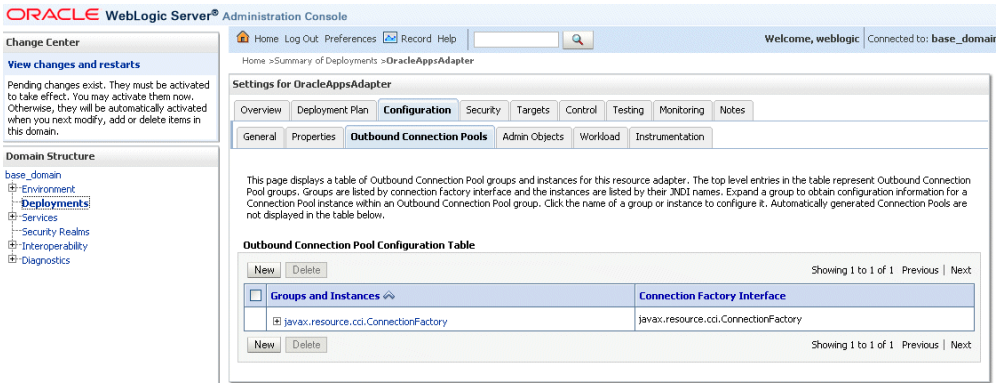
Summary of Deployments



The Overview Settings for the selected application name appears.

5. Click the **Configuration** tab and the **Outbound Connection Pools** subtab to display the Outbound Connection Pool groups and instances for this selected resource adapter.

Outbound Connection Pool Configuration Table

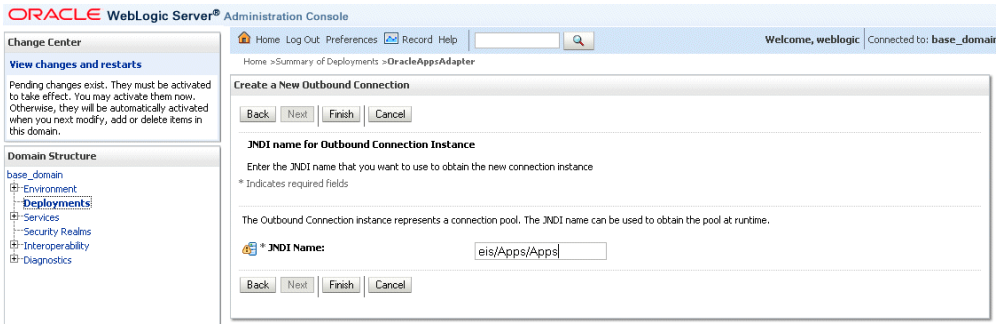


You can click the `javax.resource.cci.ConnectionFactory` link to expand this group and obtain configuration information for a Connection Pool instance.

6. Click **New** and select the `javax.resource.cci.ConnectionFactory` radio button to create an instance. Click **Next** to create a new outbound connection.

7. Enter a JNDI name that you want to use to obtain the new connection instance. For example, enter `eis/Apps/Apps`.

Create a New Outbound Connection



The Outbound Connection instance represents a connection pool. The JNDI name can be used to obtain the pool at runtime.

Click **Finish**.

8. This opens the Save Deployment Plan Assistant page where you can find the new deployment plan information including plan path with an XML extension (`.xml`), recently used paths, and current location.

> **Note:** The plan path must have an XML extension (`.xml`). It is recommended that this file be called `Plan.xml`, and that each plan be located in a unique directory. If the plan file exists it will be overwritten. Other files in the plan directory may be overwritten as well.

9. Click **OK** to save the configuration change for a deployment plan.

You may find some unexpected exception messages populated while processing the request. Use the following steps to resolve the issue:

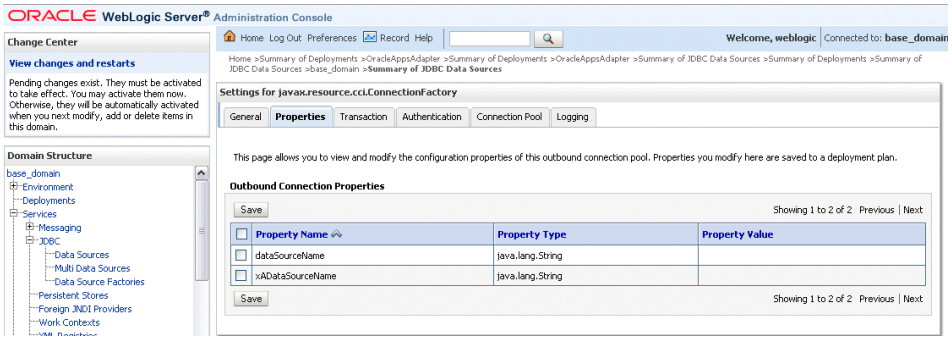
1. Click the Adapter link that you selected earlier on the breadcrumb navigation path.

