Oracle® Application Server

Adapters for CICS User's Guide

12*c* Release 1 (12.1.3)

E66950-01

November 2015

ORACLE®

Oracle Application Server Adapters for CICS User's Guide, 12*c* Release 1 (12.1.3)

E66950-01

Copyright © 2005, 2011, 2015 Oracle and/or its affiliates. All rights reserved.

Primary Author: David Goldman

Contributor: Sami Zeitoun, Itamar Ankorion

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.

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

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

Preface vii

Audience vii
Documentation Accessibility vii
Related Documents viii
Conventions viii

1 Introduction

OracleAS Adapters for CICS Overview 1-1
OracleAS Adapter for CICS 1-1
OracleAS Adapter for CICS Queue 1-2
CICS Transient Data Queues (Intrapartition) 1-2
OracleAS Adapters for CICS Architecture 1-2
Integration Flow from Oracle Application Server to the Legacy Application 1-3

2 Configuring OracleAS Adapters for CICS

Setting Up the IBM z/OS Platform in Oracle Studio 2-1
Securing Access to Oracle Connect 2-2
Setting Password Access to Oracle Studio 2-2
Specifying Users with Administrative Rights 2-3
Setting Up Run-Time User Access to the IBM z/OS Platform 2-4
Modeling Interactions for OracleAS Adapter for CICS 2-6
Configuring an Oracle Connect Adapter 2-6
Generating Outbound Interactions 2-8
Generating Inbound Interactions 2-14
Modeling Interactions for OracleAS Adapter for CICS Queue 2-20
Preliminary Considerations 2-20
COB_ADL Format 2-24
CICS TD Queue Sample 2-24
The ATYGTEVT CICS Program 2-25
Configuring the CICS Queue Adapter 2-26
Generating Outbound Interactions 2-27
Viewing the XML Schema 2-31
Creating XML Schemas 2-31

3 Integrating OracleAS Adapters for CICS with WebLogic

Overview of Integrating OracleAS Adapters for CICS with WebLogic 3-1
Configuring the CICS Adapter for Outbound 3-2
Configuring the CICS Queue Adapter for Inbound 3-3
High Availability 3-3

4 Integrating OracleAS Adapters for CICS

Configuring OracleAS Adapter for CICS 4-1
Creating JCA Configuration Files 4-1
Configuring the Binding for Outbound Interactions 4-2
Configuring the Binding for Inbound Interactions 4-3

5 Configuring an OracleAS Adapter for CICS for Outbound Interactions

Creating Outbound Interactions with the OracleAS Adapter for CICS 5-1
Requirements 5-1
Create the findDoctor and findPatient Interactions 5-1
Prepare the System 5-2
Set up Machine Access to Oracle Connect 5-2
Add a CICS Adapter 5-2
Import the Metadata 5-3
Create the JCA Configuration Files 5-3

6 Creating the Project in JDeveloper Examples

Getting Started 6-1
Prerequisites 6-1
Beginning Tasks 6-2
Deploy a BPEL Outbound Process 6-2
Design-Time Tasks 6-2
Create a BPEL Project for a BPEL Outbound Process 6-2
Use a Third-Party Adapter to Read the Doctor Data 6-4
Design a BPEL Outbound Process 6-6
Edit the Invoke_1 Activity 6-7
Deploy the Project 6-8
Run-time Configuration 6-8
Deploy a Mediator Process 6-8
Design-Time Tasks 6-8
Create a Mediator Project for a Mediator Outbound Process 6-9
Use a Third-Party Adapter to Read the Doctor Data 6-10
Design an Outbound Mediator 6-12
Create Mapping Rules and Map the Files 6-13
Deploy the Project 6-14

7 Troubleshooting OracleAS Adapter for CICS

Troubleshooting the Daemon 7-1

Starting the Daemon ... 7-1
Task: Starting the Daemon ... 7-1
Shutting Down the Daemon ... 7-2
Monitoring the Daemon During Run time ... 7-2
Daemon (Computer) Options ... 7-2
Workspace Options ... 7-3
Server Options ... 7-3
Daemon Logs ... 7-4
The Daemon Log Monitor ... 7-4
The Workspace Log Monitor ... 7-5
The Server Log Monitor ... 7-5
Resolving Communication Errors ... 7-5
Resolving Specific Errors ... 7-6

8 Advanced Features of OracleAS Adapters for CICS

Configuring the Daemon for High Availability ... 8-1
Adding a New Daemon Workspace Configuration ... 8-1
Editing the Workspace ... 8-2
Configuring the Server Mode ... 8-2
Configuring a Binding Environment ... 8-6
Debug ... 8-7
General ... 8-8
Language ... 8-8
Modeling ... 8-9
ODBC ... 8-9
OLEDB ... 8-9
Optimizer ... 8-9
Query Processor ... 8-9
Parallel Processing ... 8-9
Transactions ... 8-9
Tuning ... 8-10
XML ... 8-10
Migration Considerations ... 8-11
Security Considerations ... 8-11
Setting Design Time Security ... 8-12
Setting Run-time Security ... 8-12
Transaction Support ... 8-13

A Advanced Tuning of the Metadata

Adapter Metadata ... A-1
Adapter Metadata General Properties ... A-1
Adapter Metadata Interactions ... A-3
Adapter Metadata Schema Records ... A-4
Adapter Metadata XML ... A-6

B Advanced Tuning of the Daemon

Control B-1
Logging B-3
Security B-5
Workspaces B-7
General B-7
Server Mode B-11
Security B-16

C Back-end Adapter Data Type Support

Data Type Mapping C-1

D Globalization Settings

Defining the Language and Code Page D-1

E Editing XML Files in Oracle Studio

Preparing to Edit XML Files in Oracle Studio E-1
Making Changes to the XML File E-2
Remove Objects E-2
Add DTD Information E-2
Edit Namespaces E-3
Add Elements and Attributes E-5
Replace an Element E-5

Index

Preface

This guide is the primary source of user and reference information on OracleAS Adapter for CICS, which enables client applications to access transactions running under CICS through the Sun J2EE Connector Architecture (J2CA) API.

This document describes the features of OracleAS Adapter for CICS.

This preface covers the following topics:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

Audience

This manual is intended for Oracle integration administrators who perform the following tasks:

- Installing and configuring OracleAS Adapter for CICS
- Diagnosing errors
- Using OracleAS to access CICS transactions

Note: You should understand the fundamentals of OracleAS, WebLogic, the UNIX and Microsoft Windows operating system before using this guide to install or administer OracleAS Adapter for CICS.

Note: For the purposes of this version of the Oracle Weblogic Server, Oracle Application Server refers to the Oracle WebLogic Server and OracleAS refers to Oracle WLS.

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.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/support/contact.html or visit http://www.oracle.com/accessibility/support.html if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Other Product One Release 7.0 documentation set or in the Oracle Other Product Two Release 6.1 documentation set:

- *Oracle Application Server Adapter Concepts Guide*
- *Oracle Application Server Adapter Installation Guide*
- *Oracle Application Server Adapter Concepts Guide*
- *Oracle Application Server Containers for J2EE User's Guide*
- *Oracle Application Server Containers for J2EE Services Guide*
- *Oracle Application Server Containers for J2EE Security Guide*

Conventions

The following text conventions are used in this document:

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

1 Introduction

Oracle Application Server (AS) Adapters for CICS enable you to connect Oracle Application Server to a Customer Information Control System (CICS).

This chapter provides an overview of the features and architecture of OracleAS Adapter for CICS.

This chapter contains the following topics:

- OracleAS Adapters for CICS Overview
- OracleAS Adapters for CICS Architecture

1.1 OracleAS Adapters for CICS Overview

OracleAS Adapter for CICS include the following adapters:

- OracleAS Adapter for CICS
- OracleAS Adapter for CICS Queue

CICS is a family of application servers that provides industrial-strength, online transaction management and connectivity for mission-critical applications on IBM mainframe systems.

The application components in CICS are executable programs called transactions. These programs are typically written in COBOL but may possibly be written in other languages, such as PL/I, C, and Java. OracleAS Adapter for CICS interaction are mapped to specific programs.

1.1.1 OracleAS Adapter for CICS

OracleAS Adapter for CICS includes the following features:

- Uses `COMMAREA` transactions to communicate with clients, using a communication buffer described in the transaction program (such as a COBOL copybook). There is an input structure and an output structure (each with a limit of 32Kb). In many cases, transactions are written so that the input structure prefixes the output structure.
- Uses EXCI (EXternal Call Interface) to invoke CICS transactions from tasks running within the IBM z/OS system. EXCI is efficient because it uses cross-memory mechanisms.
- Enables storing valid CICS user names and passwords, which are passed to CICS for authentication as part of the client request.

- Captures and maintains a metadata schema for CICS by importing COBOL copybooks and transforming them into mapping definitions for Oracle Connect on the IBM z/OS Series platform and by associating the data structures with specific physical files.
- Uses an enterprise application integration (EAI) model. Users of a requesting application can model the interactions that they want implemented, specifying what each interaction does, when it occurs, and the inputs and outputs expected for each interaction.
- Maps data structures to facilitate access to CICS programs within Oracle Application Server.
- Supports two-phase commit and can fully participate in a distributed transaction.

Note: To use OracleAS Adapter for CICS with 2PC, you must install and configure RRS and also install CICS TS 1.3 or higher.

1.1.2 OracleAS Adapter for CICS Queue

OracleAS adapter for CICS Queue is used for pulling messages from a CICS transient data queue.

To pull message from the queue, the `getEvents` interaction is executed. This interaction is added to the adapter schema automatically. When executed, all messages in the queue are pulled.

1.1.2.1 CICS Transient Data Queues (Intrapartition)

Intrapartition refers to data on direct-access storage devices used with one or more programs running as separate tasks. Data directed to or from these internal queues must consist of variable-length records. All intrapartition transient data destinations are held as queues in the same data set, managed by CICS.

An intrapartition destination requires a resource definition containing information that locates the queue in the intrapartition data set. These queues can be associated with either a terminal or an output data set. When data is written to the queue by a user task, the queue can be used subsequently as input data by other tasks within the CICS region.

All access is sequential, goverened by read and write pointers. Once a record has been read, it cannot be read by another task. Intrapartition data may ultimately be transmitted upon request to the terminal or retrieved sequentially from the output data set.

The intrapartition transient data queue is of the Logically Recoverable type. This data queue type is recovered on warm and emergency restarts. If a unit of work (UOW) updates a logically recoverable queue, and subsequently backs out the changes it has made, the changes made to the queue are also backed out. On a warm or emergency restart, the committed state of a logically recoverable intrapartition queue is recovered.

1.2 OracleAS Adapters for CICS Architecture

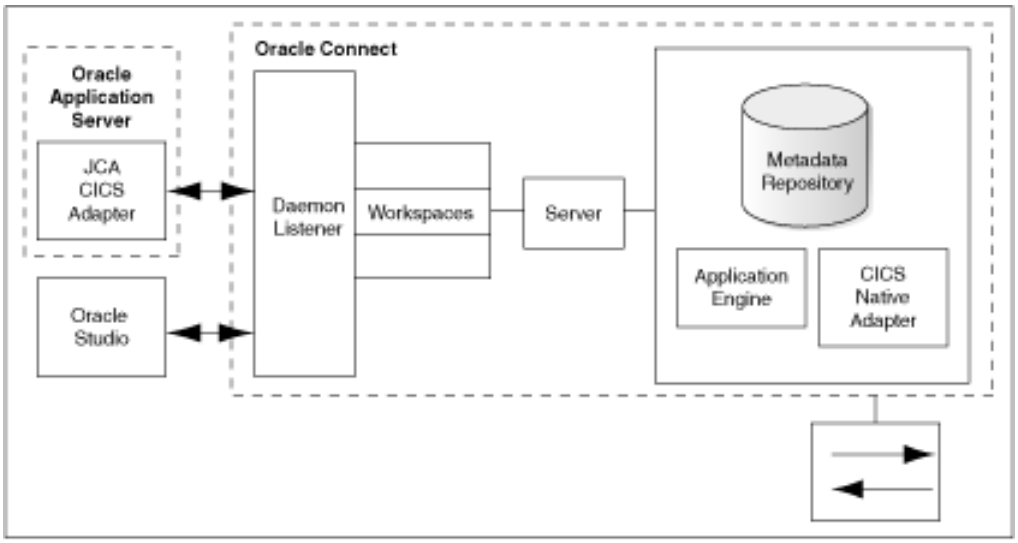
OracleAS Adapter for CICS include the following components:

- J2CA 1.5 CICS adapter: The J2CA CICS adapter is a standard resource adapter that is compliant with J2EE Connector Architecture, providing J2EE components connectivity.

- Oracle Connect: Oracle Connect runs on the legacy system and handles requests from the J2CA 1.5 CICS adapter, that runs within Oracle Application Server Containers for J2EE).
- Oracle Studio: Oracle Studio is the configuration tool for Oracle Connect. Configuration tasks using Oracle Studio are performed on a Windows or Linux computer. Oracle Studio uses perspectives that enables you to generate specific information necessary to model OracleAS Adapter for CICS.

  Figure 1–1 illustrates the components of OracleAS Adapter for CICS:

Figure 1–1 OracleAS Adapter for CICS Architecture



See Also: *Oracle Application Server Adapter Concepts Guide*

1.2.1 Integration Flow from Oracle Application Server to the Legacy Application

The J2CA 1.5 CICS adapter converts the J2CA interaction invocation received from Oracle Application Server to XML format and passes the XML format to Oracle Connect on the legacy server. The daemon listens for the request coming from the J2CA 1.5 CICS adapter client and assigns a server process to handle the request.

The properties of the server process such as connection pooling requirements are defined by a workspace definition within the daemon. The server process includes an instance of the application engine, which converts the XML format into native structures understandable by CICS and passes the converted XML to the back-end adapter. The back-end adapter builds an interaction based on the metadata for the back-end adapter stored in the repository and the incoming converted XML, and passes it to the legacy application. The results of this execution are passed back to the application engine, using the back-end adapter, where these results are converted to XML and passed back to the client.

2

Configuring OracleAS Adapters for CICS

This chapter describes how to configure Oracle Connect using Oracle Studio.

All modeling of Oracle Connect is performed using Oracle Studio. To use Oracle Studio, you first configure it to enable access to the IBM z/OS platform where CICS runs.

This chapter includes the following topics:

- Setting Up the IBM z/OS Platform in Oracle Studio
- Securing Access to Oracle Connect
- Modeling Interactions for OracleAS Adapter for CICS
- Modeling Interactions for OracleAS Adapter for CICS Queue
- Viewing the XML Schema
- Creating XML Schemas

Note: The following tasks assume you have permission to access the IBM z/OS platform and that the Oracle Connect daemon is running on this computer.

Check with the system administrator to ensure these requirements are fulfilled.

2.1 Setting Up the IBM z/OS Platform in Oracle Studio

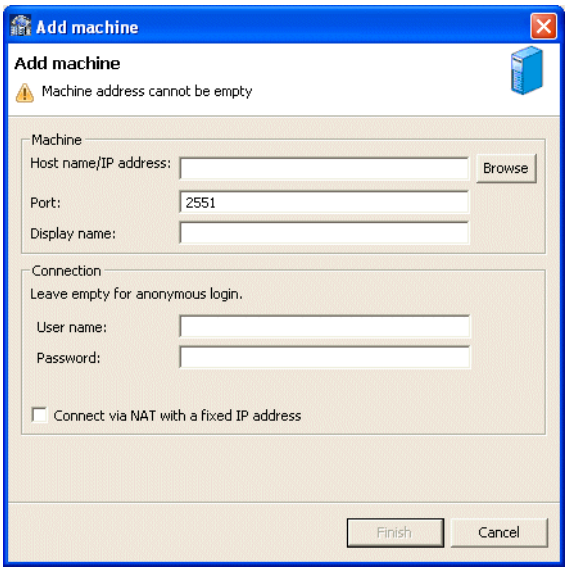
Perform the following steps to configure the IBM z/OS, using Oracle Studio:

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. Right-click **Machines** in the Configuration Explorer and select **Add Machine**. The Add Machine screen is displayed, as shown in Figure 2–1.
3. Enter the name of the computer you want to connect to, or click **Browse** to select the computer from the list of computers that is displayed and uses the default port, `2551`.
4. Enter the user name and password of the user who was specified as the administrator when Oracle Connect was installed.

Note: Selecting **Anonymous connection** enables anyone having access to the computer to be an administrator.

The Add Machine screen is shown in the following figure:

Figure 2–1 The Add Machine screen



5. Click **Finish**.

   The computer is displayed in the Configuration Explorer.

2.2 Securing Access to Oracle Connect

Oracle Studio includes mechanisms to secure access to Oracle Connect both during modeling and at run time.

During modeling the following security mechanisms can be applied:

- Setting Password Access to Oracle Studio
- Specifying Users with Administrative Rights

During run time client access to Oracle Connect is provided by the user profile:

- Setting Up Run-Time User Access to the IBM z/OS Platform

2.2.1 Setting Password Access to Oracle Studio

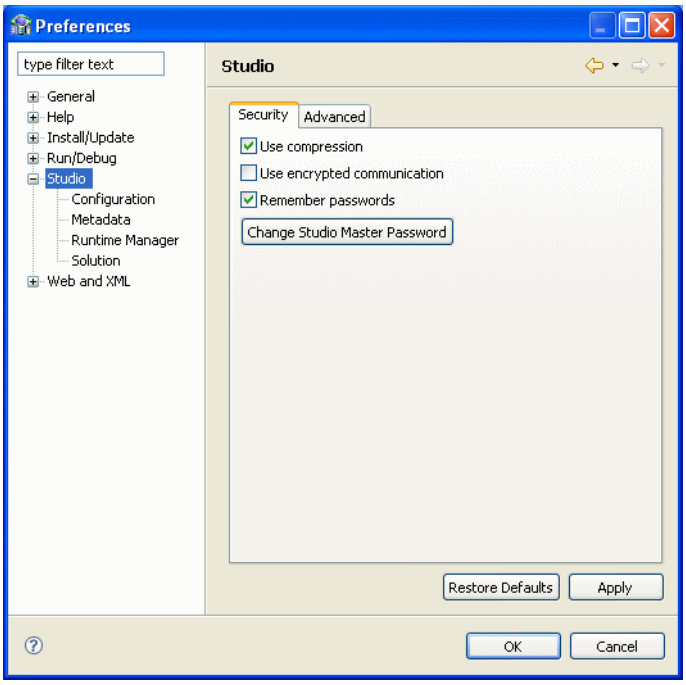
Initially, any operation performed using Oracle Studio does not require a password. You can set a password so that the first operation that involves accessing the server from Oracle Studio requires a password to be entered.

Perform the following steps:

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**. Oracle Studio opens.
2. Select **Window** from the menu bar, and then select **Preferences**.

   The Preferences screen is displayed as shown in the following figure:

Figure 2–2 The Preferences screen



3. Select **Studio**.
4. Click **Change Studio Master Password**.
5. In the Change master password screen, leave the **Enter current master password** field empty and type a new master password.
6. Confirm the password.
7. Click **OK**.

2.2.2 Specifying Users with Administrative Rights

By default, only the user who was specified during the installation as an administrator has the authorization to modify settings on that computer from Oracle Studio. This user can then authorize other users to make changes or to view the definitions for a selected computer. Adding a computer to Oracle Studio is described in "Setting Up the IBM z/OS Platform in Oracle Studio" on page 2-1.

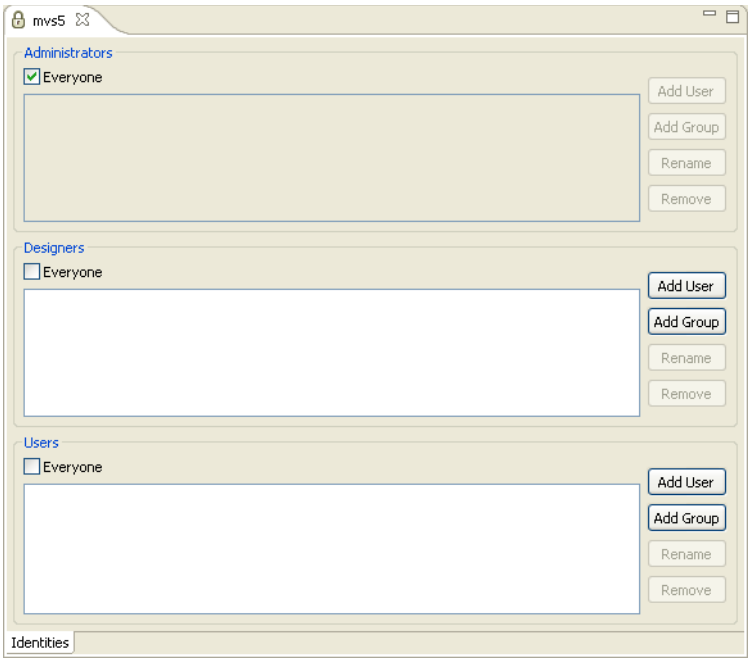
Note: The default during installation is to enable all users to be administrators.

Perform the following steps to specify a user with administrative rights:

1. From the **Start** menu, select **Programs**, **Oracle** and then select **Studio**. Oracle Studio opens, displaying the Design perspective.
2. Right-click the computer in the Configuration Explorer and select **Administration Authorization**.

   The Administration Authorization screen is displayed, as shown in the following figure:

Figure 2–3 The Identities Tab



This screen has the following sections:

Administrators: Administrators can view and modify all the definitions in Oracle Studio for the selected computer. On initial entry to Oracle Studio, every user is defined as a system administrator.

Designers: Designers can view all the definitions for the computer in Oracle Studio and can modify any of the definitions under the Bindings and Users nodes for the selected computer. For example, Oracle Studio database administrator can add new data sources and adapters and can change metadata definitions for a table in a data source.

Users: Users can view all the definitions for the computer in Oracle Studio for the selected computer. Regular users cannot modify any of the definitions.

3. Add users or groups of users by clicking **Add User** or **Add Group** for the relevant sections. The user or group that is added must be recognized as a valid user or group for the computer.

   Once a name has been added to a section, only the user or group that logs on with that user name has the relevant authorization.

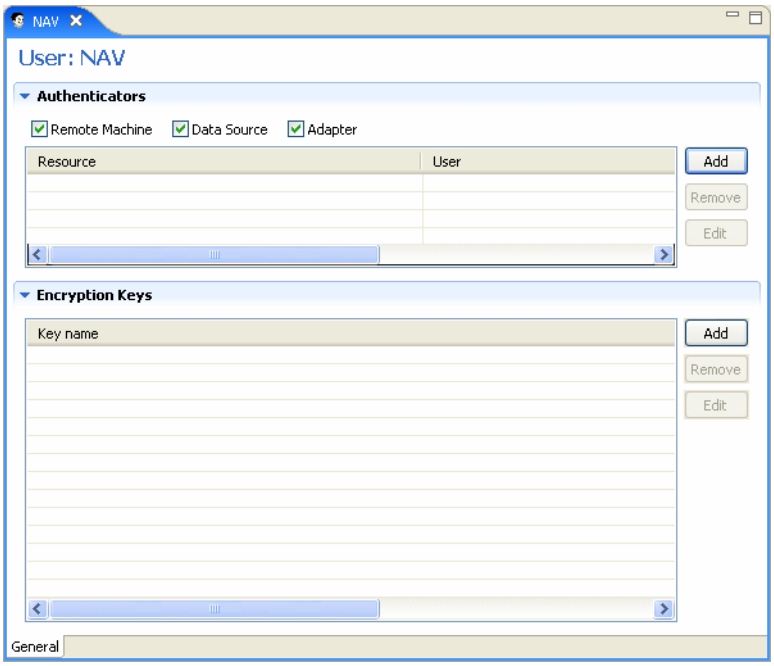
2.2.3 Setting Up Run-Time User Access to the IBM z/OS Platform

During run time, client access to Oracle Connect is provided by the user profile. A user profile contains name and password pairs that are used to access a computer, data source or application during run time, when anonymous access is not allowed.

1. From the **Start** menu, select, **Programs**, **Oracle**, and then select **Studio**. Oracle Studio opens.
2. From the Design perspective, Configuration view, expand the **Machines** folder, then expand the machine where you want to set the user name and password.
3. Expand **Users**.

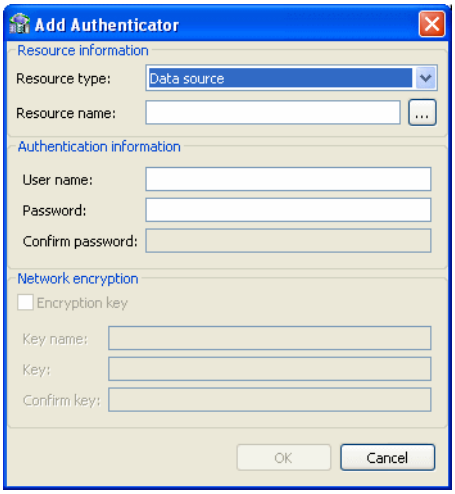
4. Right-click the **NAV** user profile and select **Edit User**. The User editor is displayed, as shown in the following figure:

Figure 2–4 The User Editor pane



5. In the User editor, click **Add** to display the Add Authenticator screen.
6. Select **Remote Machine** from the **Resource type** list, as shown in the following figure:

Figure 2–5 The Add Authenticator screen



7. Enter the name of the IBM z/OS computer defined in Oracle Studio.
8. Enter the name and password used to access the computer and confirm the password.
9. Click **OK**.

2.3 Modeling Interactions for OracleAS Adapter for CICS

Modeling interactions for OracleAS Adapter for CICS involve defining an Oracle Connect back-end adapter using Oracle Studio. All the definitions specified in Oracle Studio are written to the IBM z/OS platform.

This section contains the following topics:

- Configuring an Oracle Connect Adapter
- Generating Outbound Interactions
- Generating Inbound Interactions

2.3.1 Configuring an Oracle Connect Adapter

To work with the Oracle Connect, you must configure adapter definitions on the IBM z/OS platform to handle the interactions to and from a CICS program. You use Oracle Studio to specify these definitions.

Perform the following steps:

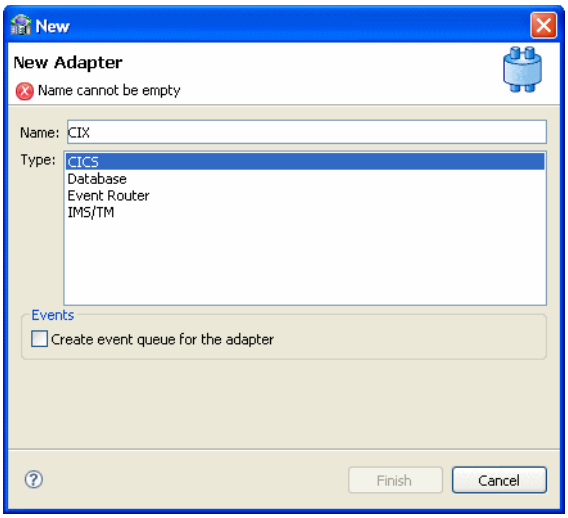
1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. In the Design perspective, Configuration view, expand **Machine** folder.
3. Expand the machine defined in "Setting Up the IBM z/OS Platform in Oracle Studio" on page 2-1.
4. Expand the **Bindings**. The binding configurations available on this computer are listed.
5. Expand the **NAV** binding. The NAV binding configuration includes branches for data sources and adapters that are located on the computer.
6. Right-click **Adapters** and select **New Adapter** to start the New Adapter wizard.
7. Enter a name for the back-end adapter.

Note: The word *event* is a reserved word and cannot be used to name an adapter.

---

8. Select **CICS** as the back-end adapter type from the **Type** list as shown in the following figure:

Figure 2–6 The New Adapter wizard



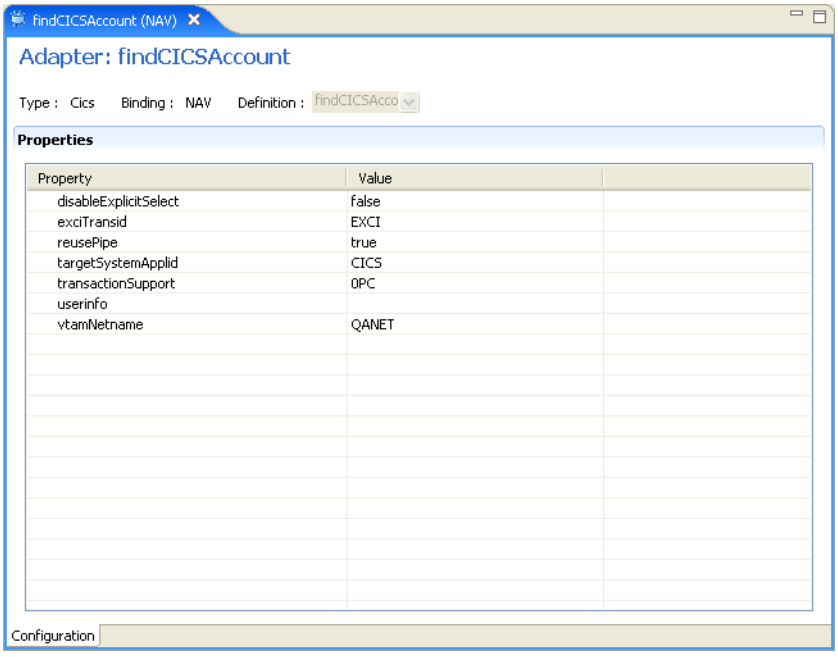
9. Select **Create event queue for the adapter**.
10. Click **Finish**. The back-end adapter is added to the list and the definition opens for editing.

> **Note:** Other adapters that are displayed in the **Type** list are not supported with the version of Oracle Connect installed at the site.

11. Click the **Properties** tab and change any of the properties for the adapter, as required.

    The **Properties** tab is shown in the following figure:

Figure 2–7 The Adapter Properties tab



The **Properties** tab comprises fields, as listed in the following table:

Table 2–1 CICS Adapter Properties

Property	Description
`exciTransid`	The CICS TRANSID. This value must be EXCI or a copy of this transaction
`targetSystemApplid`	The VTAM applid of the CICS target system. You can determine this value by activating the `CEMT` transaction on the target CICS system. On the bottom right corner of the screen appears the legend `APPLID=`*`target_system`*.
`transactionSupport`	For future use. The level of transaction support for this adapter that is reported to the transaction manager In the current version leave this value as `0PC`
`vtamNetname`	The VTAM netname of the specific connection being used by EXCI (and MRO) to relay the program call to the CICS target system. For example, if you issue to `CEMT` the following command: `CEMT INQ CONN` You see on the display screen that the netname is BATCHCLI (this is the default connection supplied by IBM upon the installation of CICS). If you plan to use the IBM defaults, then specify `BATCHCLI` as the VTAM_netname parameter, otherwise, define a specific connection (with EXCI protocol) and use the netname you provided there for this parameter.

2.3.2 Generating Outbound Interactions

Oracle Connect requires metadata describing the adapter interactions, including the structures used to pass information to and from the adapter.

If COBOL copybooks describing the adapter input and output structures are available, then you can import the adapter definition using the Metadata Import wizard in Oracle Studio Design perspective to generate interaction metadata. If the metadata is provided in multiple COBOL copybooks, with different filter settings (such as whether the first six columns are ignored or not), then you first import the metadata from copybooks with the same settings and later import the metadata from the other copybooks.

If COBOL copybooks describing the input and output structures are not available, then you must manually define the metadata.

The following information is required during the import procedure:

The COBOL copybooks: These are copied to the computer running Oracle Studio as part of the import procedure.

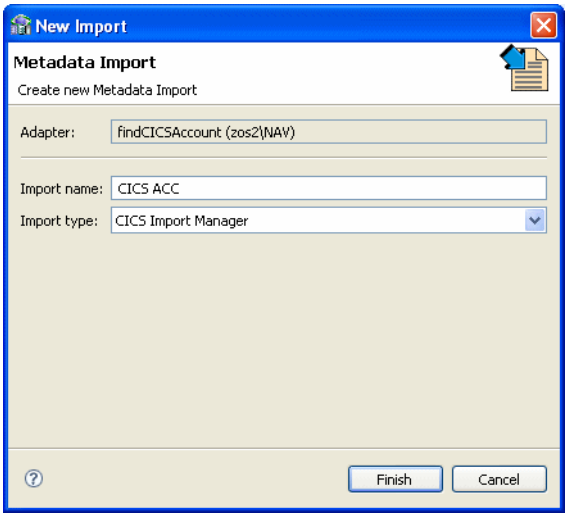
The names of the CICS programs run through the application adapter.

Use the Metadata Import procedure using Oracle Studio to generate interaction metadata, as follows:

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. In the Design perspective, Configuration view, expand **Machine** folder.
3. Expand the machine defined in "Setting Up the IBM z/OS Platform in Oracle Studio" on page 2-1.

4. Expand the **Bindings**. The binding configurations available on this computer are listed.
5. Expand the **NAV** binding. The NAV binding configuration includes branches for data sources and adapters that are located on the computer.
6. Expand the **Adapters** folder.
7. Right-click the CICS adapter you defined in "Configuring an Oracle Connect Adapter" on page 2-6 and select **Show Metadata View** to display the Metadata view, with the CICS back-end adapter displayed under the **Adapters** list.
8. Right-click the **Imports** node and select **New Import**. The New Metadata Import screen is displayed.
9. Enter a name for the import. The name can contain letters and numbers and the underscore character only.
10. Select **CICS Import Manager** from the **Import type** list, as shown in the following figure:

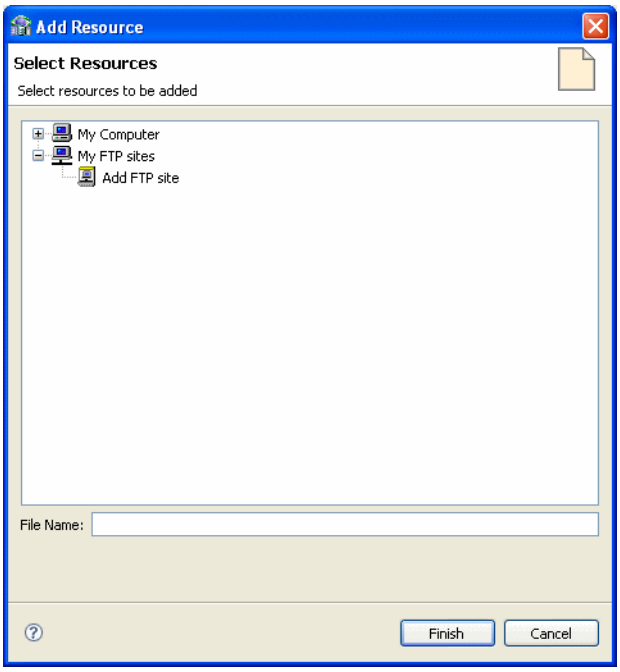
Figure 2–8 The Metadata Import wizard



1. Click **Finish**. The Metadata Import wizard is displayed.
2. Click **Add**. The Select Resources screen is displayed, which provides the option to select files from the local computer or copy the files from another computer.
3. If the files are on another computer, then right-click **My FTP Sites** and select **Add**. Optionally, double-click **Add FTP site**. The Add FTP Site screen is displayed.
4. Enter the server name or IP address where the COBOL copybooks reside and enter a valid user name and password to access the computer (if anonymous access is used, click the Anonymous connection check box) then click **OK**.

   The FTP site is added to the list of available sites, as shown in the following figure:

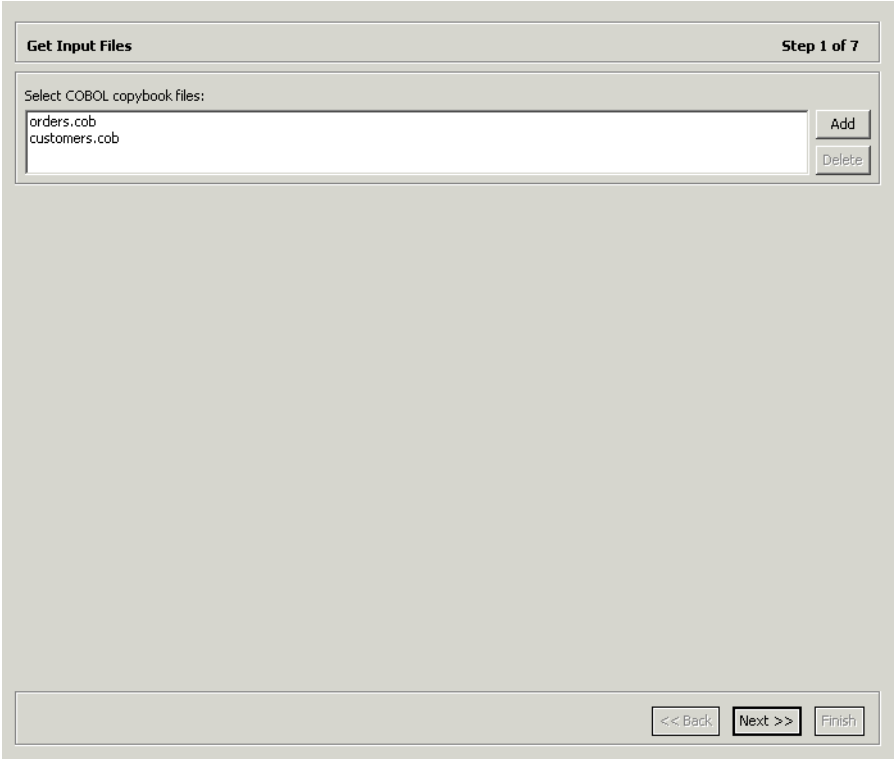
Figure 2–9 The Select Rosources screen



Note: The selected server is accessed using the user name as the high-level qualifier, which enables you to browse and transfer files.

5. Right-click the computer and select **Set Transfer Type**. Enter the transfer type (`ASCII` or `BINARY`) and click **OK**.
6. Expand the list of the added site until you find the necessary COBOL files. To change the high-level qualifier, right-click the computer and select Change Root Directory. Enter the new high-level qualifier enclosed in quotes, and click **OK**.
7. Select the required COBOL copybook file or files and click **Finish**. The selected file or files are displayed in the Metadata Import wizard as shown in the following figure:

Figure 2–10 The Get Input Files screen

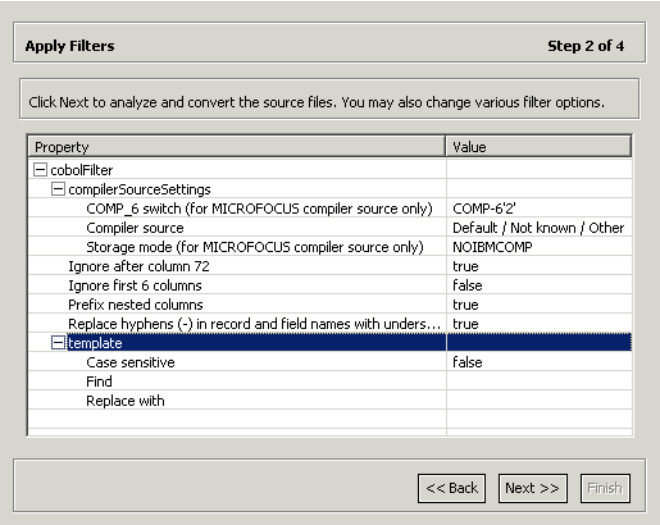


Note: You can import the metadata from one COBOL copybook and later add to this metadata by repeating the import procedure using different COBOL copybooks.

The format of the COBOL copybooks must be identical. That is, you cannot import a COBOL copybook that uses the first six columns with a COBOL copybook that ignores the first six columns. In this type of case you must repeat the import procedure.

8. Click **Next**. The Apply Filters screen is displayed as shown in the following figure.

Figure 2–11 The Apply Filters screen



9. Apply filters to the copybooks, as needed.

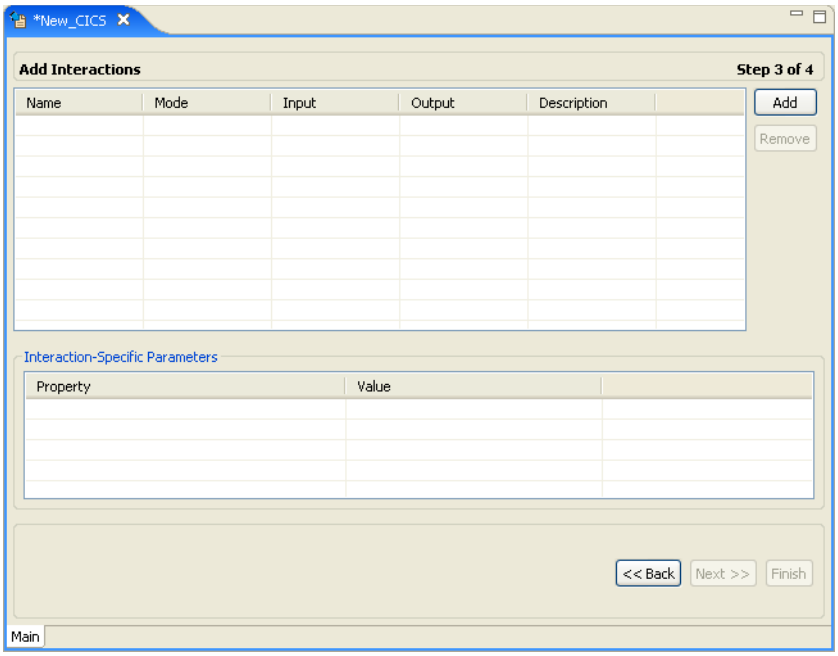
The following table lists the available filters:

Table 2–2 Metadata Filters

Filter	Description
COMP_6 switch	The MicroFocus COMP-6 compiler directive. Specify either COMP-6'1' to treat COMP-6 as a COMP data type or COMP-6'2' to treat COMP-6 as a COMP-3 data type.
Compiler source	The compiler vendor.
Storage mode	The MicroFocus Integer Storage Mode. Specify either `NOIBMCOMP` for byte storage mode or `IBMCOMP` is for word storage mode.
Ignore after column 72	Ignores columns 73 to 80 in the COBOL copybook.
IgnoreFirst6	Ignores the first six columns in the COBOL copybook.
Replace hyphens (-) in record and field names with underscores (_)	Replaces all hyphens in either the record or field names in the metadata generated from the COBOL with underscore characters.
Prefix nested columns	Prefix all nested columns with the previous level heading.
Case sensitive	Specifies whether to be sensitive to the search string case.
Find	Searches for the specified value.
Replace with	Replaces the value specified for Find with the value specified here

10. In the import wizard, click **Next**. The Add Interactions screen is displayed, as shown in the following figure:

Figure 2–12 The Add Interactions screen



11. Click **Add** to add an interaction for the CICS adapter. Provide the information as listed in the following table:

Table 2–3 Interaction Parameters

Parameter	Description
`Name`	The name of the interaction. You can change the default name specified.
`Mode`	The interaction mode. You can select one of the following: ■ sync-send-receive: The interaction sends a request and expects to receive a response. This is the default mode. ■ sync-receive: The interaction expects to receive a response ■ sync-send: The interaction sends a request and does not expect to receive a response.
`Input`	Identifies an input record. The input record is the data structure for the interaction. The records generated from the COBOL files specified at the beginning of the procedure are listed. Select the relevant record for the interaction. Note: You must specify an input record for each interaction before you can click **Next**. If the interaction does not require an input record, then the record specified here is ignored.
`Output`	Identifies an output record. The output record is the data structure for the results of the interaction. The records generated from the files specified at the beginning of the procedure are listed. Select the relevant record for the interaction. **Note**: You must specify an output record for the interaction if the mode is set to sync-send-receive or sync-receive, before you can click **Next**.
`Description`	Free text describing the interaction.

Table 2–3 (Cont.) Interaction Parameters

Parameter	Description
`Interaction-Specific Parameters`	CICS specific parameters: ▪ `commitMode`: The interaction commit mode. ▪ `maxSegmentSize`: The maximum size of the segment. The default value is 32768 (bytes). ▪ `transaction`: The name of the transaction that runs. You must specify a transaction name for each interaction. The `maxSegmentSize` parameter enables dynamically splitting large messages into smaller segments. Although the default value can be changed, the logic of CICS transaction must correspond accordingly. The transaction must perform a GU call followed by a series of GN calls to compile the entire input message. A transaction output message, larger than 32K, is handled by the OTMS C/I interface which performs the task of assembling the output segments into a single buffer.

12. Add as many interactions as necessary and click **Next** to generate the metadata definitions for the adapter.
13. For **Do you want to transfer the metadata to the server**, click **Yes** to transfer the data from the Windows computer to the IBM z/OS computer and click **Finish**.

    The metadata is imported based on the options specified and it is stored on the IBM z/OS platform. An XML representation of the metadata is also generated.

    After performing the import, you can view the metadata in Oracle Studio Design perspective **Metadata** tab. You can also make any fine adjustments to the metadata and maintain it, as necessary.