2. Click the **Configuration** tab and the **Outbound Connection Pools** subtab to display the Outbound Connection Pool groups and instances.

   Click the `javax.resource.cci.ConnectionFactory` expand node to display the newly configured JNDI (such as `eis/Apps/Apps`) for the outbound instance in the configuration table.

3. Click the outbound instance name (such as `eis/Apps/Apps`) to display the outbound connection properties.

Outbound Connection Properties Page

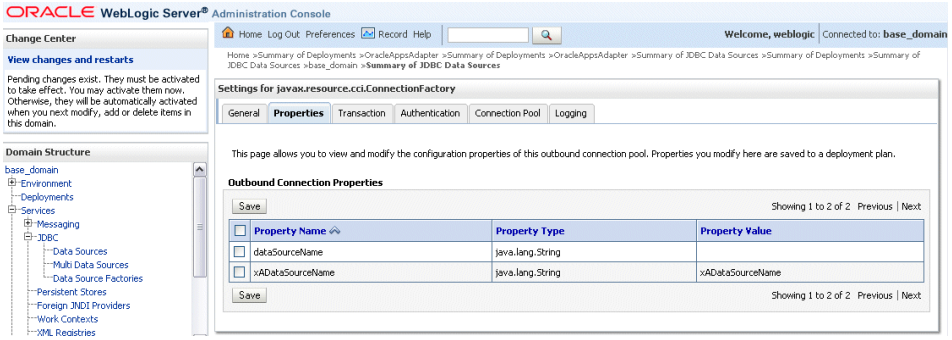


4. Place your mouse in the Property Value column for the 'xADataSourceName' property name and click to enable the input text box.

   Enter `xADataSourceName` in the text box, press [Enter] key, and then click **Save** to update the deployment plan.

   The Summary of JDBC Data Source page appears with `xADataSourceName` listed in the table.

Outbound Connection Properties Page



B

Troubleshooting and Workarounds

This appendix covers the following topics:

- General Issues and Workarounds

General Issues and Workarounds

This section describes the following issues and workarounds:

- **Populating Default Values for Record Types While Using PL/SQL APIs**

  Certain PL/SQL APIs exposed from Oracle E-Business Suite take record types as input. Such APIs expect default values to be populated for parameters within these record types for successful execution.

  The default values are `FND_API.G_MISS_CHAR` for characters, `FND_API.G_MISS_DATE` for dates, and `FND_API.G_MISS_NUM` for numbers. Oracle Adapter for Oracle Applications can default these values when the parameters within the record type are passed as nil values, for example, as shown below:

```
<PRICE_LIST_REC>
<ATTRIBUTE1 xsi:nil="true"/>
<ATTRIBUTE2 xsi:nil="true"/>
<ATTRIBUTE3 xsi:nil="true"/>
...
</PRICE_LIST_REC>
```

  This can be achieved with the help of a function in a Transform activity, or by directly passing the XML input with nil values and then assigning them to the record types within an Assign activity.

- **Recreating Wrapper Packages While Using Existing PL/SQL SOA Composites Against a Different Release Instance**

  When a user has a SOA composite of a PL/SQL API created against an Oracle E-Business Suite Release 11i instance and intends to use it against the Release 12 instance or vice versa, for the compatibility in the target instance, the wrapper

package of the SOA composite must be recreated. This approach updates the signature in the generated wrapper SQL file for the target instance and avoids the possible confusion whether the signature is the same or has changed in the target instance.

- **WSDL Context Information Default Values**

  Applications context is used in passing header variables that may be required in a business activity or to complete a BPEL process. When setting the context, it takes into account the values passed for the header properties including *Username*, *Responsibility*, *Responsibility Application*, *Security Group*, and *NLS Language*.

  If the values for the new header properties *Responsibility Application*, *Security Group*, and *NLS Language* are not passed, context information will be determined based on *Username* and *Responsibility*.

  The default Username is `SYSADMIN`, the default Responsibility is `SYSTEM ADMINISTRATOR`, the default Security Group Key is `Standard`, and the default NLS Language is `US`.

  You can change the default values specified in the generated WSDL. 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 values. The context information can be specified by configuring an Invoke activity.

  For more information about applications context, see Supporting for Normalized Message Properties, page 3-8.

- **Correlation ID Defaults to BPEL for XML Gateway Transactions**

  The Adapter Configuration wizard of Oracle Adapter for Oracle Applications does not specify a correlation ID for XML Gateway transactions for inbound or outbound interfaces. Instead, a default correlation ID of `BPEL` is automatically set in the WSDL file. To make this configuration work, you must configure Oracle Applications to set the same correlation ID value of `BPEL` for the corresponding XML Gateway transactions.