    See Also: Appendix A, "Advanced Tuning of the Metadata" for details about fine tuning the adapter metadata.

2.3.3 Generating Inbound Interactions

Inbound interactions are defined as events in Oracle Studio. When you defined the Oracle Connect for CICS back-end adapter with *Events* selected, as described in Configuring an Oracle Connect Adapter on page 2-6, an event adapter was defined automatically. The event adapter is defined with the same name as the back-end adapter with the word *event* appended to it. The back-end adapter and the event adapter are linked by Oracle Studio.

You can skip from the adapter definition to the event definition by right-clicking the adapter or event in the Configuration Explorer list and selecting the **Linked Event** or **Linked Adapter** option, respectively.

The event adapter requires metadata describing the inbound interactions, including the structure used to pass information.

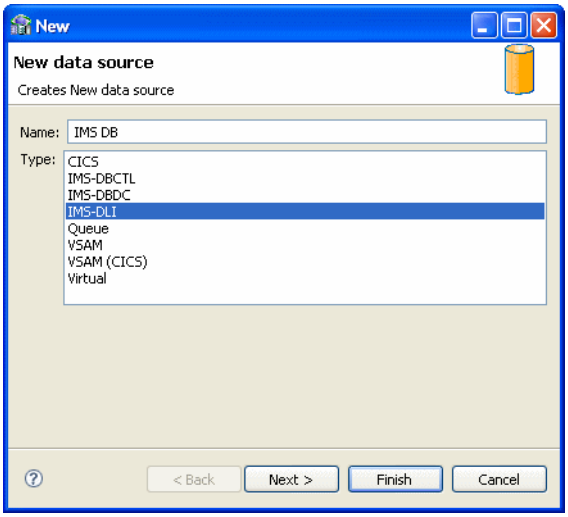
Note: The generation of inbound interactions involves similar steps to the steps described to generate outbound interactions. For details, see "Generating Outbound Interactions" on page 2-8.

Use the Metadata Import procedure in Oracle Studio to generate interaction metadata, as follows:

1. In the Configuration Explorer, right-click the CICS back-end adapter defined in "Configuring an Oracle Connect Adapter" on page 2-6.
2. Select **Linked Event**, to skip to the event adapter.
3. Right-click the event adapter and select **Edit Event**.
4. Select the **Properties** tab to add the names of Oracle Application Server users who can retrieve inbound interactions and z/OS users who can send inbound interactions.
5. To add Oracle Application Server users, expand the **Routers** node and right-click the users property to add the user. The Properties tab is shown in the following figure:
6. Enter the name of Oracle Application Server user in the Value column for the item added.
7. To add z/OS users, expand the **Senders** node and right click the users property to add the user.
8. Enter the name of the z/OS user in the Value column for the item added.
9. Click **Save** to save the changes.
10. Right-click the event adapter in the Configuration Explorer and select **Edit metadata**, to display the Metadata tab, with the event adapter displayed under the Events list.
11. Right-click the Imports node and select **New Import**. The New Import screen is displayed.
12. Enter a name for the import. The name can contain letters and numbers and the underscore character only.
13. Select **Event Queue Import Manager Using Cobol COPYBOOK Input Files** as the import type, and click **Finish**.

    The New Import screen is shown in the following figure:

Figure 2–13 The New Import screen



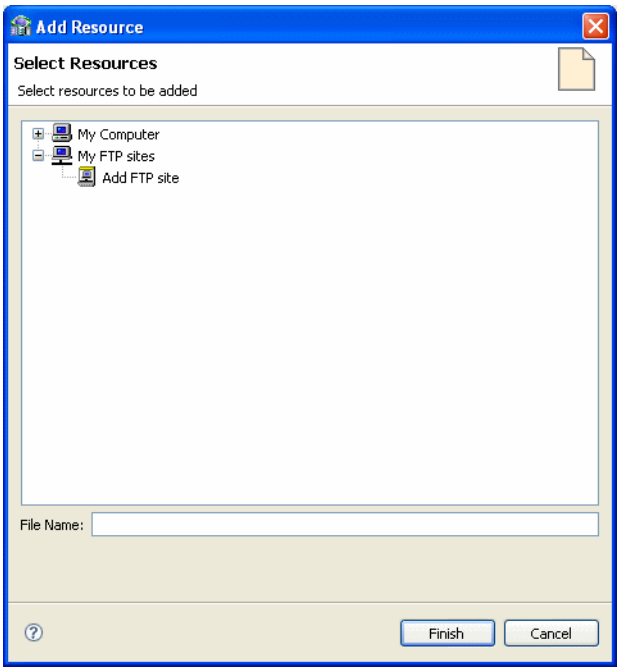
After defining an import type, the metadata import wizard opens in Oracle Studio. COBOL copybooks are used to create the metadata. The import wizard generates record structures, which are used for the record structures for inbound interactions.

1. Click **Add** in the Metadata Import wizard. The Select Resources screen is displayed, which provides the option to select files from the local computer or copy the files from another computer.
2. If the files are on another computer, then right-click **My FTP Sites** and select **Add**. Optionally, double-click **Add FTP Site**. The Add FTP Site screen is displayed.
3. Enter the server name or IP address where the COBOL copybooks reside and enter a valid user name and password to access the computer (if anonymous access is used, click **Anonymous connection**) then click **OK**. The FTP site is added to the list of available sites.

   Note: The selected server is accessed using the user name as the high-level qualifier, enabling you to browse and transfer files.

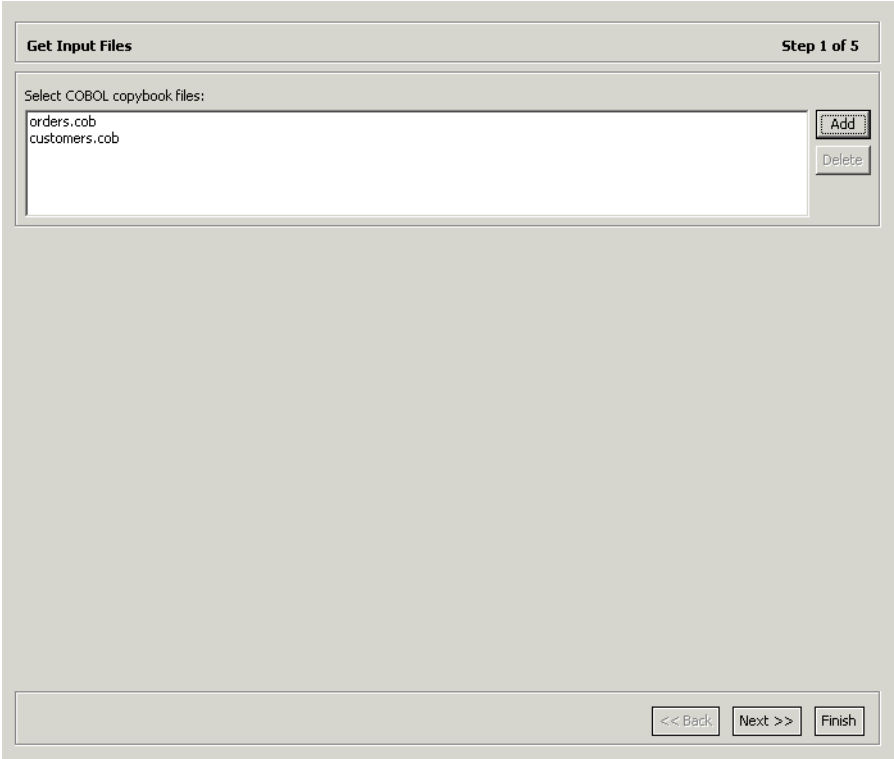
   The Add Resource screen is shown in the following figure:

Figure 2–14 The Add Resources screen



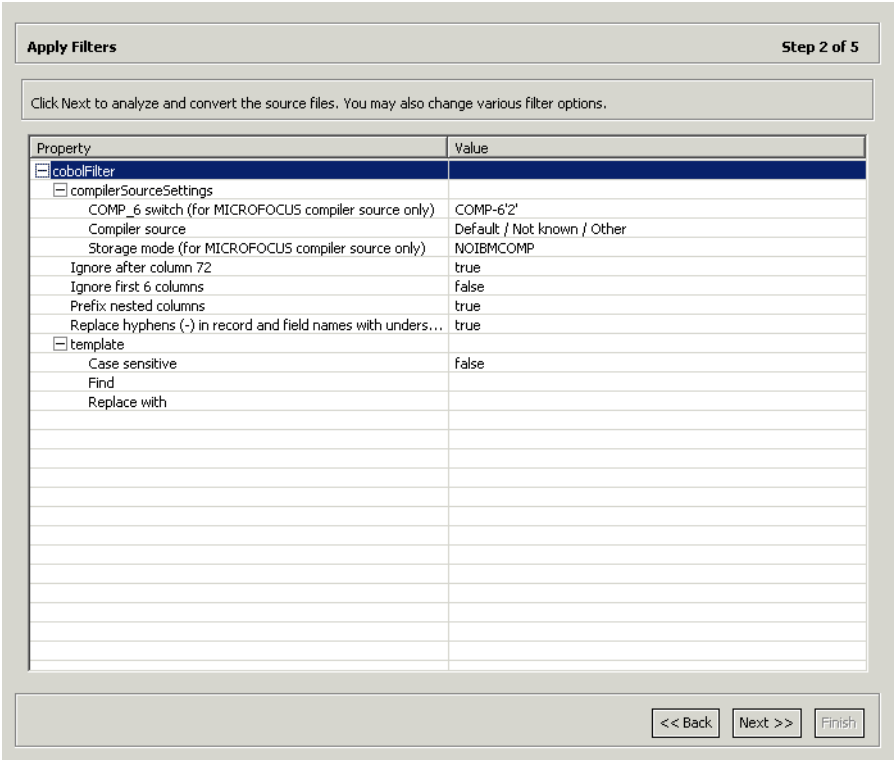
4. Right-click the computer and select **Set Transfer Type**. Enter the transfer type (ASCII or BINARY) and click **OK**.
5. Expand the node of the added site and locate the necessary COBOL files. To change the high-level qualifier, right-click the computer and select **Change Root Directory**. Enter a high-level qualifier enclosed in quotes, and click **OK**.
6. Select the file or files and click **Finish**. The selected file or files are displayed in the metadata import wizard, as shown in the following figure:

Figure 2–15 The Get Input Files screen



7. In the Import wizard, click **Next**. The Apply Filters screen is displayed as shown in the following figure:

Figure 2–16 The Apply Filters screen



8. Apply filters to the copybooks. The following table lists the available filters:

Table 2–4 Metadata Filters

Filter	Description
COMP_6 switch	The MicroFocus COMP-6 compiler directive. Specify either COMP-6'1' to treat COMP-6 as a COMP data type or COMP-6'2' to treat COMP-6 as a COMP-3 data type.
Compiler source	The compiler vendor.
Storage mode	The MicroFocus Integer Storage Mode. Specify either NOIBMCOMP for byte storage mode or IBMCOMP is for word storage mode.
Ignore after column 72	Ignores columns 73 to 80 in the COBOL copybook.
IgnoreFirst6	Ignores the first six columns in the COBOL copybook.
Replace hyphens (-) in record and field names with underscores (_)	Replaces all hyphens in either the record or field names in the metadata generated from the COBOL with underscore characters.
Prefix nested columns	Prefix all nested columns with the previous level heading.
Case sensitive	Specifies whether to be sensitive to the search string case.
Find	Searches for the specified value.
Replace with	Replaces the value specified for Find with the value specified here

9. Click **Next**. The Add Events screen is displayed.

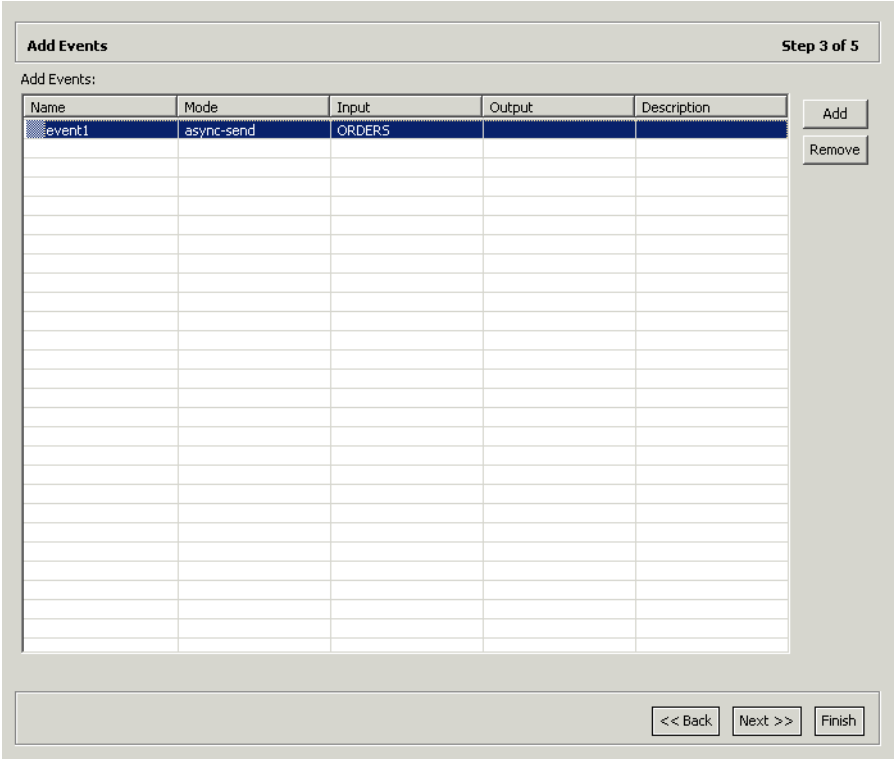
10. Click **Add** to add an event for the CICS adapter and provide the following information:

Table 2–5 Event Parameters

Parameter	Description
`Name`	The name of the event. You can change the default name specified.
`Mode`	The event mode. You can select the **async-send** mode, which is the event sends a request and does not expect to receive a response.
`Input`	Identifies an input record. The input record is the data structure for the event. The records generated from the COBOL files specified at the beginning of the procedure are listed. Select the relevant record for the event. Note: You must specify an input record for each event before you can click **Next**. If the interaction does not require an input record, then the record specified here is ignored.
`Description`	Free text describing the event.

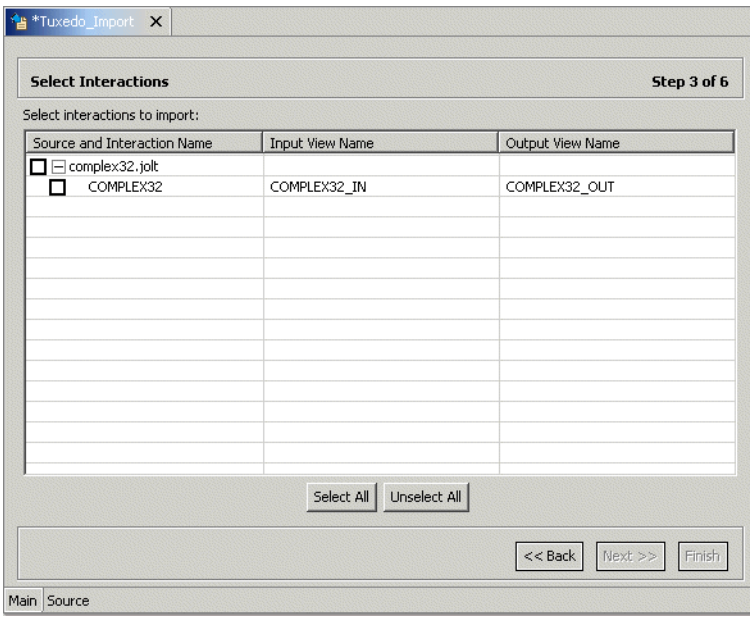
The Add Events screen is shown in the following figure:

Figure 2–17 The Add Events screen



11. Add the required events.
12. Click **Next** to generate the metadata definitions for the adapter.
13. For **Do you want to transfer the metadata to the server**, click **Yes** to transfer the data from the Windows computer to the IBM z/OS computer, and click **Finish**, as shown in the following figure:

Figure 2–18 The Import Metadata screen



The metadata is imported based on the options specified and it is stored on the IBM z/OS platform. An XML representation of the metadata is also generated.

After performing the import, you can view the metadata in Oracle Studio Design perspective **Metadata** tab. You can also make any fine adjustments to the metadata and maintain it, as necessary.

> **See Also:** Appendix A, "Advanced Tuning of the Metadata" for details about fine tuning the adapter metadata.

2.4 Modeling Interactions for OracleAS Adapter for CICS Queue

Modeling interactions for OracleAS Adapter for CICS Queue involves defining an Oracle Connect back-end adapter using Oracle Studio. All the definitions specified in Oracle Studio are written to the IBM z/OS platform.

This section contains the following topics:

- Preliminary Considerations
- Configuring the CICS Queue Adapter
- Generating Outbound Interactions

2.4.1 Preliminary Considerations

CICS TD queues can receive events with different structures. Therefore it is essential to appropriately pre-plan the mapping of these events, based on the various applications that write to these queues. If events with different structures are written to a queue, then use variants. These variants enable you to pre-define the fields which are used to identify the events that are read from the queue.

Queuing systems use headers which provide metadata for events written to its queues. When headers are used, each event is made of the header part and the data part. The header contains fields which represent the common fields of all the variants defined in the main record (`AR_RECKEY` in "Adapter Definition" on page 2-21), and variants which define multiple event types.

The adapter definition includes the `eventHeader` boolean parameter, which determines the structure of the event records. When set to `true`, it indicates that the payload of each event has the following structure:

```
<event-name>
  <header... header-attributes/>
  <data ...>
   ...
  </data>
</event-name>
```

Where *`header-attributes`* represent the common fields of all the variants defined.

If simple records are used, then the header field contains no fields and the data is simply the entire named record. When complex records are used (in this case, with a variant field), then the remaining header fields are the fields from the first level of the header's `eventsRecord` field, and the data part for a specific instance of an event is the `Case` of the variant field, as defined by the header's `eventsVariantField` parameter.

When the `eventHeader` parameter is set to `false`, CICS Queue handles the payload of each event as a single record structure. This record can be simple or complex (with variants). This is useful when the queue contains only one type of events.

The COBOL copybook used for the CICS Queue adapter metadata should be prepared in advance. This file should include the various event types that are sent to the CICS Queue. Next, you should prepare a file in `cob_adl` format (for more information, see "COB_ADL Format" on page 2-24), which includes the variant description: the redefined record, the selector field and the values for the variant cases in the order of appearance in the COBOL copybook (these files are used in the metadata import procedure for the OracleAS Adapter for CICS Queue).

The methodology for preparing a single, suitable COBOL copybook that is imported as part of the Queue adapter metadata definition, is as follows:

- Copying a common root record which includes a SELECTOR field, and placing it as the root record in a new copybook.
- Copying the REDEFINED records, and placing them under the root record.
  - The first record is copied as is.
  - The subsequent records are also copied, but a code line, which sets the redefinition of the first record (REDEFINE), is added to each of them.

Example 2–1 Adapter Definition

Consider the following adapter definition sample. It is the metadata generated after importing the required files for the Queue adapter.

```
<?xml version="1.0" encoding="UTF-8"?>
<adapterDef type="CICSQ" transactionLevelSupport="2PC" name="sela_q1">
   <adapterSpec eventsHeader="false" eventsRecord="AR_RECORD"
    eventsVariantField="VAR_0" />
   <schema noAlignment="true">
      <record name="AR_RECORD">
         <field name="AR_SELECTOR" type="string" nativeType="numstr_u" size="3" />
         <field name="AR_RECKEY" type="AR_RECORD__AR_RECKEY" />
         <field name="VAR_0" type="AR_RECORD__VAR_0" selector="AR_SELECTOR" />
      </record>
      <record name="AR_RECORD__AR_RECKEY">
         <field name="AR_INST" type="string" nativeType="string" size="2" />
      </record>
      <variant implicit="true" name="AR_RECORD__VAR_0">
         <field name="AR_CNTL_RECORD" type="AR_RECORD__VAR_0__AR_CNTL_RECORD"
           case="1" />
         <field name="AR_TIR_RECORD" type="AR_RECORD__VAR_0__AR_TIR_RECORD"
           case="2" />
         <field name="AR_TCR_RECORD" type="AR_RECORD__VAR_0__AR_TCR_RECORD"
           case="3" />
      </variant>
      <record name="AR_RECORD__VAR_0__AR_CNTL_RECORD">
         <field name="AR_CNTL_ROOT" type="AR_RECORD__VAR_0__AR_CNTL_RECORD__AR
           _CNTL_ROOT" />
         <field name="AR_CNTL_SEGMENT" type="AR_RECORD__VAR_0__AR_CNTL_RECORD__AR
           _CNTL_SEGMENT" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_CNTL_RECORD__AR_CNTL_ROOT">
         <field name="AR_CNTL_CTR" type="string" nativeType="numstr_u" size="3" />
         <field name="FILLER" type="string" nativeType="string" size="6" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_CNTL_RECORD__AR_CNTL_SEGMENT">
         <field name="AR_CN_KEY" type="AR_RECORD__VAR_0__AR_CNTL_RECORD__AR_CNTL
           _SEGMENT__AR_CN_KEY" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_CNTL_RECORD__AR_CNTL_SEGMENT__AR_CN_KEY">
```

```
        <field name="AR_CN_TIR_CTR" type="string" nativeType="numstr_u" size="3" />
        <field name="AR_CN_TCR_CTR" type="string" nativeType="numstr_u" size="3" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_TIR_RECORD">
         <field name="AR_TIR_ROOT" type="AR_RECORD__VAR_0__AR_TIR_RECORD__AR_TIR
           _ROOT" />
         <field name="AR_TIR_SEGMENT" array="100" type="AR_RECORD__VAR_0__AR_TIR
           _RECORD__AR_TIR_SEGMENT" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_TIR_RECORD__AR_TIR_ROOT">
         <field name="AR_TIR_CTR" type="string" nativeType="numstr_u" size="3" />
         <field name="FILLER2" type="string" nativeType="string" size="6" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_TIR_RECORD__AR_TIR_SEGMENT">
         <field name="AR_TI_KEY" type="AR_RECORD__VAR_0__AR_TIR_RECORD__AR_TIR
           _SEGMENT__AR_TI_KEY" />
         <field name="AR_TI_HRS_RANGE" type="AR_RECORD__VAR_0__AR_TIR_RECORD__AR
           _TIR_SEGMENT__AR_TI_HRS_RANGE" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_TIR_RECORD__AR_TIR_SEGMENT__AR_TI_KEY">
         <field name="AR_TI_CLASS" type="string" nativeType="string" size="3" />
         <field name="AR_TI_RESID" type="string" nativeType="string" size="1" />
         <field name="AR_TI_COLL" type="string" nativeType="string" size="2" />
         <field name="AR_TI_MAJOR" type="string" nativeType="string" size="4" />
         <field name="AR_TI_SPPGM" type="string" nativeType="string" size="3" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_TIR_RECORD__AR_TIR_SEGMENT__AR_TI_HRS
       _RANGE">
         <field name="AR_TI_LO_HR" type="string" nativeType="numstr_u" size="2" />
         <field name="AR_TI_HI_HR" type="string" nativeType="numstr_u" size="2" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_TCR_RECORD">
         <field name="AR_TCR_ROOT" type="AR_RECORD__VAR_0__AR_TCR_RECORD__AR_TCR
           _ROOT" />
         <field name="AR_TCR_SEGMENT" array="99" type="AR_RECORD__VAR_0__AR_TCR
           _RECORD__AR_TCR_SEGMENT" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_TCR_RECORD__AR_TCR_ROOT">
         <field name="AR_TCR_CTR" type="string" nativeType="numstr_u" size="3" />
         <field name="FILLER3" type="string" nativeType="string" size="6" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_TCR_RECORD__AR_TCR_SEGMENT">
       <field name="AR_TC_KEY" type="AR_RECORD__VAR_0__AR_TCR_RECORD__AR_TCR
         _SEGMENT__AR_TC_KEY" />
       <field name="AR_TC_HOURS" type="string" nativeType="numstr_u" size="2" />
       <field name="AR_TC_TUIT_TYPE" type="string" nativeType="string" size="1" />
      </record>
      <record name="AR_RECORD__VAR_0__AR_TCR_RECORD__AR_TCR_SEGMENT__AR_TC_KEY">
         <field name="AR_TC_COURS" type="string" nativeType="string" size="11" />
      </record>
   </schema>
</adapterDef>
```

The following is a sample of three COBOL copybooks. Note the common `AR-RECORD` record (marked in bold) in each copybook. This field is used as the root record of the final copybook.

```
01  AR-RECORD1.
  03  AR-SELECTOR             PIC 9(03).
    03  AR-RECKEY.
```

```
        05  AR-INST             PIC X(02).
**
**  CONTROL RECORD
**
    03  AR-CNTL-RECORD.
        05  AR-CNTL-ROOT.
            07  AR-CNTL-CTR     PIC 9(03).
            07  FILLER          PIC X(06).
        05  AR-CNTL-SEGMENT.
            07  AR-CN-KEY.
                09  AR-CN-TIR-CTR  PIC 9(03).
                09  AR-CN-TCR-CTR  PIC 9(03).



01  AR-RECORD2.
  03  AR-SELECTOR             PIC 9(03).
    03  AR-RECKEY.
        05  AR-INST             PIC X(02).
**
**  TIR - INSTITUTIONAL TUITION RATES.
**
    03  AR-TIR-RECORD.
        05  AR-TIR-ROOT.
            07  AR-TIR-CTR      PIC 9(03).
            07  FILLER          PIC X(06).
        05  AR-TIR-SEGMENT   OCCURS 100 TIMES.
            07  AR-TI-KEY.
                09  AR-TI-CLASS PIC X(03).
                09  AR-TI-RESID PIC X(01).
                09  AR-TI-COLL  PIC X(02).
                09  AR-TI-MAJOR PIC X(04).
                09  AR-TI-SPPGM PIC X(03).
            07  AR-TI-HRS-RANGE.
                09  AR-TI-LO-HR PIC 9(02).
                09  AR-TI-HI-HR PIC 9(02).



01  AR-RECORD3.
  03  AR-SELECTOR             PIC 9(03).
    03  AR-RECKEY.
        05  AR-INST             PIC X(02).
**  TCR - COURSE TUITION AND FEES.
**
    03  AR-TCR-RECORD.
        05  AR-TCR-ROOT.
            07  AR-TCR-CTR      PIC 9(03).
            07  FILLER          PIC X(06).
        05  AR-TCR-SEGMENT   OCCURS 99 TIMES.
            07  AR-TC-KEY.
                09  AR-TC-COURS PIC X(11).
            07  AR-TC-HOURS     PIC 9(02).
            07  AR-TC-TUIT-TYPE PIC X(01).
```

The following provides an example of the final COBOL copybook. Note the first record has been copied "as is", while the subsequent records have a `REDEFINE` appended to them.

```
01  AR-RECORD.
```

```
  03  AR-SELECTOR             PIC 9(03).
    03  AR-RECKEY.
        05  AR-INST              PIC X(02).
**
**  CONTROL RECORD
**
    03  AR-CNTL-RECORD.
        05  AR-CNTL-ROOT.
            07  AR-CNTL-CTR     PIC 9(03).
            07  FILLER          PIC X(06).
        05  AR-CNTL-SEGMENT.
            07  AR-CN-KEY.
                09  AR-CN-TIR-CTR  PIC 9(03).
                09  AR-CN-TCR-CTR  PIC 9(03).
**
**  TIR - INSTITUTIONAL TUITION RATES.
**
    03  AR-TIR-RECORD   REDEFINES   AR-CNTL-RECORD.
        05  AR-TIR-ROOT.
            07  AR-TIR-CTR      PIC 9(03).
            07  FILLER          PIC X(06).
        05  AR-TIR-SEGMENT   OCCURS 100 TIMES.
            07  AR-TI-KEY.
                09  AR-TI-CLASS PIC X(03).
                09  AR-TI-RESID PIC X(01).
                09  AR-TI-COLL  PIC X(02).
                09  AR-TI-MAJOR PIC X(04).
                09  AR-TI-SPPGM PIC X(03).
            07  AR-TI-HRS-RANGE.
                09  AR-TI-LO-HR PIC 9(02).
                09  AR-TI-HI-HR PIC 9(02).
**
**  TCR - COURSE TUITION AND FEES.
**
    03  AR-TCR-RECORD   REDEFINES   AR-CNTL-RECORD.
        05  AR-TCR-ROOT.
            07  AR-TCR-CTR      PIC 9(03).
            07  FILLER          PIC X(06).
        05  AR-TCR-SEGMENT   OCCURS 99 TIMES.
            07  AR-TC-KEY.
                09  AR-TC-COURS PIC X(11).
            07  AR-TC-HOURS     PIC 9(02).
            07  AR-TC-TUIT-TYPE PIC X(01).
```

2.4.1.1 COB_ADL Format

The following provides an example of the variant description file, which must comply to `cob_adl` format:

variant_record_name, *selector_field_name*, "case1", "case2",..., "caseN"

For the above samples, the variant description file is as follows:

```
AR-CNTL-RECORD, AR-SELECTOR, "1", "2", "3"
```

2.4.1.2 CICS TD Queue Sample

The following is sample of the commands used to define a CICS TD Queue on the z/OS platform:

```
CEDA DEFINE TDQUEUE

 TDqueue      ==> YTST
 Group        ==> ATY
 DEscription  ==>
 TYPE         ==> INTra              Extra | INTra | INDirect
EXTRA PARTITION PARAMETERS
 DAtabuffers    :                    1-255
 DDname         :
 DSname         :
 Sysoutclass    :
 Erroroption    :                    Ignore | Skip
 Opentime       :                    Initial | Deferred
 REWind         :                    Leave | Reread
 TYPEFile       :                    Input | Output | Rdback
 RECORDSize     :                    0-32767
 BLOCKSize      :                    0-32767
 RECORDFormat   :                    Fixed | Variable
 BLOCKFormat    :                    Blocked | Unblocked

INTRA PARTITION PARAMETERS
 Atifacility  ==> File               Terminal | File | System
 RECOVstatus  ==> Logical            No | Physical | Logical
 Facilityid   ==>
 TRAnsid      ==>
 TRIggerlevel ==> 00000              0-32767

Userid       ==>
NDOUBT ATTRIBUTES
WAIT         ==> Yes                No | Yes
WAITAction   ==> Reject             Queue | Reject
NDIRECT PARAMETERS
Indirectname   :
EMOTE PARAMETERS
REMOTEName   ==>
REMOTESystem ==>
```

2.4.1.3 The ATYGTEVT CICS Program

The `ATYGTEVT` CICS program returns events from the CICS Transient Data Queue.

For the CICS programs to work transactionally with the TD queues, the TD queues must be set to `Logically Recoverable`, where only one task at a time can write to the TD queue and only when that task ends (or when it issues a sync-point) can other tasks write to that queue.

To use the ATYGTEVT program, perform the following steps:

1. Copy the program from `NAVROOT.LOAD` to a CICS `DFHRPL` library (such as `CICS.USER.LOAD`).
2. Define the `ATYGTEVT` program under CICS using any available group (such as `ATY` group) as follows:

   ```
   CEDA DEF PROG(ATYGTEVT) G(ATY) LANG(C) DA(ANY) DE(ORACLE CICSQ GET EVENTS)
   ```

3. Install the ATYGTEVT program as follows:

```
CEDA IN G(ATY)
```

2.4.2 Configuring the CICS Queue Adapter

To work with the Oracle Connect, you must configure adapter definitions on the IBM z/OS platform to handle the interactions from a CICS queue. You use Oracle Studio to specify these definitions.

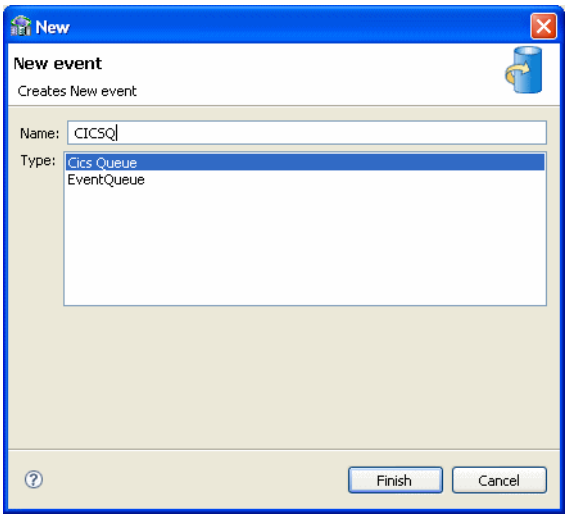
Perform the following steps:

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. Expand the **Machines** folder.
3. In the Design perspective, Configuration view expand the computer defined in "Setting Up the IBM z/OS Platform in Oracle Studio" on page 2-1.
4. Expand the **Bindings**. The binding configurations available on this computer are listed.
5. Expand the **NAV** binding. The NAV binding configuration includes branches for data sources, adapters, and events that are located on the computer.
6. Right-click **Events** and select **New Event** to open the New Event wizard.
7. Enter a name for the queue event.

> **Note:** The word *event* is a reserved word and cannot be used to name an event.

8. Select **CICS Queue** as the event type from the **Type** list, as shown in the following figure:

Figure 2–19 The New Adapter Screen



9. Click **Finish**.

> **Note:** Other events that are displayed in the **Type** list are not supported with the version of Oracle Connect installed at the site.

2.4.3 Generating Outbound Interactions

OracleAS adapter for CICS Queue requires metadata describing the interaction, including the structure used to pass information.

During the import procedure, a COBOL copybook file and a variant description file are copied to the computer running Oracle Studio.

Perform the following steps to generate the interaction metadata:

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. Expand the **Machines** folder.
3. In the Design perspective, Configuration view expand the computer defined in "Setting Up the IBM z/OS Platform in Oracle Studio" on page 2-1.
4. Expand the **Bindings**. The binding configurations available on this computer are listed.
5. Expand the **NAV** binding. The NAV binding configuration includes branches for data sources, adapters, and events that are located on the computer.
6. Expand the **Events** folder and select the CICS Queue you created in "Configuring the CICS Queue Adapter" on page 2-26.
7. Expand the **Bindings**. The binding configurations available on this computer are listed.
8. Right-click the queue adapter and select **Show in Metadata View**.

   The **Metadata** tab is displayed, with the queue adapter displayed under the **Adapters** list.
9. Right-click **Imports** and select **New Import**.

   The New Import screen is displayed.
10. Enter a name for the import. The name can contain letters, numbers and the underscore character only.
11. Select **CICS Queue Import Manager** from the **Import Type** list.
12. Click **Finish**.

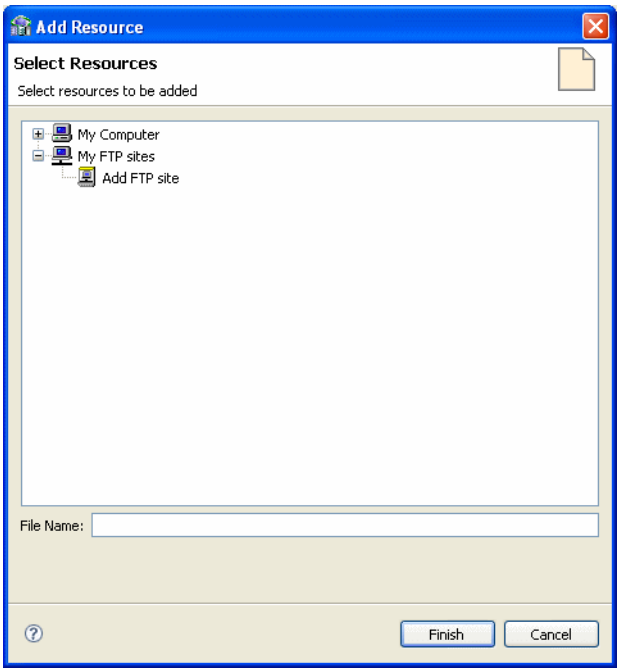
After defining an import type, the metadata import wizard opens in Oracle Studio. COBOL copybooks are used to create the metadata. The import wizard generates record structures, which are used to define the records for outbound interactions.

1. Click **Add** in the Metadata Import wizard. The Select Resources screen is displayed (Figure 2–20), which provides the option to select files from the local computer or copy the files from another computer.
2. If the files are on another computer, then right-click **My FTP Sites** and select **Add**. Optionally, double-click **Add FTP Site**. The Add FTP Site screen is displayed.
3. Enter the server name or IP address where the COBOL copybooks reside and enter a valid user name and password to access the computer (if anonymous access is used, click **Anonymous connection**) then click **OK**. The FTP site is added to the list of available sites.

Note: The selected server is accessed using the user name as the high-level qualifier, enabling you to browse and transfer files.

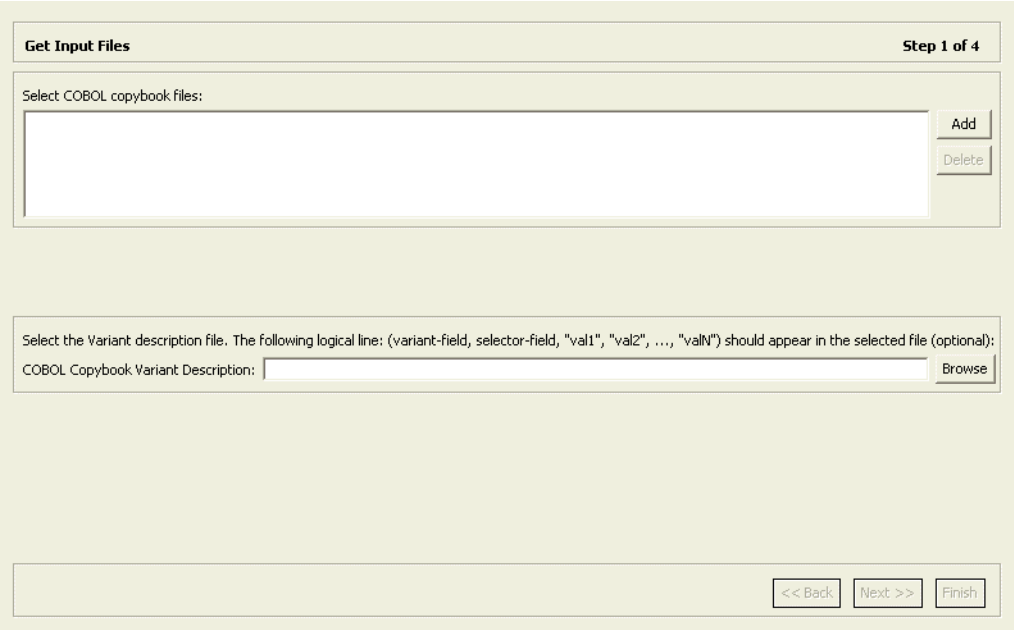
The Select Resource screen is shown in the following figure:

Figure 2–20 The Select Resources screen



4. Right-click the computer and select **Set Transfer Type**. Enter the transfer type (ASCII or BINARY) and click **OK**.
5. Expand the node of the added site and locate the necessary COBOL files. To change the high-level qualifier, right-click the computer and select **Change Root Directory**. Enter a high-level qualifier enclosed in quotes, and click **OK**.
6. Select the file or files and click **Finish**. The selected file or files are displayed in the metadata import wizard.

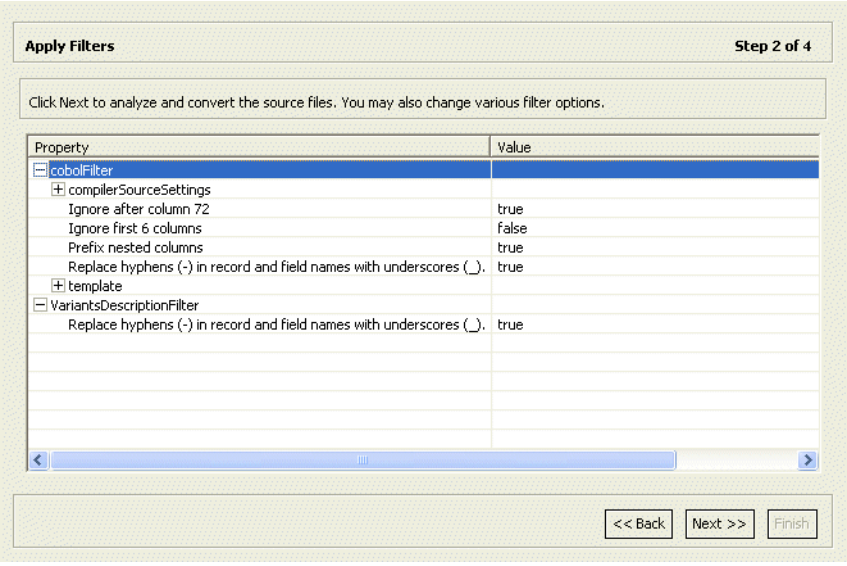
Figure 2–21 The Get Input Files screen



7. Click **Browse**, and navigate to the variant description COBOL file. If required, repeat steps 2-6, described above.
8. Click **Finish**.
9. In the Import wizard, click **Next**.

   The Apply Filters screen is displayed as shown in the following figure:

Figure 2–22 The Apply Filters screen



10. Apply filters to the copybooks. The available filters are listed in the table below. In addition, if required, apply the variant description filter.

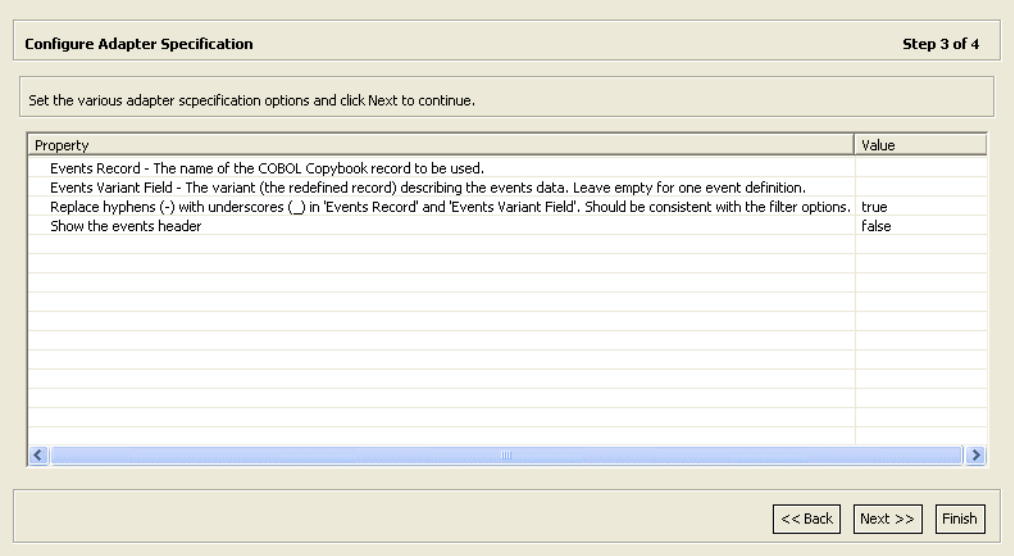
Table 2–6 Variant Description FIlter

Filter	Description
Replace hyphens (-) in record and field names with underscores (_).	When set to `true`, replaces all hyphens in either the record or field names in the metadata generated from the COBOL with underscore characters.

11. Click **Next**.

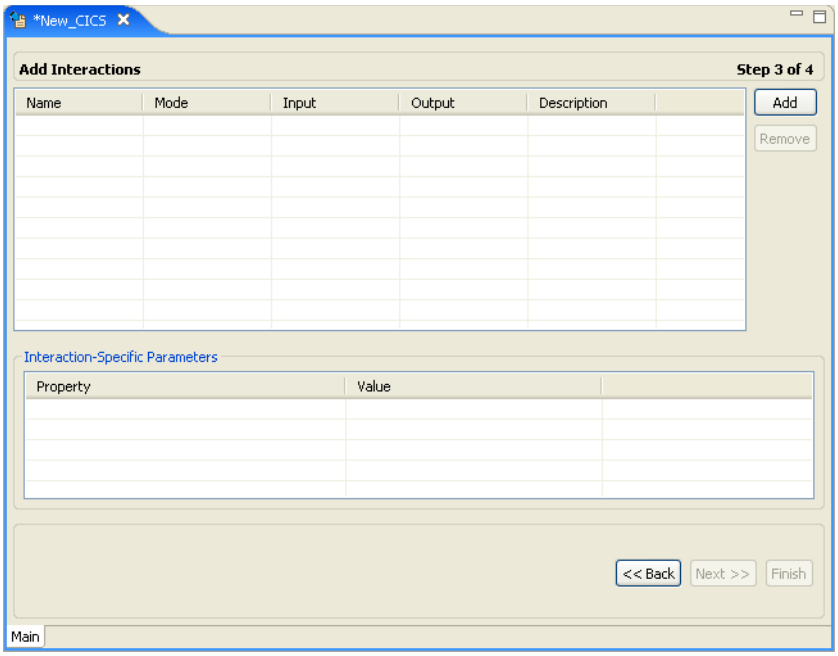
    The Configure Adapter Specification screen is displayed, as shown in the following figure:

Figure 2–23 The Configure Adapter Specification screen



12. In the Value column of the **Events Record** field, enter the name of the COBOL Copybook record that is used.
13. In the Value column of the **Events Variant Field** field, enter the name of the variant (the redefined record) which describes the relevant event's data. This property should be specified only ,if multiple event types are expected. Otherwise leave blank.
14. Specify whether to replace hyphens (`true/false`).
15. Specify whether to show the events header in the metadata (`true/false`).
16. Click **Next** to generate the metadata definitions for the adapter.
17. For **Do you want to transfer the metadata to the server**, click **Yes** to transfer the data to the IBM z/OS computer, and click **Finish**.