  If you want the Adapter to use a different correlation ID than the default, you need to configure a correlation ID in Oracle Applications, then edit the `Correlation="BPEL"` line contained in the `<jca:operation>` section of the adapter service WSDL. Replace `BPEL` with the string value of the correlation ID you specified in Oracle Applications.

- **Workaround for Stored Procedures Using Complex Types and the DEFAULT Clause**

  When working with stored procedures for which the Adapter Configuration wizard must generate wrapper SQL stored procedures, there is a current limitation on DEFAULT clauses not being carried over to the generated wrapper stored

procedures.

As a workaround, perform the following steps one time only for a given stored procedure:

1. Open the generated wrapper SQL script.
2. Copy all default clauses from the base-stored procedure into the corresponding wrapper.
3. Use SQL*Plus to reload the wrapper SQL script into the database.
4. Edit the generated XSD. If a parameter has a DEFAULT clause, its corresponding element in the XSD must have the extra attribute: `db:default="true"`

   For example, with the following SQL:

   ```
   FINANCE$INVOICE(isTrue INTEGER DEFAULT 1, value NUMBER DEFAULT 0)
   ```

   The elements in the XSD for `isTrue` and `value` must have the new attribute:

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

- **Cannot Create a Partner Link If the Underlying API Has Been Recreated**

  The generation of a wrapper for an API that was recreated with the same name, but with a different set of parameters, will fail.

  > **Note:** This can happen for both packaged procedures and top-level or root procedures that require generated wrappers.

  The following example illustrates the problem:

  1. Create the initial API that, in this case, is defined at the top level:

     ```
     SQL> create procedure test (a number, b varchar2, c BOOLEAN)
     ```

     The `BOOLEAN` parameter indicates that a wrapper is necessary.

  2. Use the database adapter for stored procedures in the Adapter Configuration Wizard to generate and load the wrapper for this API.
  3. Drop the API, then recreate it with a different set of parameters:

     ```
     SQL> drop procedure test
     SQL> create procedure test (a number, b varchar2, c number, d
     BOOLEAN)
     ```

  4. An attempt to generate a partner link for this API using the Adapter Configuration Wizard will fail with the following message:

     ```
     The wrapper procedure, TOPLEVEL$TEST, could not be found
     ```

5. As a workaround, exit JDeveloper BPEL Designer and restart it after recreating the stored procedure, but before attempting to create the second partner link.

Index

A

Adapter for Oracle Applications
  architecture, 1-3
  features, 1-2
  installing, 1-4
  integration with BPEL PM, 1-4
  integration with Oracle WebLogic Server, 1-5
  interfaces, 2-1
  issues and workarounds, B-1
  New Features, 1-6
  Oracle E-Business Suite Support, 2-4
  Oracle Integration Repository, 2-3
  overview, 1-1
Adapter for Oracle Applications Concepts
  Applications Context, 3-1
  Function Security, 3-15
  Module Browser, 3-35
  Using J2EE Data Source, 3-34
agent
  See business events concepts
APIs
  See PL/SQL APIs
applications context
  example, 3-2
Applications Context
  Design-Time Tasks, 3-9
  Multiple Language Support, 3-14
  multiple organization setups, 3-12
  Normalized Message Properties, 3-8

B

B2B, 4-2
base tables, 7-2
business events, 5-1
  debugging, 5-45
  troubleshooting, 5-44
business events concepts, 5-3
  agent, 5-4
  deferred subscription processing, 5-3
  event, 5-3
  event activity, 5-3
  event data, 5-3
  event key, 5-3
  event message, 5-3
  event subscription, 5-3
business events outbound
  Assign activity, 5-29
  configuring Adapter for Oracle Applications, 5-4
  creating a new BPEL project, 5-5
  creating a partner link, 5-10
  deploying the BPEL process, 5-34
  design-time prerequisites, 5-5
  design-time tasks, 5-4
  file adapter partner link, 5-21
  Invoke activity, 5-27
  Receive activity, 5-19
  run-time tasks, 5-33
  testing the BPEL process, 5-36

C

concurrent program
  adding a new partner link for file adapter, 6-

30
concurrent programs
adding a partner link, 6-7
configuring Adapter for Oracle Applications, 6-2
configuring the Assign activity, 6-48
configuring the Invoke activity, 6-41
creating a new BPEL project, 6-3
debugging, 6-62
deploying the BPEL process, 6-54
design-time steps, 6-2
overview, 6-1
prerequisite to configure, 6-2
run-time steps, 6-54
testing the BPEL process, 6-57
troubleshooting, 6-61
verifying records, 6-60
connection information, A-2
Applications Database Connection, A-12
data source configuration, A-3
WebLogic Server Connection, A-8