Figure 2–24 The Import Metadata screen



2.5 Viewing the XML Schema

The XML schema describing the adapter interactions can be viewed by selecting the **Source** tab when you view the metadata as XML. For more information, see Appendix E, "Editing XML Files in Oracle Studio".

2.6 Creating XML Schemas

The XML schema describing the adapter interactions and the input and output records for these interactions are created automatically during the import procedure, as described in "Generating Outbound Interactions" on page 2-8 and in "Generating Inbound Interactions" on page 2-14.

3

Integrating OracleAS Adapters for CICS with WebLogic

This chapter includes the following topics:

- Overview of Integrating OracleAS Adapters for CICS with WebLogic
- Configuring the CICS Adapter for Outbound
- Configuring the CICS Queue Adapter for Inbound

3.1 Overview of Integrating OracleAS Adapters for CICS with WebLogic

Oracle Application Server provides a complete Java 2 Enterprise Edition (J2EE) environment that executes on the Java Virtual Machine (JVM) of the standard Java Development Kit (JDK). WebLogic is J2EE certified and provides all the J2EE specific containers, APIs, and services. WebLogic supports the J2CA 1.5 standard.

J2CA defines standard Java interfaces for simplifying the integration of applications with the EIS. OracleAS Adapter are deployed as a Resource Adapter (RA) within the WebLogic container.

The contract between the WebLogic client application and the resource adapter is defined by the common client interface (CCI). The contract between the WebLogic container and the resource adapter is defined by the service provider interface (SPI). The SPI API addresses the connection management, transaction management and the security management.

Connection management enables application components to connect to an EIS and leverage any connection pooling provided by the application server.

Transaction management enables an application server to use a transaction manager to manage transactions across multiple resource managers.

Lifecycle management contracts enable an application server to initialize a resource adapter instance during the deployment of the adapter or application server startup. In addition, it enables the application server to notify the resource adapter instance during server shutdown or undeployment of the adapter.

The lifecycle contract provides the mechanism for the application server to manage the lifecycle of the resource adapter instance.

Work management contracts enable the resource adapter to perform its logic by using threads dispatched by an application server, rather than creating threads on its own. The handshake is done through a `Work` instance submission. This makes the application server threads management more efficient, providing better control over their execution contexts (like security and transaction).

Message inflow contracts enable the resource adapter to send asynchronous messages to an application server message endpoints regardless of the messaging style, semantics, or delivering infrastructure. This mechanism enables a wide variety of message providers (like Java Message Service (JMS), or Java API for XML (JAXM)) to be plugged into an application server through the resource adapter.

> **See Also:** Oracle Application Server Adapter Concepts Guide, Oracle Application Server Containers for J2EE User's Guide, Oracle Application Server Containers for J2EE Services Guide, and Oracle Application Server Containers for J2EE Security Guide.

3.2 Configuring the CICS Adapter for Outbound

Deploy the Oracle Connect RAR into the WebLogic server.

You may create the appropriate connection factories for each CICS Adapter.

This table describes the connection factory properties that are relevant for the CICS Adapter. Do not change the values of any of the other properties.

Table 3–1 Connection Factory Properties for Outbound Interactions

Property	Description
`eisName`	Required. Sets the name of the adapter to use. The adapter is defined in Oracle Connect server using Oracle Studio.
`serverName`	Required. Sets the TCP/IP address or host name where the Oracle Connect daemon is running.
`workspace`	Optional. Specifies the name of an Oracle Connect server workspace to use. The default workspace is `Navigator`.
`portNumber`	Optional. Specifies the TCP/IP port where the Oracle Connect daemon is running on the server. The default port is `2551`.
`userName`	Optional. Specifies a user who can access the Oracle Connect server. The user is defined in the Oracle Connect daemon configuration. For more information, see Daemon "Security" on page B-5 and workspace "Security" on page B-16 for details about users allowed to access an Oracle Connect server.
`password`	Optional. Specifies a valid password for the user.
`firewallProtocol`	Optional. Specifies the firewall protocol used: either `none` or `fixedNat` (the Nat protocol using a fixed address for the daemon). The default is set to `none`.
`connectTimeout`	Optional. Specifies the connection timeout in seconds. The default is 0, indicating that there is no connection timeout.
`encryptionProtocol`	Optional. Specifies the name of encryption protocol to use. The default is set to `null`. The RC4 protocol is supported.
`encryptionKeyName`	Optional. Specifies the name of the symmetric encryption key to use.
`encryptionKeyValue`	Optional. Specifies the value of the symmetric encryption key to use.

Table 3–1 (Cont.) Connection Factory Properties for Outbound Interactions

Property	Description
fakeXa	Optional. Set to true or false, according to the value of the transaction support property of the corresponding CICS adapter. When set to true, the XA APIs are internally converted to local transaction APIs. This property is used for outbound interactions only.

3.3 Configuring the CICS Queue Adapter for Inbound

Deploy the Oracle Connect RAR into the WebLogic server.

You may create the appropriate connection factories for each CICS Queue Adapter.

This table describes of the connection factory properties that are relevant for the CICS Queue Adapter. Do not change the values of any of the other properties.

Table 3–2 Connection Factory Properties for Inbound Interactions

Property	Description
eisName	Required. Sets the name of the adapter to use. The adapter is defined in Oracle Connect server using Oracle Studio.
serverName	Required. Sets the TCP/IP address or host name where the Oracle Connect daemon is running.
workspace	Optional. Specifies the name of an Oracle Connect server workspace to use. The default workspace is Navigator.
portNumber	Optional. Specifies the TCP/IP port where the Oracle Connect daemon is running on the server. The default port is 2551.
userName	Optional. Specifies a user who can access the Oracle Connect server. The user is defined in the Oracle Connect daemon configuration. For more information, see Daemon "Security" on page B-5 and Workspace "Security" on page B-16 for details about users allowed to access an Oracle Connect server.
password	Optional. Specifies a valid password for the user.
firewallProtocol	Optional. Specifies the firewall protocol used: either none or fixedNat (the Nat protocol using a fixed address for the daemon). The default is set to none.
connectTimeout	Optional. Specifies the connection timeout in seconds. The default is 0, indicating that there is no connection timeout.
encryptionProtocol	Optional. Specifies the name of encryption protocol to use. The default is set to null. The RC4 protocol is supported.
encryptionKeyName	Optional. Specifies the name of the symmetric encryption key to use.
encryptionKeyValue	Optional. Specifies the value of the symmetric encryption key to use.

3.4 High Availability

The Oracle legacy adapters for OracleAS support High Availability using OracleAS clusters with active-active and active-passive topologies.

In an active-active topology Oracle Application Server provides an active-active redundant model for all its components with OracleAS Clusters. In an OracleAS Cluster, two or more Oracle Application Server instances are configured to serve the same application workload. These instances can reside on the same computer or on different computers. The active instances may be front-ended by an external load balancer, which can redirect requests to any of the active instances, or by some other application-level configuration, such as address lists, to distribute the requests.

In an active-passive topology Oracle Application Server supports Active-passive model using OracleAS Cold Failover Clusters. In this case two or more application server instances are configured to serve the same application workload, but only one is active at any time.

4

Integrating OracleAS Adapters for CICS

To configure the interactions for the OracleAS Adapter for CICS, you must perform the following tasks in the specified order:

1. Configuring OracleAS Adapters for CICS
2. Integrating OracleAS Adapters for CICS with WebLogic
3. Configuring OracleAS Adapter for CICS

See Also: *Oracle Application Server Adapter Concepts Guide*.

4.1 Configuring OracleAS Adapter for CICS

This section includes the following topics:

- Creating JCA Configuration Files
- Configuring the Binding for Outbound Interactions
- Configuring the Binding for Inbound Interactions

4.1.1 Creating JCA Configuration Files

Perform the following steps to create a binding file and a WSDL file for the CICS adapter or CICS Queue, using Oracle Studio:

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. In the Design perspective, Configuration view, expand the **Machine** folder.
3. Expand the machine defined in Setting Up the IBM z/OS Platform in Oracle Studio on page 2-1.
4. Expand the **Bindings**. The binding configurations available on this computer are listed.
5. Expand the **NAV** binding. The NAV binding configuration includes branches for data sources and adapters that are located on the computer.
6. Do *one* of the following:
   - For the CICS adapter, expand the **Adapters** folder and right-click the CICS adapter you defined in Configuring an Oracle Connect Adapter on page 2-6 and select **Generate JCA Configuration Files**.

- For the CICS Queue adapter, expand the **Events** folder and right-click the CICS Queue you defined in Configuring the CICS Queue Adapter on page 2-26 and select Generate **JCA Configuration Files**.

The JCA Configuration Files dialog box opens.

7. In the **Save to** field, type the path to the location where you want to save the files, or select **Browse** to browse to a location.
8. In the **JNDI Location** field, change the default value for the WebLogic Connection Factory JNDI location, if necessary.

   You should ensure that a connection factory exists on the WebLogic server. The Connection Factory is mandatory for Outbound interactions and optional for Inbound interactions.

   For information on the binding file, see Configuring the Binding for Outbound Interactions or Configuring the Binding for Inbound Interactions.
9. Click **OK** to create the files in the correct location.

   The View Genereated Content dialog box opens. It provides links to the binding file (*.`jca`) and the WSDL file that are created. If you want to view or edit the files, click the appropiate link in the dialog box.

   Use these files when working with the BPEL Process Manager or Mediator.

4.1.2 Configuring the Binding for Outbound Interactions

You should not edit the WSDL file generated by Oracle Studio. You may edit the binding file to configure the `ExecutionTimeout` property in the `AttuInteractionSpec`.

> **Note:** The `FunctionName` property must have the same value as the corresponding `operation`.

The following is an example of a binding file for outbound applications:

```
<?xml version="1.0" encoding="UTF-8"?>
<adapter-config adapter="Legacy Adapter" name="calc"
xmlns="http://platform.integration.oracle/blocks/adapter/fw/metadata">
    <connection-factory UIConnectionName="pcbpel" csfKey="pcbpel"
location="eis/legacy/calc"/>

    <endpoint-interaction operation="add" portType="calcPortType">
        <interaction-spec
className="com.attunity.adapter.oracle.AttuInteractionSpec">
            <property name="FunctionName" value="add"/>

            <property name="ExecutionTimeout" value="120"/>
            </interaction-spec>
    </endpoint-interaction>
    <endpoint-interaction operation="display" portType="calcPortType">
        <interaction-spec
className="com.attunity.adapter.oracle.AttuInteractionSpec">
            <property name="FunctionName" value="display"/>

            <property name="ExecutionTimeout" value="120"/>
            </interaction-spec>
    </endpoint-interaction>
```

```
    <endpoint-interaction operation="div" portType="calcPortType">
        <interaction-spec
className="com.attunity.adapter.oracle.AttuInteractionSpec">
            <property name="FunctionName" value="div"/>

            <property name="ExecutionTimeout" value="120"/>
            </interaction-spec>
    </endpoint-interaction>
    <endpoint-interaction operation="mul" portType="calcPortType">
        <interaction-spec
className="com.attunity.adapter.oracle.AttuInteractionSpec">
            <property name="FunctionName" value="mul"/>

            <property name="ExecutionTimeout" value="120"/>
            </interaction-spec>
    </endpoint-interaction>
    <endpoint-interaction operation="sub" portType="calcPortType">
        <interaction-spec
className="com.attunity.adapter.oracle.AttuInteractionSpec">
            <property name="FunctionName" value="sub"/>

            <property name="ExecutionTimeout" value="120"/>
            </interaction-spec>
    </endpoint-interaction>
</adapter-config>
```

4.1.3 Configuring the Binding for Inbound Interactions

You should not edit the WSDL file generated by Oracle Studio. You may edit the binding file to configure the `AttuActivationSpec` properties. Most of these properties can also be provided using the Connection Factory. For information on how to configure the Connection Factory, see Configuring the CICS Queue Adapter for Inbound. For information on the properties you that must be defined in the binding, see Table 4–1, "ActivationSpec Properties Defined in the Binding".

The binding file specifies the name of the adapter's connection factory as the value of the `adapterInstanceJndi` attribute of the `<jca:address>` element in the `<service>` section. If a connection factory exists on the application server, its properties are taken. Otherwise, the properties specified by the `AttuActivationSpec` are used. If a value is specified by both the connection factory and the `AttuActivationSpec`, the `AttuActivationSpec` property overrides the value in the connection factory. If you want to use the value specified in the connection factory, you must delete the property from the binding file.

For a description of these properties, see Configuring the CICS Queue Adapter for Inbound

The following is an example of a binding file for inbound applications:

```
<?xml version="1.0" encoding="UTF-8"?>
<adapter-config adapter="Legacy Adapter" name="eventQ1"
xmlns="http://platform.integration.oracle/blocks/adapter/fw/metadata">
    <resource-adapter
className="com.attunity.adapter.oracle.AttuResourceAdapter"/>
<endpoint-activation operation="eventStream" portType="eventQ1PortType">
<connection-factory UIConnectionName="pcbpel" csfKey="pcbpel"
location="eis/legacy/OracleDisam_CDCQueue"/>
        <activation-spec
className="com.attunity.adapter.oracle.AttuActivationSpec">
            <property name="MessagesInBatch" value="50"/>
```

```
            <property name="WaitTime" value="30"/>

            <property name="EisName" value="eventQ1"/>

            <property name="ServerName" value="localhost"/>

            <property name="PortNumber" value="2551"/>

            <property name="UserName" value=""/>

            <property name="Password" value=""/>

            <property name="Workspace" value="eventQ1"/>

            <property name="RetryInterval" value="15"/>

            <property name="ConnectTimeout" value="0"/>

            <property name="Compression" value="true"/>

            <property name="FirewallProtocol" value=""/>

            <property name="EncryptionProtocol" value=""/>

            <property name="EncryptionKeyName" value=""/>

            <property name="EncryptionKeyValue" value=""/>
            </activation-spec>
    </endpoint-activation>
</adapter-config>
```

The following table describes the properties you must configure in the binding file.

Table 4–1 ActivationSpec Properties Defined in the Binding

Property	Description
`messagesInBatch`	Optional. The maximum number of events that can be sent by the adapter to the SOA process. The default value is `50`.
`support2PC`	Optional. Set to `true` to enable 2PC support. You should set this value to true only if the Oracle adapter is configured for two-phase commit. The default value is `false`.
`waitTime`	Optional. Defines the maximum response time (in seconds) for Oracle Connect to send a response to the CICS queue adapter. The default is set to 30 seconds.
`retryInterval`	Optional. The amount of time the resource adapter waits if no events are found in the server before issuing the next request. The default is set to 15 seconds.

5

Configuring an OracleAS Adapter for CICS for Outbound Interactions

This chapter provides an example for creating an OracleAS adapter for CICS. To work with an OracleAS adapter for CICS, you must also configure either a BPEL process or a Mediator process. For information on using BPEL or Mediator, see Creating the Project in JDeveloper Examples.

5.1 Creating Outbound Interactions with the OracleAS Adapter for CICS

To work with the OracleAS Adapter for CICS and create outbound interactions you must create a CICS adapter, import the metadata, then create interactions that are used to access the data.

5.1.1 Requirements

The following is required to perform the tasks described in this chapter:

- Computer with Windows operating system
- Mainframe computer with z/OS operating system
- Oracle Connect v 12.1.3 installed on the Mainframe computer
- Oracle Studio v 12.1.3 installed on the Windows computer.

5.2 Create the findDoctor and findPatient Interactions

You use a CICS adapter to create the interactions used to get the data that you need. The interactions are used as services when you set up the endpoint in JDeveloper. For more information on using JDeveloper to create an endpoint, see Creating the Project in JDeveloper Examples.

Do the following to create the interactions.

- Prepare the System
- Set up Machine Access to Oracle Connect
- Add a CICS Adapter
- Import the Metadata
- Create the JCA Configuration Files

5.2.1 Prepare the System

Make sure that the Hospital database is available under CICS on the Mainframe computer.

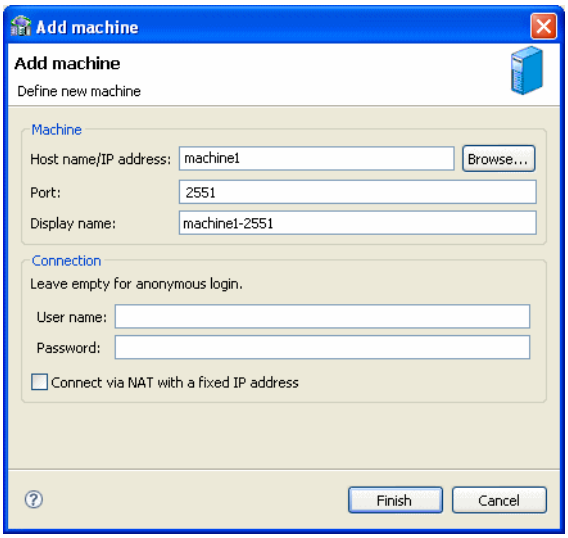
5.2.2 Set up Machine Access to Oracle Connect

You begin by configuring access to the Mainframe computer. You configure this access with Oracle Connect. Open Oracle connect and follow the directions Setting Up the IBM z/OS Platform in Oracle Studio. Enter the following information in the Add machine dialog box:

- **Host name/IP address**: Enter the name or IP address for the Mainframe computer whith the Hospital database. This should be installed in the same directory as Oracle Connect.
- **Port**: Enter the port number where the daemon is running. The default port is 2551.
- **Display name**: You do not need to enter any information in this field (By default, the display name is the host name and the port number).
- **User name**: If the computer you are accessing needs an administrator password. If so enter the name o f the computer's administrator.
- **Password**: If necessary, enter the computer administrator's password.
- **Connect via NAT with fixed IP address**: Select this if the machine uses the NAT (Network Address Translation) firewall protocol, with a fixed configuration, mapping each external IP to one internal IP, regardless of the port specified.

The following figure shows the Add Machine dialog box with the correct information:

Figure 5–1 Add Machine Dialog Box



5.2.3 Add a CICS Adapter

You now create an adapter, which lets you create interactions to get the information you are seeking. To create an adapter, do the following

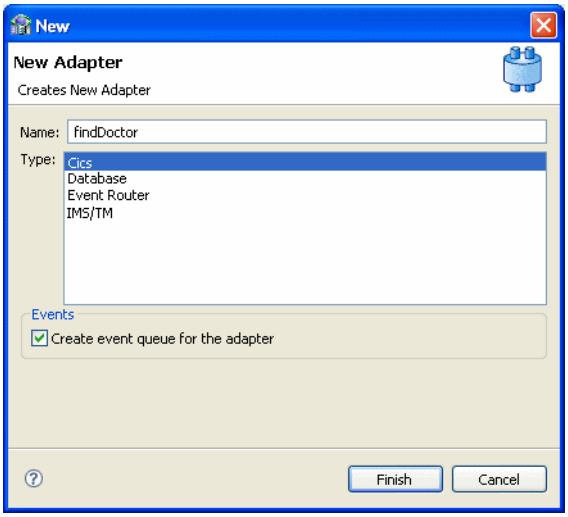
1. Expand the computer you just added and then expand the **NAV** binding.
2. Right-click the **Adapters** folder and then select **New adapter**.

3. Enter the following information for your adapter, then click **Finish**.
   - **Name**: findDoctor
   - **Type**: CICS
4. Click **Finish**.

For a detailed explanation on how to set up a CICS adapter, see Configuring an Oracle Connect Adapter.

The following figure shows the Add adapter dialog box with the correct information:

Figure 5–2 Add Adapter Dialog Box



5.2.4 Import the Metadata

You now import the metadata from the CICS data into the adapter. For information on how to import data, see Generating Outbound Interactions.

In this example you should import the metadata for the `Doctor` and `Patient` columns of the Hospital database. This database is supplied as an example with Oracle Connect and should be installed when you install Oracle Connect on your Mainframe computer.

You need the following information for the metadata import:

- For the **Import name**, enter **findDoctor**.
- In the **Get Input Files** step, add the file, `hospital.cob`.
- In the **Apply Filters** step, just click **Next**.
- In the **Add interactions** step, under the **Name** column, enter findDoctor, under the **Input** column, select DOCTOR and under the **Output** column, select DOCTOR.
- In the **Import Metadata** step, select **Yes** and then click **Finish**.

5.2.5 Create the JCA Configuration Files

You must create both a WSDL and JCA file to set up your connection with JDeveloper. You create the files using Oracle Studio before setting up the BPEL process or Mediator connections.

For information on how to create JCA configuration files, see Creating JCA Configuration Files. To create the files for the `findDoctor` adapter, do the following:

1. From the Cofiguration pane in Oracle Studio, expand the NAV binding and the Adaper folder.
2. Right-click the findDoctor adapter and select Generate JCA Configuration Files.
3. In the dialog box, save the files to a place you can find when you create the BPEL process or Mediator connection.

6

Creating the Project in JDeveloper Examples

This chapter contains the following topics:

- Getting Started
- Deploy a BPEL Outbound Process
- Deploy a Mediator Process

6.1 Getting Started

This section describes the components necessary to work with the samples and how to prepare your system. It has the following topics:

- Prerequisites
- Beginning Tasks

6.1.1 Prerequisites

The following are installation and configuration requirements for using the examples.

- Personal computer running Microsoft Windows XP, Windows Vista, Windows Server 2003 or Windows Server 2008 with one GB of RAM.
- The CICS adapter that you are using must be deployed to the Oracle Application Server.
- OracleAS CICS Adapter must be configured for outbound processing. See Configuring an OracleAS Adapter for CICS for Outbound Interactions.
- Oracle JDeveloper must be installed.
- Legacy adapter must be deployed.
- Oracle Studio with the necessary machines loaded. See Setting Up the IBM z/OS Platform in Oracle Studio.

This chapter provides examples of the configuration steps that are used when using BPEL to work with the Oracle CICS adapters. You should be familiar with the following before working with these examples:

- How to create J2CA connections. See Configuring the CICS Adapter for Outbound for more information.
- How to configure the OracleAS adapters for CICS using Oracle Studio. See Configuring an Oracle Connect Adapter for more information.

Integration between the OracleAS adapters for CICS and BPEL has the following processes:

- **Design Time**: The OracleAS adapters for CICS are configured with Oracle Studio as described in Configuring an Oracle Connect Adapter.
- **Run time**: After you deploy the BPEL process with JDeveloper, you can test your configurations. See Run-time Configuration.

6.1.2 Beginning Tasks

Before you design the outbound process, you must perform the following:

- Perform the tasks in Configuring an OracleAS Adapter for CICS for Outbound Interactions to create the CICS adapter, import the metadada, and create the interactions.
- Configure the connection for the J2CA CICS resource adapter. You do this by creating a connection factory.

  For more information, see Configuring the CICS Adapter for Outbound.
- Create the WSDL and binding files for the interaction. You do this in Oracle Studio before you begin to build your endpoint in JDeveloper. Oracle Studio creates the WSDL and binding (jca) files automatically, based on the interaction you create. See Create the JCA Configuration Files for information on creating these files.

6.2 Deploy a BPEL Outbound Process

This section describes how to create a BPEL outbound process that connects to a OracleAS adapter for CICS.

To configure a process with the BPEL process manager, you use JDeveloper. This section describes how to create a BPEL outbound process for CICS reource adapters using JDeveloper. The following sections describe how to deploy a BPEL outbound process.

- Design-Time Tasks
- Run-time Configuration

Before you design the outbound process, integrate and connect your OracleAS adapter for CICS with the BPEL process manager.

6.2.1 Design-Time Tasks

This section describes the design-time steps necessary to deploy a BPEL outbound process. The processes use the Web service called FINDDOCTOR. This service finds a list of doctors from the data. The following are the steps required to create the outbound process.

- Create a BPEL Project for a BPEL Outbound Process
- Use a Third-Party Adapter to Read the Doctor Data
- Design a BPEL Outbound Process
- Edit the Invoke_1 Activity
- Deploy the Project

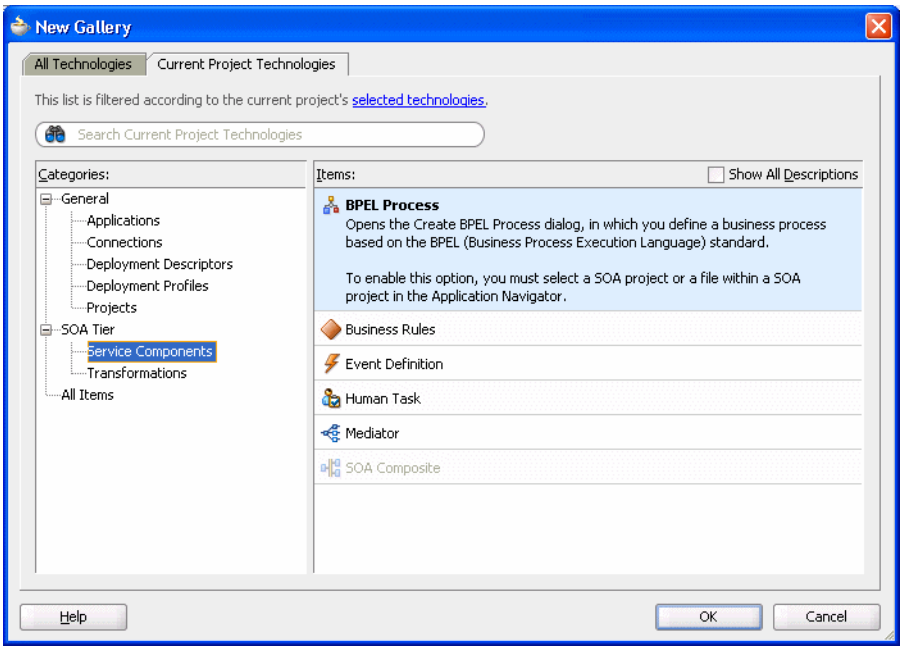
6.2.1.1 Create a BPEL Project for a BPEL Outbound Process

Perform the following to create a BPEL project for outbound.

1. In JDeveloper, create a new SOA project.

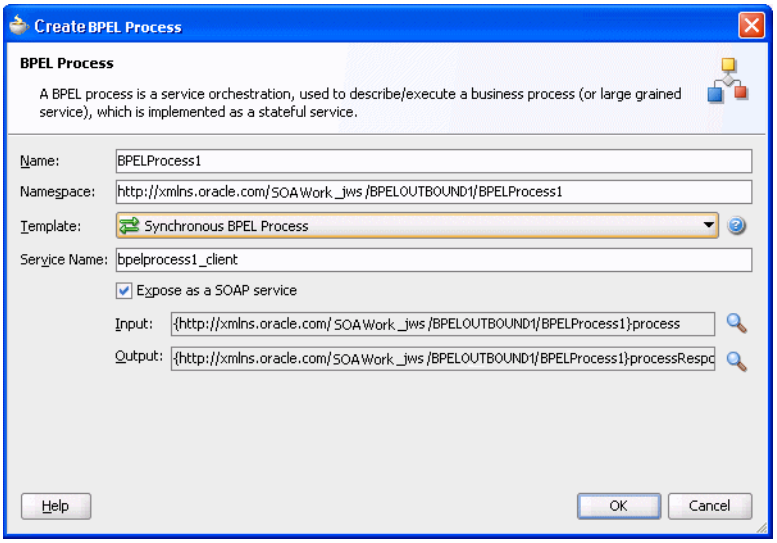
2. Rright-click the SOA project and select **New**. The New Gallery is displayed as shown in Figure 6–1.

Figure 6–1 New Gallery



3. In the **Categories** list, expand **SOA Tier** and select **Service Components**.
4. From the **Items** list, select **BPEL Process** and click **OK**. The BPEL Process screen is displayed as shown in Figure 6–2.

Figure 6–2 BPEL Process



5. Enter the following in the BPEL Process screen:
   - Enter a **Name**, for example **BPELOUTBOUND1**.
   - Enter the **Directory** where you want to save the project.

- Select a Synchronous BPEL Process **Template**.

6. Click **OK**. The BPEL Process is dislpayed in the visual editor.

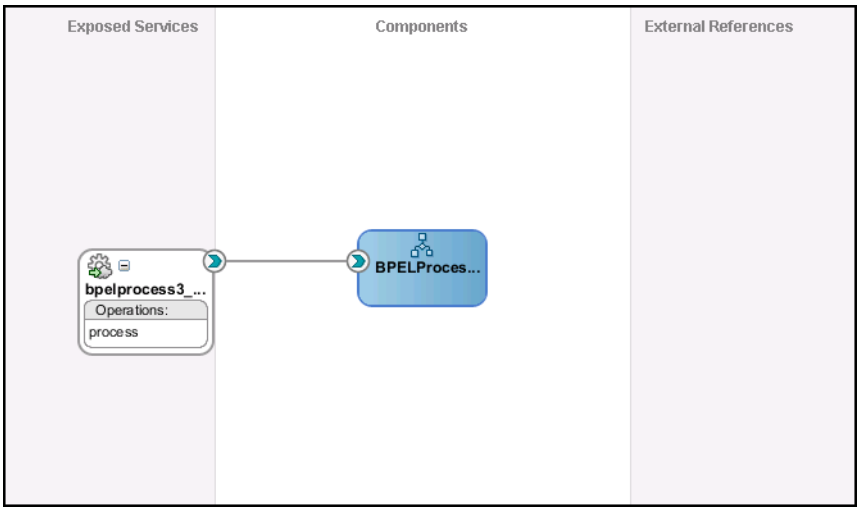
   Next you Use a Third-Party Adapter to Read the Doctor Data.

6.2.1.2 Use a Third-Party Adapter to Read the Doctor Data

You must use a third-party adapter to read the DOCTOR data. SOA executes the SQL statment defined in the interaction and passes it to a resource adapter. The resource adapter then processes the information and returns the data. Perform the following to create the third-party adapter.

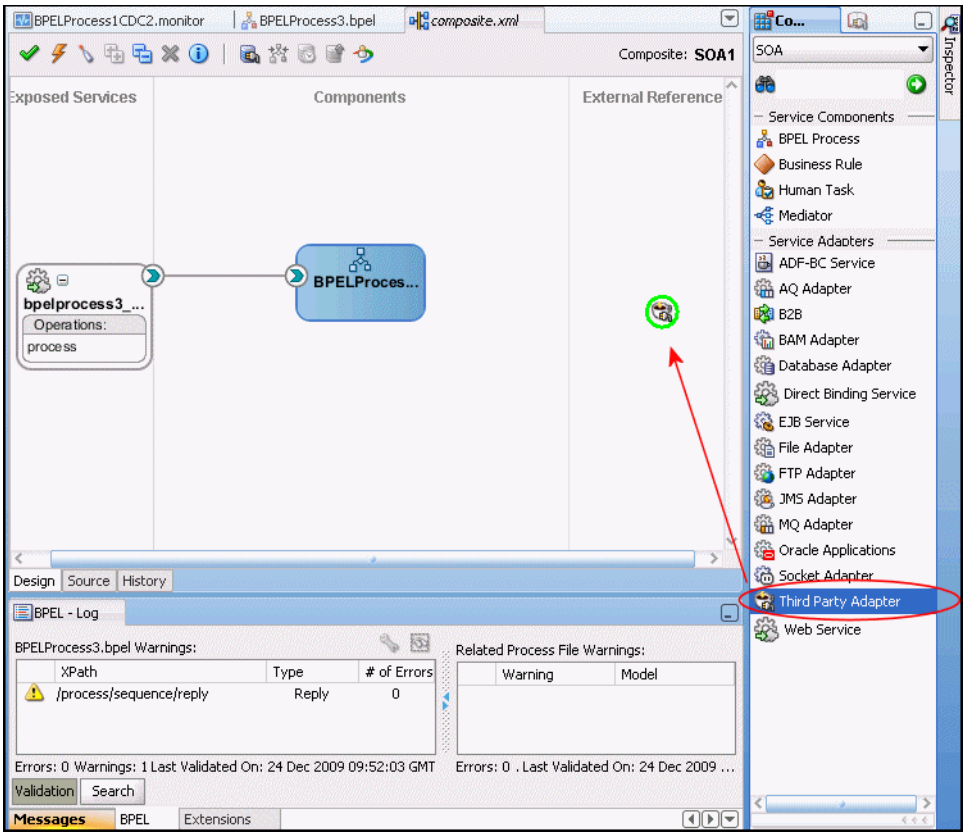
1. From the **Projects** area, on the left side, double-click the `composite.xml`. At the bottom of the visual editor, click the **Design** tab, if necessary. Figure 6–3 shows the Composite View in the visual editor:

Figure 6–3 Composite View



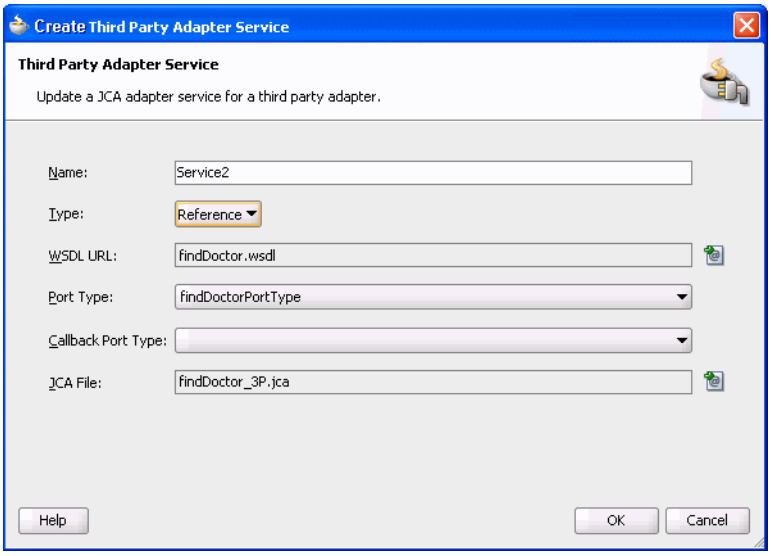
2. Drag a third-party adapter into **External References** lane of the visual editor, as shown in Figure 6–4.

Figure 6–4 Third-Party Adapter



When you drag the third-pary adapter into JDeveloper the Create Third Party Adapter Service screen is displayed as shown in figure Figure 6–5.

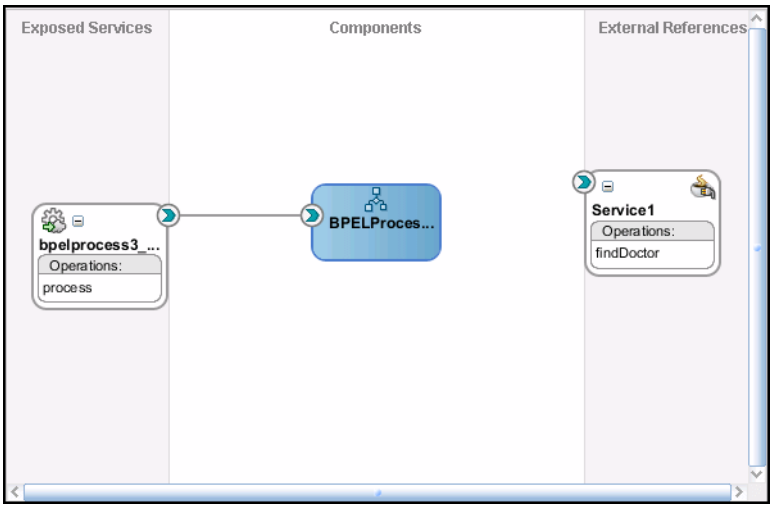
Figure 6–5 Create Third-Party Adapter Service



3. Click the button next to the **WSDL URL** field and browse to find the `findDoctor.wsdl` file that you created in Create the JCA Configuration Files.
4. In the **Type** list, select **Reference**.

5. Click the button next to the **JCA File** field and browse to find the `findDoctor.jca` file that you created in Create the JCA Configuration Files.
6. Click **OK** to close the dialog box. You should now have a composite design as shown in Figure 6–6.

Figure 6–6 Ready to Create BPEL Process (Composite View)



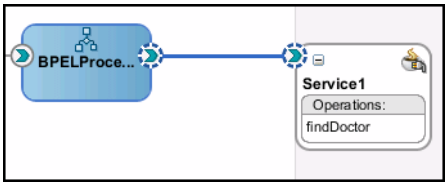
Continue with the Design a BPEL Outbound Process step.

6.2.1.3 Design a BPEL Outbound Process

In this step, you design the BPEL process. Do the following to design the BPEL outbound process:

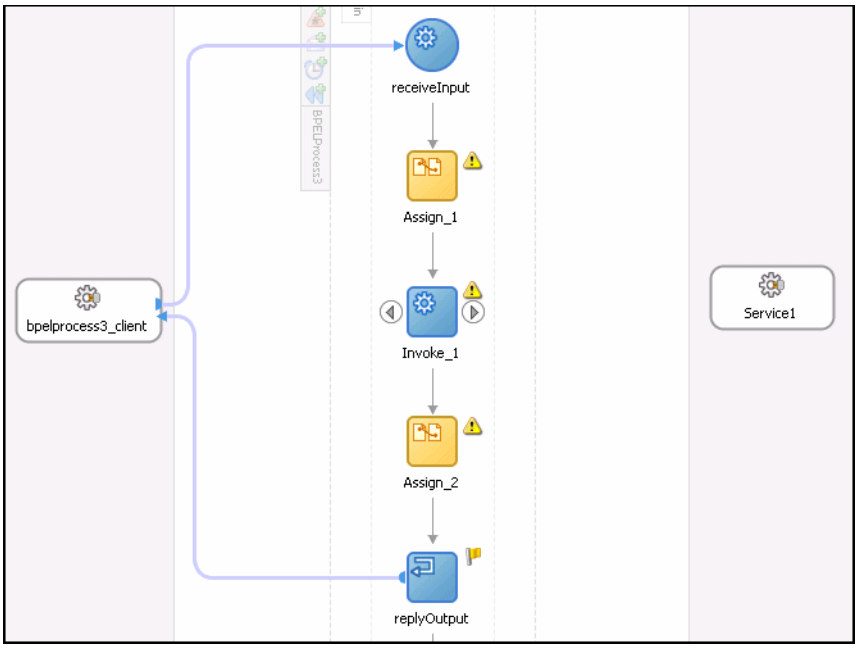
1. In the Composite, connect the BPEL process to the third-pary adapter as shown in Figure 6–7.

Figure 6–7 Connect to Third-Party Adapter



2. Double-click the BPEL process.
3. From the Component Palette, BPEL Process Activities section, drag an Assign, Invoke, and another Invoke activity into the editor for the process as shown in Figure 6–8.

Figure 6–8 Outbound Process



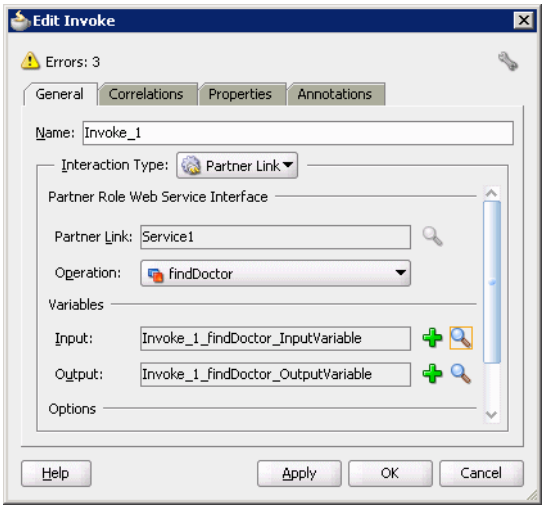
4. Continue with the Edit the Invoke_1 Activity.

6.2.1.4 Edit the Invoke_1 Activity

You must now edit the Invoke activity so that it invokes the `findDoctor` interaction.

1. In the Visual Editor, connect the `Invoke_1` activity to the third-party adapter. The Edit Invoke screen is displayed s shown in Figure 6–9.

Figure 6–9 Edit Invoke



2. In the Invoke dialog box, ensure that the Partner Role Web Service Interface section has the following values:
   - **Operation**: `FINDDOCTOR`
3. Click the plus sign next to the **Input** field and select the `Invoke_1_findDoctor_InputVariable`.

4. Click the plus sign next to the Output field and select the `Invooke_1_findDoctor_OutputVariable`.
5. Click **OK** then save then Deploy the Project.

6.2.1.5 Deploy the Project

Perform the following to deploy the project to the Oracle Application Server:

1. From the Application Navigator, right-click the project and find **Deploy** and then select the project you are deploying.
2. From the Deploy wizard, select **Deploy to Application Server.**
3. Continue to click **Next** till the end of the wizard and then click **Finish** to deploy.

6.2.2 Run-time Configuration

This section describes how to test the project and run it to see that the outbound process is executed successfully. Do the following to test the outbound process.

1. Create a file and call it `FindDoctor.xml` as shown in the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<CU:FindDoctorData
  xmlns:CU="http://xmlns.oracle.com/Esb/FindDoctorData"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<Record1>CICS</Record1>
</CU:FindDoctorData>
```

2. Save the file to the `C:/temp` folder that was defined for the first adapter. The file is deleted after the SOA server reads the file and completes the process.
3. Open the `C:\tempOut` folder that was defined in the second adapter. A file with the following contents is copied to this folder.