D

deferred subscription processing, 5-3
See also business events concepts
design-time tasks
business events outbound, 5-4

E

EDI
adding a new partner link for file adapter, 9-15
adding a partner link, 9-8
configuring Adapter for Oracle Applications, 9-2
configuring the Assign activity, 9-25
configuring the Invoke activity, 9-20
creating a new BPEL project, 9-3
deploying the BPEL process, 9-30
design-time steps, 9-2
overview, 9-1
prerequisites to configuring Adapter for Oracle Applications, 9-2
run-time steps, 9-30
testing the BPEL process, 9-32
verifying records in Oracle Applications, 9-34
event
See business events concepts
event activity, 5-3
event data, 5-3
event key, 5-3
event message, 5-3
event subscription, 5-3
example WSDL file, A-1

F

Features, 2-1

I

integration architecture, 4-2
interface tables, 7-2
adding a file adapter partner link, 7-23
adding a partner link, 7-6
configuring Adapter for Oracle Applications, 7-3
configuring the Assign activity, 7-31
configuring the Invoke activity, 7-28
creating a new BPEL project, 7-3
deploying the BPEL process, 7-35
design-time steps, 7-3
overview, 7-2
prerequisites to configure, 7-3
run-time steps, 7-35
testing the BPEL process, 7-36

L

Logging
Correlating Messages, 3-32
enabling logging, 3-21
overview, 3-20
Searching and Viewing Logs, 3-26

M

message queues, 4-3
inbound queues, 4-3
outbound queues, 4-4

P

PL/SQL APIs
adding a new partner link, 8-8
adding a new partner link for file adapter, 8-

22
configuring Adapter for Oracle Applications, 8-2
configuring the Assign activity, 8-40
configuring the Invoke activity, 8-33
configuring the Transform activity, 8-38
creating a new BPEL project, 8-4
debugging, 8-49
deploying the BPEL process, 8-43
design-time steps, 8-2
overview, 8-1
prerequisite to configure, 8-2
run-time steps, 8-43
testing the BPEL process, 8-45
troubleshooting, 8-49
prerequisites
business events outbound, 5-5

R

run-time tasks
business events outbound, 5-33

S

standards-based messaging, 4-2

V

views, 7-2
adding a new partner link for file adapter, 7-58
adding a partner link, 7-44
configuring Oracles Adapter for Oracle Applications, 7-39
configuring the Assign activity, 7-68
configuring the Invoke activity, 7-64
creating a new BPEL project, 7-40
deploying the BPEL process, 7-73
design-time steps, 7-39
overview, 7-2
prerequisites to configure, 7-39
run-time steps, 7-72
testing the BPEL process, 7-74

W

wrapper APIs, 8-27
DEFAULT clause handling in wrapper procedures, 8-32
describing parameters with DEFAULT clause, 8-30
WSDL, A-1

X

XML Gateway
debugging, 4-77
overview, 4-2
troubleshooting, 4-76
XML gateway envelope, 4-4
ATTRIBUTE3, 4-6
DOCUMENT_NUMBER, 4-5
MESSAGE_STANDARD, 4-4
MESSAGE_TYPE, 4-4
parameters defined by the application, 4-6
parameters not used, 4-6
PARTY_SITE_ID, 4-5
PASSWORD, 4-5
PAYLOAD, 4-6
PROTOCOL_ADDRESS, 4-5
PROTOCOL_TYPE, 4-5
TRANSACTION_SUBTYPE, 4-5
TRANSACTION_TYPE, 4-4
USERNAME, 4-5
XML gateway inbound
adding a partner link for the file adapter, 4-22
configuring Adapter for Oracle Applications, 4-7
configuring the Assign activity, 4-32
configuring the Invoke activity, 4-26
creating a new BPEL project, 4-9
creating a partner link, 4-12
deploying the BPEL process, 4-36
design-time steps, 4-7
prerequisites to configure, 4-7
run-time steps, 4-36
testing the BPEL process, 4-38
verifying records, 4-40
XML gateway outbound
design-time steps, 4-44
prerequisites to configure, 4-45
run-time steps, 4-68
testing the BPEL process, 4-71
XML Gateway outbound
deploying the BPEL process, 4-68

XML Gateway Outbound
adding a new partner link, 4-51
adding a new partner link for file adapter, 4-59
Adding a Receive Activity, 4-56
configuring the Assign activity, 4-65
configuring the Invoke activity, 4-63
creating a new BPEL project, 4-47