```
<?xml version="1.0"?>
<findDoctorData xmlns="http://xmlns.oracle.com/Esb/findDoctorData">
  <Record1 xmlns>="">CICS</Record1>
```

6.3 Deploy a Mediator Process

This section describes how to create a Mediator outbound process that connects to a OracleAS adapter for CICS.

You use JDeveloper. to configure a process with Mediator This section describes how to create a Mediator outbound process for CICS reource adapters using JDeveloper. The following sections describe how to deploy a BPEL outbound process.

- Design-Time Tasks

6.3.1 Design-Time Tasks

This section describes the design-time steps necessary to deploy a Mediator outbound process. The processes use the Web service called FINDDOCTOR. This service finds a list of doctors from the data. The following are the steps required to create the outbound process.

- Create a Mediator Project for a Mediator Outbound Process
- Use a Third-Party Adapter to Read the Doctor Data
- Design an Outbound Mediator

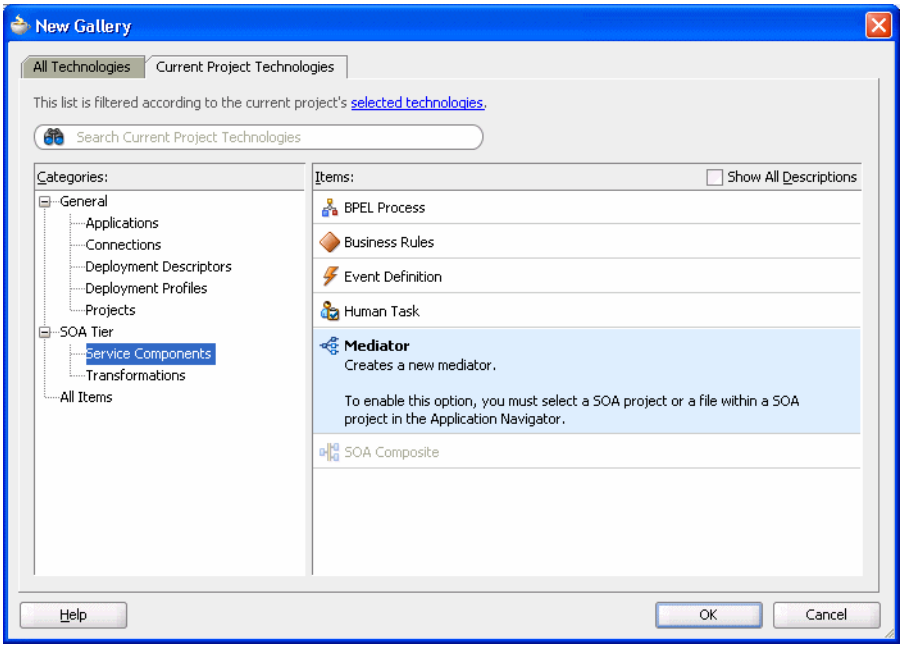
- Create Mapping Rules and Map the Files
- Deploy the Project

6.3.1.1 Create a Mediator Project for a Mediator Outbound Process

Do the following to create a Mediator project for outbound.

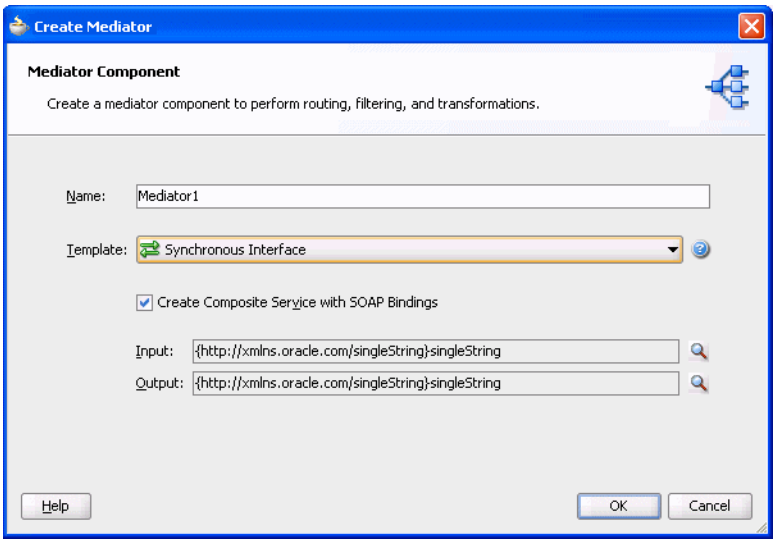
1. In JDeveloper, create a new SOA project.
2. Right-click the SOA project and select **New**. The New Gallery is displayed as shown in Figure 6–10.

Figure 6–10 New Gallery



3. In the **Categories** list, expand **SOA Tier** and select **Service Components**.
4. From the **Items** list, select **Mediator** and click **OK**. The Creator Mediator screen is displayed as shown in Figure 6–11.

Figure 6–11 Mediator



5. Enter the following in the Create Mediator screen:
   - Enter a **Name**, for example **MEDOUTBOUND1**.
   - Select a Synchronous Mediator **Template**.
6. Click **OK**. The Mediator is dislpayed in the visual editor.

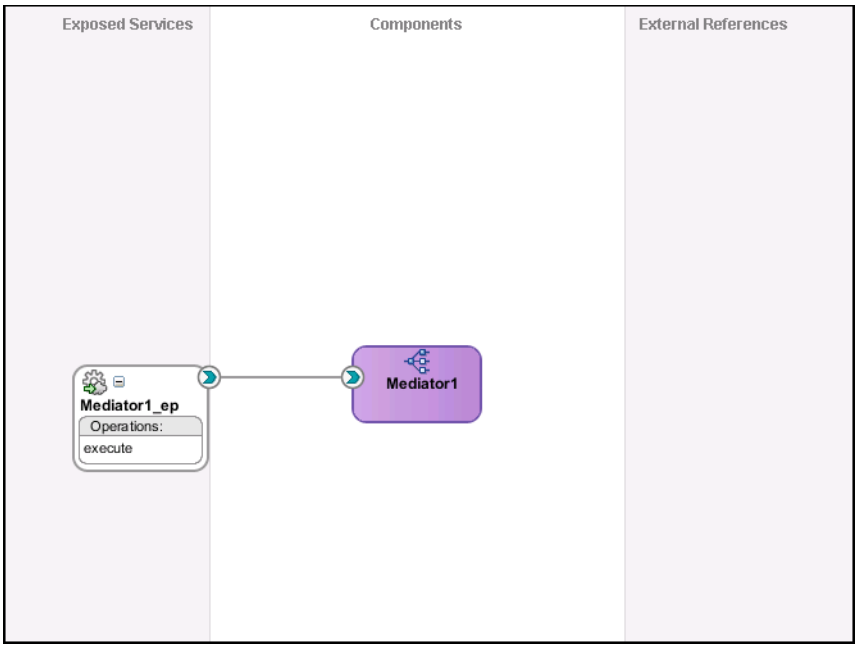
   Continue with Use a Third-Party Adapter to Read the Doctor Data.

6.3.1.2 Use a Third-Party Adapter to Read the Doctor Data

You must use a third-party adapter to read the DOCTOR data. SOA exectes the SQL statment defined in the interaction and passes it to a resource adapter. The resource adapter then processes the information and returns the data. Do the following to create the third-party adapter.

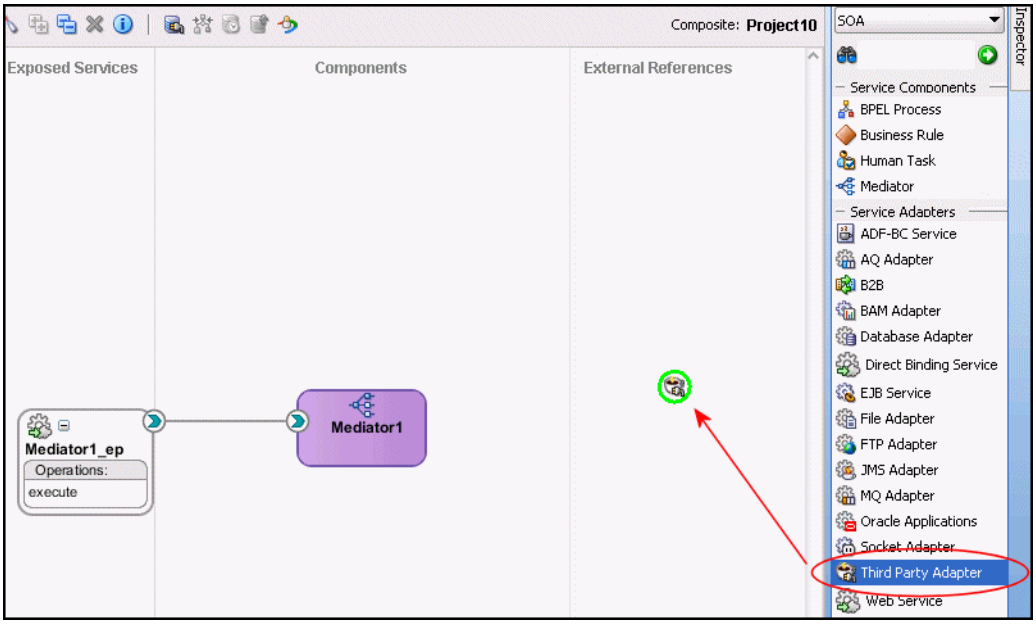
1. From the **Projects** area, on the left side, double-click the `composite.xml`. At the bottom of the visual editor, click the **Design** tab, if necessary. Figure 6–12 shows the Composite View in the visual editor:

Figure 6–12 Composite View



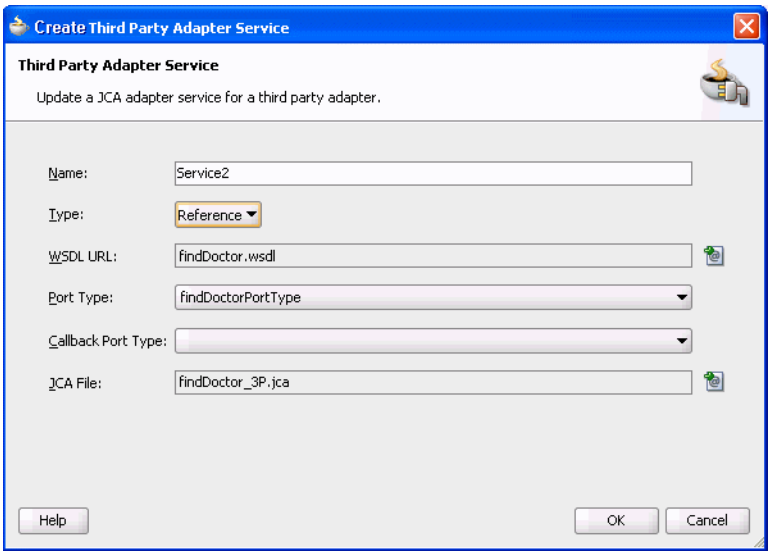
2. Drag a third-party adapter into **External References** lane of the visual editor, as shown in Figure 6–4.

Figure 6–13 Third-Party Adapter



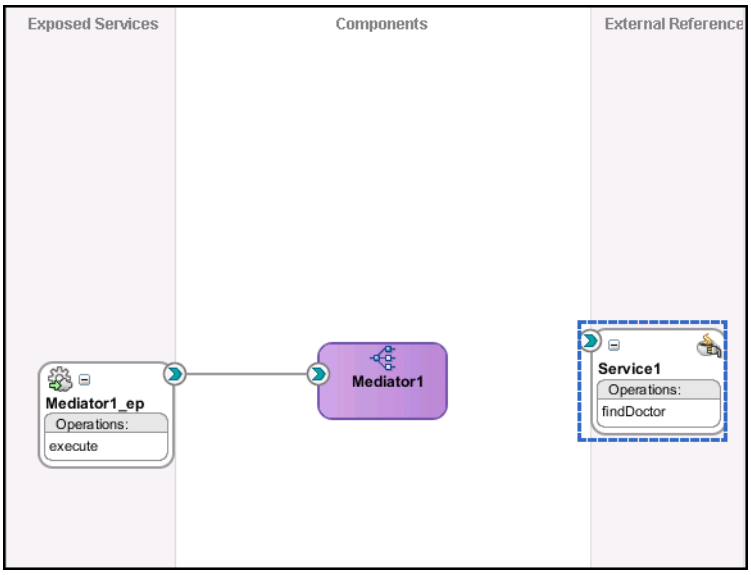
When you drag the third-pary adapter into JDeveloper the Create Third Party Adapter Service screen is displayed as shown in figure Figure 6–14.

Figure 6–14 Create Third-Party Adapter Service



3. Click the button next to the **WSDL URL** field and browse to find the `findDoctor.wsdl` file that you created in Create the JCA Configuration Files.
4. In the **Type** list, select **Reference**.
5. Click the button next to the **JCA File** field and browse to find the `findDoctor.jca` file that you created in Create the JCA Configuration Files.
6. Click **OK** to close the dialog box. You should now have a composite design as shown in Figure 6–15.

Figure 6–15 Ready to Create Mediator (Composite View)



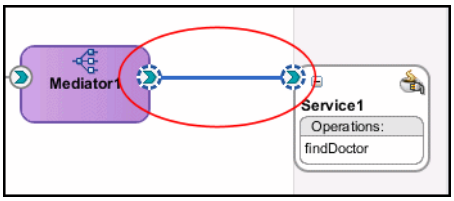
Continue with the Design an Outbound Mediator step.

6.3.1.3 Design an Outbound Mediator

In this step, you design the outbound Mediator. Perform the following to design the Mediator outbound process:

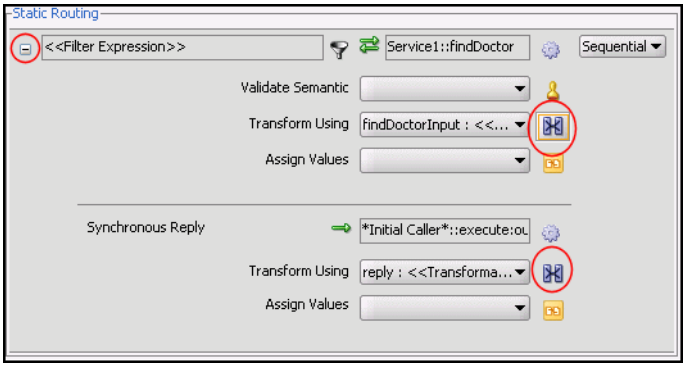
1. In the Composite, connect the Mediator to the third-party adapter as shown in Figure 6–16.

Figure 6–16 Connect to Third-Party Adapter



2. Double-click the Mediator process.
3. Expand the Filter Expression as shown in Figure 6–17.

Figure 6–17 Filter Expression



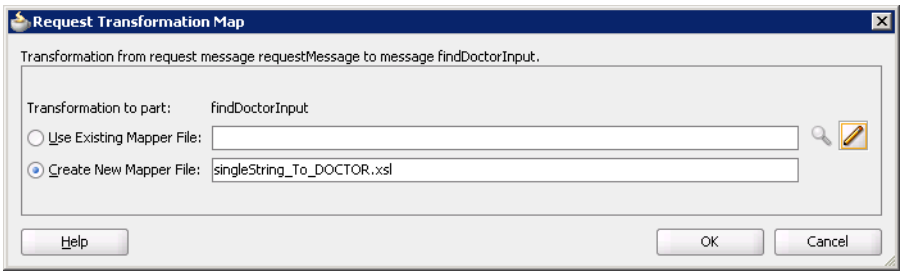
4. Continue with Create Mapping Rules and Map the Files.

6.3.1.4 Create Mapping Rules and Map the Files

You must perform the following steps to create mapping files and map the files.

1. In the upper section of the Filter Expression screen, click the icon next to the **Transform using** field as shown in Figure 6–17. The Request Transformation Map screen is displayed as shown in Figure 6–18.

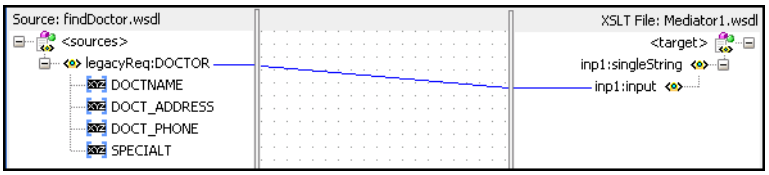
Figure 6–18 Request Transformation Map



2. In the Request Transformation Map, select **Create New Mapper** file. Use the default value, which should be `singleString_To_Doctor.xsl`.
3. Click **OK**.
4. In the visual editor, click to return to the **mediator.mplan** tab.

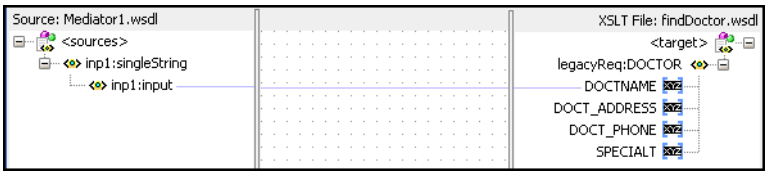
5. In the bottom section of the Filter Expression screen, click the icon next to the Transform using field as shown in Figure 6–17. The Request Transformation Map screen is displayed.
6. In the Request Transformation Map, select **Create New Mapper** file. Use the default value, which should be `Doctor_To_singleString`.xsl.
7. Click the `singleString_To_Doctor` tab. Connect the `Doctor` element fromthe `source:findDoctor.wsdl` to the `inp1:input` element in the `Mediatorx.wsdl` as shown in Figure 6–19.

Figure 6–19 Map Doctor to Single String



8. Click the `Doctor_To_singleString` tab. Connect the `inp1:input` element fromthe `source:Mediatorx.wsdl` to the `DOCTOR` attribute in the `findDoctor.wsdl` as shown in Figure 6–20.

Figure 6–20 Map Single String to Doctor



9. Save and then Deploy the Project.

6.3.1.5 Deploy the Project

Perform the following to deploy the project to the Oracle Application Server:

1. From the Application Navigator, right-click the project and find **Deploy** and then select the project you are deploying.
2. From the Deploy wizard, select **Deploy to Application Server.**
3. Continue to click **Next** till the end of the wizard and then click **Finish** to deploy.

7

Troubleshooting OracleAS Adapter for CICS

Troubleshooting OracleAS Adaptesr for CICS involve checking various definitions and properties in Oracle Connect, including daemon status, workspace options, server parameters, and various system logs.

This chapter contains the following topics:

- Troubleshooting the Daemon
- Resolving Communication Errors
- Resolving Specific Errors

7.1 Troubleshooting the Daemon

Troubleshooting the daemon and the communication between Oracle Application Server and OracleAS Adapter for CICS is performed using Oracle Studio. It is used to monitor the daemon and server activity and control what happens to the daemon and server processes.

> **See Also:** "Advanced Tuning of the Daemon" for details about the configuration settings.

This section contains the following topics:

- Starting the Daemon
- Shutting Down the Daemon
- Monitoring the Daemon During Run time
- Daemon Logs

7.1.1 Starting the Daemon

The daemon is started when OracleAS Adapter for CICS is installed. In case you have shut down the daemon, as described in "Shutting Down the Daemon" on page 7-2, you can restart the daemon as described in the following task.

> **Note:** The daemon is started on the IBM z/OS platform. It cannot be started remotely using Oracle Studio.

7.1.1.1 Task: Starting the Daemon

Activate *INSTROOT*.USERLIB(ATTDAEMN) as a started task to invoke the daemon. For example, in the SDSF screen enter the following command:

```
'/s ATTDAEMN'
```

Where INSTROOT is the high-level qualifier where Oracle Connect is installed.

> **See Also:** *Oracle Application Server Legacy Adapters Installation Guide* for details about the ATTDAEMN JCL.

7.1.2 Shutting Down the Daemon

To shut down the daemon use Oracle Studio, as follows:

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. Select the computer defined in "Setting Up the IBM z/OS Platform in Oracle Studio".
3. Right-click the computer and select **Open Runtime Perspective**.
4. In the Runtime Explorer, right-click the computer and select **Shutdown Daemon**.

7.1.3 Monitoring the Daemon During Run time

Use the Runtime Manager perspective of Oracle Studio to monitor the daemon during run time.

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. Right-click the computer defined in "Setting Up the IBM z/OS Platform in Oracle Studio" in the Configuration Explorer and select **Open Runtime Perspective**.

You can manage the daemon by expanding the relevant node, daemon, workspace or server process, and choosing the relevant option, as described in the following sections.

7.1.3.1 Daemon (Computer) Options

Right-click the daemon to display the options available for it, including the ability to display the daemon log.

The following options are available at the daemon level:

Table 7–1 Daemon Options

Option	Description
Edit Daemon Configuration	Opens the daemon editor, enabling you to reconfigure the daemon. For more information, see "Advanced Tuning of the Daemon" for details about the configuration settings.
Status	Checks the status of the daemon. The information about the daemon includes the daemon name, configuration used, the active client sessions, and logging information.
Reload Configuration	Reloads the configuration after any changes. Any servers currently started are not affected by the changed configuration. For more information, see "Advanced Tuning of the Daemon" for details about the configuration settings.
View Log	Displays the daemon log. For more information, see "Daemon Logs" on page 7-4.
View Events	DIsplays the daemon events log.

Table 7–1 (Cont.) Daemon Options

Option	Description
Daemon Properties	Displays information about the computer where the daemon is running, such as the physical address and any username and password needed to access the computer.
Shutdown Daemon	Shuts down the daemon on the computer.
Recycle servers	Closes all unused servers and prepares all active servers to close when the client disconnects. New connection requests are allocated with new servers.
Kill servers	Immediately closes all active and unused servers. **Note**: It is recommended to use this option with caution, as it may lead to data loss.
Rename	Enables to change the name of the daemon displayed in the Runtime Explorer.
Remove	Removes the daemon from the Runtime Explorer.
Refresh	Refreshes the display.

7.1.3.2 Workspace Options

Right-click a workspace to display the options available for the workspace, including the ability to display the workspace log.

The following table lists the available options:

Table 7–2 Workspace Options

Option	Description
Edit Workspace Configuration	Opens the daemon editor to enable you to reconfigure the workspace. For more information, see "Advanced Tuning of the Daemon" for details about the configuration settings.
Status	Checks the status of the workspace, whether it is available or not.
View Log	Displays the log for all servers for the workspace. For more information, see "Daemon Logs" on page 7-4.
View Events	Displays the workspace events log.
Recycle Servers	Closes all unused servers and prepares all active servers to close when the client disconnects. New connection requests are allocated with new servers.
Kill Servers	Immediately closes all active and unused servers. **Note**: Use this option with caution, as it may lead to data loss.
Refresh	Refreshes the display.

7.1.3.3 Server Options

Right-click a server to display the options available for the server, including the ability to display the server log.

The options available for the server level are listed in the following table:

Table 7–3 Server Options

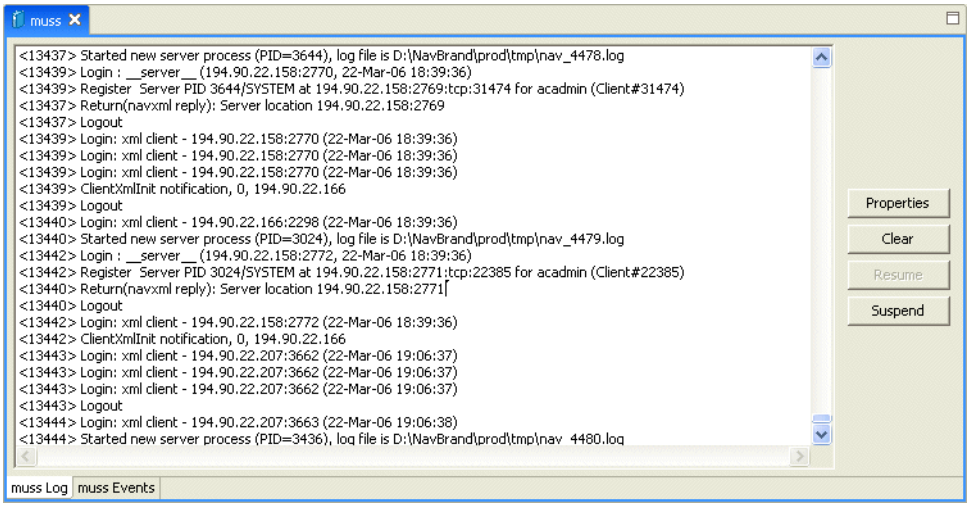
Option	Description
Status	Checks the status of the server. The information about the server includes the server mode and the number of active client sessions for the server.
View Log	Displays the server log. For more information, see "Daemon Logs" on page 7-4.
View Events	This option is for future use.
Kill server	Ends the server process, regardless of its activity status. **Note**: Use this option with caution, as it may lead to data loss.
Refresh	Refreshes the display.

7.1.4 Daemon Logs

Oracle Connect produces several logs that you can use to troubleshoot problems. The daemon manages the following logs:

- Daemon log
- Workspace log
- Server process log

The Runtime Manager perspective of Oracle Studio provides a monitor for these logs, as shown in the following figure:



Display the required log by right-clicking the level you want (daemon, workspace or server) and selecting **View Log**. Each log is displayed in a separate tab. You can flick between logs by clicking the required tab.

7.1.4.1 The Daemon Log Monitor

The daemon log displays activity between clients and the daemon, including clients logging in and logging out from the daemon.

You can change the level of logging by clicking **Properties**. The following levels of logging are available:

- `none`: The log displays who has logged in and out from the daemon.

- `error`: The log displays who has logged in and out from the daemon and any errors that have been generated.
- `debug`: The log displays who has logged in and out from the daemon, any errors that have been generated and, any tracing that has been specified in the daemon configuration.

See Also: Daemon "Logging" on page B-3 for details.

7.1.4.2 The Workspace Log Monitor

The workspace log displays information about the workspace being used by the client.

You can change the level of logging by clicking **Properties**. The following levels of logging are available:

- `none`: The log displays who has connected and disconnected from the server process.
- `error`: The log displays who has connected and disconnected from the server process and any errors that have been generated.
- `debug`: The log displays who has connected and disconnected from the server process, any errors that have been generated, and any tracing that has been specified in the daemon configuration.

See Also: Workspace "General" on page B-7.

7.1.4.3 The Server Log Monitor

The server log displays activity between clients and the server process used by that client to handle the client request.

You can change the level of logging by clicking **Properties**. The following levels of logging are available:

- `none`: The log displays who has connected and disconnected from the server process.
- `error`: The log displays who has connected and disconnected from the server process and any errors that have been generated.
- `debug`: The log displays who has connected and disconnected from the server process, any errors that have been generated, and any tracing that has been specified in the daemon configuration.

See Also: Workspace "General" on page B-7.

7.2 Resolving Communication Errors

When Oracle Studio disconnects from the IBM z/OS computer, the computer is displayed in Oracle Studio with an X in a red circle. If this situation occurs, try to access the computer later.

The following table describes the various scenarios that may exist when Oracle Application Server disconnects from the IBM z/OS computer.

Table 7–4 Scenarios When a Client Is Disconnected

Scenario	Idle (Not Processing a Client Request)	Processing a Client Request
Explicit Disconnect (client explicitly closes connection or client program terminates)	The server is immediately notified of the disconnect and either becomes available for use by another client or terminates (if it is not reusable).	The server does not know that the client has disconnected and continues processing. When processing completes, the server tries to reply to the client and immediately gets an error that the connection was lost. The server either becomes available for use by another client or terminates (if it is not reusable).
Abrupt Disconnect (client closed without proper shutdown or client system hanged and communication disconnected)	The server does not know that the client has disconnected and remains in the idle state. After timing out based on whichever comes first of the value for the client idle timeout daemon workspace parameter or the TCP/IP KEEPALIVE parameter, the server is notified of the disconnect and either becomes available for use by another client or terminates (if it is not reusable).	The server does not know that the client has disconnected and continues processing. When processing completes, the server tries to reply to the client. After an interval (typically several minutes, depending on the TCP/IP configuration), during which the TCP/IP subsystem retries sending the message to the client, the server assumes that the client has terminated and notifies the server that the connection has been closed. The server either becomes available for use by another client or terminates (if it is not reusable).

To troubleshoot client/server communication problems, you must be familiar with the following:

- Daemon configuration settings.
- Oracle Connect security.
- TCP/IP subsystem. Oracle Application Server Adapter for CICS uses TPC/IP for internal intermachine communications.
- System details, such as the account name and password of the administrator account, the IP address of the computers involved and whether a portmapper is being used.

7.3 Resolving Specific Errors

The following error messages relate to errors received from Oracle Connect.

C007: Server initialization failed.

Cause: The daemon failed to start its network service.

Action: Check the processes being run on the system to see whether another daemon or program is using the port specified in the oc4j-ra-xml file for the adapter.

Action: Check the TCP/IP subsystem on the current computer by trying to ping it or run FTP or Telnet to or from it.

Action: Check whether the daemon has privileges to use the TCP/IP services on the current computer with the port specified in the oc4j-ra-xml file for the adapter.

C008: Setting server event handler failed.

Cause: Internal error.

Action: Contact Oracle Support Services.

C009: IRPCD process has been terminated by user request.

Cause: This message is informational only. The daemon successfully shut down.

Action: No action required.

C00A: Application %s not found.

Cause: The requested workspace does not exist.

Action: Check that the workspace defined in the oc4j-ra-xml file is also defined in the daemon configuration on the IBM z/OS platform. Use the **Status** option in the Runtime Manager perspective.

C00B: Invalid IRPCD client context.

Cause: A non-Oracle Connect program is trying to connect to the daemon.

Action: Check the processes and end the relevant process with a system command.

C00C: Daemon request requires a server login.

Cause: A non-Oracle Connect server or program was trying to use a daemon service reserved for Oracle Connect servers.

Action: Check the processes and end the relevant process with a system command.

C00D: Daemon request requires a client login.

Cause: The requested daemon requires a valid client login, which was not supplied.

Action: Reissue the command and specify a user name and password.

Action: Edit the user profile in Oracle Studio to specify a valid user name and password for the IBM z/OS platform.

See Also: "Setting Up Run-Time User Access to the IBM z/OS Platform" on page 2-4.

C00E: Daemon request requires an administrator login.

Cause: The requested daemon service requires an administrative login.

Action: Edit the daemon security in Oracle Studio to specify a valid administrator user name and password.

See Also: Daemon "Security" on page B-5.

C00F: Anonymous client logins are not allowed.

Cause: The daemon is configured to require a valid user name and password, which were not supplied.

Action: Enable anonymous client access in daemon security in Oracle Studio.

See Also: Daemon "Security" on page B-5.

Action: Edit the user profile in Oracle Studio to specify a valid user name and password for the IBM z/OS platform.

See Also: Setting Up Run-Time User Access to the IBM z/OS Platform on page 2-4.

C010: Anonymous server logins are not allowed.

Cause: Internal error.

Action: Contact Oracle Support Services.

C011: Client has already timed out.

Cause: A server process was started on behalf of a client and the client has timed out before the server completed its startup.

Action: Increase the Connect timeout value for the server workspace in the Workspace General tab of the daemon configuration.

> **See Also:** Workspace "General" on page B-7.

C012: Invalid username/password.

Cause: Invalid user name/password supplied when logging on to the daemon.

Action: See the daemon log file because the user name/password were not accepted.

Action: Edit the user profile in Oracle Studio to specify a valid user name and password for the IBM z/OS platform.

> **See Also:** Setting Up Run-Time User Access to the IBM z/OS Platform on page 2-4.

Action: Ensure that the daemon is started from an APF-authorized account that is allowed to check for system user names and passwords.

C014: Client connection limit reached - try later.

Cause: The maximum number of server processes for the workspace has been reached, and none of the active servers could accept the client connection.

Action: Increase the value of the `Number of sub-tasks` in the **Server Mode** tab of the workspace configuration.

> **See Also:** "Server Mode" on page B-11.

Action: Try running the command later.

C015: Failed to start server process.

Cause: The Oracle Connect daemon failed to start a server process or the started server failed upon starting up.

Action: See the daemon and server logs for the reason the server did not start. For example, you might receive an message with a reason specified in the log file similar to the following: `[C015] Failed to start NAVIGATOR server process: No server account name defined for anonymous client; code: -1601: SQL code: 0`

Action: If you use impersonation, check the user profile on the client. Also see C069.

To set impersonation: APF authorize all the steplibs in the server script on z/OS computer. For example:

```
setprog... ada622-volume adavol
       CICS.CICS.SDFHEXCI - p390dx
       INSTROOT.load - 111111
       INSTROOT.loadaut - 111111
```

INSTROOT is the high level qualifier where Oracle Connect is installed.

In the Workspace **Security** tab of the Navigator workspace under the daemon node in the Configuration Explorer, check **Use specific workspace account** and clear the **Workspace account** field of all values.

C016: Unexpected server state.

Cause: Internal error.

Action: Contact Oracle Support Services.

C017: Active daemon clients exist. Shutdown canceled.

Cause: One or more clients are still connected to the daemon.

Action: Wait until all the clients log off the daemon and then retry the shutdown operation.

C019: Request is not granted because someone else is locking it.

Cause: A request to lock a resource managed by the daemon was denied because another user has locked the resource.

Action: Wait for the other user to release the resource.

C01A: Lock %s not found.

Cause: A request to free a resource was denied because the caller did not lock that resource (for example, another user shut down the daemon you are working with).

Action: Contact Oracle Support Services.

C01B: Unexpected error in %s.

Cause: Internal error.

Action: Contact Oracle Support Services.

C01C: Cannot update configuration without _APPLICATIONS lock.

Cause: Internal error.

Action: Contact Oracle Support Services.

C01D: Need to lock the application first.

Cause: Internal error.

Action: Contact Oracle Support Services.

C01F: Cannot set configuration of a deleted application.

Cause: Internal error.

Action: Contact Oracle Support Services.

C020: Failed in looking up host name (gethostname())

Cause: Cannot connect to the remote computer.

Action: Check that the name specified for the computer in the oc4j-ra-xml file is correct.

Action: Check that a domain name server (DNS) is available to look up the host name.

Action: Check the TCP/IP subsystem on the computer by trying to ping it or run FTP or Telnet to or from it.

C021: Required variable %s not found

Cause: An environment variable required by the Oracle Connect server was not defined when the server started up.

Action: Check whether the startup script makes any changes to the environment variables used by Oracle Connect.

Action: Check whether the system-defined environment size is sufficiently large for Oracle Connect.

C022: Server failed to connect and register with the daemon.

Cause: An Oracle Connect server started by the daemon was not able to connect or register back with the daemon.

Action: Try to connect again.

Action: Increase the Connect timeout value for the server workspace in the Workspace **General** tab.

See Also: Workspace "General" on page B-7.

Action: Check that the startup script for the workspace launches the correct version of Oracle Connect.

Action: Increase the value of the `Set maximum number of servers` and/or `Maximum` parameter for the `Clients per server limit` in the Workspace **Server Mode** tab.

See Also: "Server Mode" on page B-11.

C023: Call made to unregistered module %d.

Cause: Internal error.

Action: Contact Oracle Support Services.

C024: Failed to create a socket.

Cause: An error occurred within the TCP/IP subsystem.

Action: Check whether you have sufficient system privileges.

Action: Check the TCP/IP subsystem on the computer by trying to ping it or run FTP or Telnet to or from it.

C025: Failed to set socket option %s

Cause: An error occurred within the TCP/IP subsystem.

Action: Check whether you have sufficient system privileges.

Action: Check the TCP/IP subsystem on the computer by trying to ping it or run FTP or Telnet to or from it.

C026: Failed to bind server to port %s

Cause: An Oracle Connect server or daemon was not able to bind to the specified port.

Action: Check whether another program is holding the port that was specified in the oc4j-ra-xml file for the adapter.

Action: Check whether you have sufficient system privileges.

C027: Cannot create TCP service for %s

Cause: An error occurred within the TCP/IP subsystem

Action: Check the TCP/IP subsystem on the computer by trying to ping it or run FTP or Telnet to or from it.

C028: Unable to register (%s, %d, tcp)

Cause: This error may happen when a portmapper is used (*host*:a) but the portmapper is not available.

Action: Enable the portmapper.

Action: Avoid using the portmapper (by not using :a when starting the daemon).

C029: Failed to create a server thread

Cause: Internal error.

Action: Contact Oracle Support Services.

C02A: Server thread failed to start

Cause: Internal error.

Action: Contact Oracle Support Services.

C02B: Stopping the %s server - no client

Cause: A server that was started by the Oracle Connect daemon to service a client did not get a client connection request within one minute. The server terminates.

Action: In most cases, the client was terminated by a user request, so no specific action is required.

Action: If no client can connect to the server, it may be that the server has multiple network cards and the Oracle Connect daemon is not aware of this. In this case, start the daemon with an IP address.

C02C: Unexpected event - a termination signal intercepted

Cause: Internal error.

Action: Contact Oracle Support Services.

C02D: Modified transport, context unknown/lost

Cause: Internal error.

Action: Contact Oracle Support Services.

C02F: Corrupted arguments passed to procedure

Cause: Internal error.

Action: Contact Oracle Support Services.

C030: Unable to free arguments for %s() of %s

Cause: Internal error.

Action: Contact Oracle Support Services.

C031: Cannot register a non-module RPC %s

Cause: Internal error.

Action: Contact Oracle Support Services.

C032: An IRPCD program is required

Cause: Internal error.

Action: Contact Oracle Support Services.

C033: An IRPCD super-server is required for module events

Cause: Internal error.

Action: Contact Oracle Support Services.

C034: An invalid super-server module ID was specified, %d

Cause: Internal error.

Action: Contact Oracle Support Services.

C035: Out of memory

Cause: Not enough memory to service a client request.

Action: Increase process memory quota or add memory to the system.

C036: Failed to register RPC procedure module %s

Cause: Internal error.

Action: Contact Oracle Support Services.

C037: Failed to register an invalid RPC procedure number %x

Cause: Internal error.

Action: Contact Oracle Support Services.

C038: Cannot reregister RPC procedure number %x

Cause: Internal error.

Action: Contact Oracle Support Services.

C042: Remote call to %s failed; %s

Cause: Remote call to API failed.

Action: Check the daemon log file.

Action: If necessary, change the level of detail written to the log file to help resolve the problem.

See Also: "Logging" on page B-3.

C043: Failed to connect to host %s;%s

Cause: The remote host is not correctly defined to Oracle Connect or is not working.

Action: Check the remote computer definition in the oc4j-ra-xml file for the adapter.

Action: Check that the daemon is up on the IBM z/OS platform. Use the Status option in the Runtime Manager perspective.

Action: Check the network connection by trying to ping the host computer or run FTP or Telnet to or from it.

C045: Failed to create a service thread

Cause: The server failed to create a thread to service a client request.

Action: A system or process quota limit has been exceeded. Either increase the quota or lower the `Clients per server limit` field value in the Workspace **General** tab.

See Also: Workspace "General" on page B-7.

C047: %s out of memory

Cause: Not enough memory was available to Oracle Connect to complete a requested operation.

Action: Terminate unnecessary processes running on the server.

Action: Add more memory to the system.

Action: Allow the process to use more memory.

Action: Limit the number of processes the daemon may start. If the demand for servers exceeds the number of available servers, clients get a message telling them the maximum number of servers has been reached and asking them to try again later.

C066: Communication error with the server%s

Cause: Connection to the Oracle Connect daemon or server failed, or an established session with a server has failed.

Action: Check the remote computer definition in the weblogic-ra-xml file.

Action: Check that the daemon is up on the IBM z/OS platform. Use the Status option in the Runtime Manager perspective.

Action: In case of a network problem, check the network connection by trying to ping the host computer or run ftp or telnet to or from it.

C067: Unexpected error occurred in server function %s

Cause: One of the server functions has exited with an exception, such as an Abend, or Invalid Instruction.

Action: Contact Oracle Support Services.

C068: Fail to login daemon

Cause: The daemon is not running on the server computer.

Action: Use the Status in Oracle Studio Runtime Manager perspective to check whether a daemon is running on the server

Action: Have the system administrator reinstall Oracle Connect on the server.

C069: Fail to get server

Cause: The Oracle Connect daemon on the server computer could not start a server process to serve the client. A separate message provides more detail on why the server process could not start.

Action: There are many possible causes of this error. If the cause is not clear from the related message, see the Oracle Connect daemon log file on the server

Action: The resolution to this error is highly dependent on the particular cause. The following are some typical causes and resolutions.

Action: Some process creation quota was exceeded. Either try again later or increase the quota or the other relevant system resources.

Action: The server startup script failed.

Action: The user name given is not allowed to use the requested server. Use an authorized user name.

Action: A limit on concurrent clients for a server has been reached. Try again later.

Action: If you use impersonation, check the user profile on the client. Also see C015.

C06A: Failed to connect to server

Cause: The server assigned to the client did not accept the client connection. A separate message provides more detail about why the server process did not accept the connection.

Action: See the daemon and server log files because the server was not available to accept its assigned client.

C06B: Disconnecting from server

Cause: A network failure, or a server computer failure or a server program failure caused the connection to stop. The currently active transaction is stopped as well.

Action: Oracle Connect automatically tries to reestablish a connection with a server upon the next SQL command issued against the server. Once the network or computer failure is corrected, the connection to the daemon is reestablished automatically.

C070: Server failed to send reply to the client

Cause: Server terminated unexpectedly.

Action: Unless the client was intentionally stopped (for example, using Control-C), contact Oracle Support Services.

C071: Connection to server %s was disconnected. Cursors state was lost.

Cause: Either a network failure, server failure, or server program failure caused the connection to stop. The currently active transaction is stopped as well.

Action: Normally, Oracle Connect automatically tries to create a session with the server upon the next attempt to access the server. If the network and server are accessible, the next operation should succeed. Otherwise, the network or server should be fixed before connection can be resumed.

Action: In case of a server malfunction not related to callable user code, contact Oracle Support Services.

C072: Reconnect to server %s

Cause: This is an informational message only. The client has reestablished its connection with the server.

Action: No action is required.

C073: The parameters passed to the admin server are invalid: %s

Cause: Internal error.

Action: Contact Oracle Support Services.

C074: No authorization to perform the requested operation (%s)

Cause: User/account has insufficient privileges.

Action: Grant administrative privileges to the user/account using the Administrator parameter of the Daemon **Security** or Workspace **Security** tabs in the daemon configuration.

> **See Also:** Daemon"Security" on page B-5 or "Security" on page B-16.

C075: Failed to register daemon in the TCP/IP service table

Cause: Registration of the daemon in the TCP/IP services file has failed.

Action: Check that the account running the daemon has the permissions to update the TCP/IP services file.

E001: Failed in lock/release operation

Cause: A lock or release operation of a global resource has failed. A separate message provides more details. The separate message specifies the cause of this error.

Action: There are various causes for this error, including lack of sufficient privileges or a system resource shortage.

J0006: Operation on already closed connection was requested

Cause: A request using a connection that was closed was attempted.

Action: Reopen the connection and try again.

J0028: Internal Error: Unknown XML tag %s

Cause: Internal error.

Action: Contact Oracle Support Services.

J0030: Internal Error: Method %s needs to be overwritten

Cause: Internal error.

Action: Contact Oracle Support Services.

J0031: Internal Error: Required attribute %s not found in %s verb

Cause: Internal error.

Action: Contact Oracle Support Services.

J0032: Internal Error: %s ACP object was returned instead of %s as expected

Cause: Internal error.

Action: Contact Oracle Support Services.

J0033: Internal Error: Attempt to work with closed socket

Cause: Internal error.

Action: Contact Oracle Support Services.

J0034: Internal Error: corrupted message; %s bytes read instead of %s as expected

Cause: XML sent from the client to the server has become corrupted.

Action: Check compression settings for XML transferred from the client to the server. If the setting are OK, retry sending the request from the client to the server.

J0035: Internal Error: Invalid redirection address %s returned by daemon

Cause: Internal error.

Action: Contact Oracle Support Services.

J0036: %s: %s

Cause: One of the following errors was received from the server: 0 - server.internalError, 1 - client.xmlError, 2 - client.requestError, 3 - client.noActiveConnection, 4 - server.resourceLimit, 5 - server.redirect, 6 - client.noSuchResource, 7 - client.authenticationError, 8 - client.noSuchInteraction, 9 - client.noSuchConnection, 10 - server.notImplemented, 11 - server.xaProtocolError, 12 - server.xaUnknownXID, 13 - server.xaDuplicateXID, 14 - server.xaInvalidArgument, 15 - client.autogenRejected, 16 - server.xaTransactionTooFresh, 17 - server.resourceNotAvailable, 18 - client.authorizationError, 19 - server.configurationError

Action: Review the server log file to determine the problem.

J0037: Internal Error: No ACP response when %s was expected

Cause: Internal error.

Action: Contact Oracle Support Services.

J0039: Internal Error: ACP root is not found in the XML

Cause: Internal error.

Action: Contact Oracle Support Services.

J0040: Internal Error: Input record is required for interaction %s execution

Cause: Internal error.

Action: Contact Oracle Support Services.

J0048: Invalid metadata type %s is passed to %s function

Cause: A request for metadata was not fulfilled.

Action: Check the validity of the request.

J0050: Key of the put method must be of type string

Cause: In either a GET or PUT operation, the key must be a string.

Action: Change the key used in the operation to a valid key.

J0059: Value %s is invalid for attribute %s

Cause: A request for metadata was not fulfilled.

Action: Check the validity of the request.

J0068: Value must be of type string

Cause: In a PUT operation, the value must be a string.

Action: Change the value used in the operation to a valid value.

J0069: Value must be of type MappedRecord

Cause: In a PUT operation, the value must be a mapped record.

Action: Change the value used in the operation to a valid value.

J0070: Value must be of type MappedRecord[]

Cause: In a PUT operation, the value must be mapped record array.

Action: Change the value used in the operation to a valid value.

J0071: Bad key for mapped record, #element or #element[] is required

Cause: In a PUT operation, the value must be mapped record array.

Action: Change the key used in the record to a valid key.

J0072: Value must be of type Object[]

Cause: In a PUT operation, the value must be mapped record array.

Action: Change the value used in the operation to a valid value.

J0078: In nonpersistent connection and nonkeep alive encryption is not supported - ignored

Cause: Encryption is not supported for nonpersistent connections.

Action: There is no action to take. This warning can be ignored.

J0079: Invalid argument passed to %s - Argument: %s, Value: %s

Cause: The value pass.

Action: Change the argument used to a number.

8

Advanced Features of OracleAS Adapters for CICS

Oracle Connect includes several tuning parameters that can improve performance. Specifically, the daemon can be configured to optimize communication between the IBM z/OS platform and a client. In addition, the binding environment can be tuned to optimize the request handling.

This chapter contains the following topics:

- Configuring the Daemon for High Availability
- Configuring a Binding Environment
- Migration Considerations
- Security Considerations
- Transaction Support

8.1 Configuring the Daemon for High Availability

The daemon workspace is responsible for allocating server processes to clients. You can configure a workspace to use a pool of server processes so that a server process is always available for a client request. Use Oracle Studio to maintain daemon and daemon workspace parameters to control the allocation of server processes and their management in a pool.

You can also have several daemon workspace configurations. Thus, you can create individual workspaces for use with different adapters.

8.1.1 Adding a New Daemon Workspace Configuration

Use Oracle Studio to add a new daemon configuration. You can set up different daemon configurations for different situations.

Perform the following steps to add a new daemon workspace configuration:

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. In the Design perspective, Configuration view expand the **Machines** folder.
3. Expand the machine you are working with.
4. Expand the **Daemons** folder. The daemon configurations available on this computer are listed.
5. Right-click **IRPCD** and select **New Workspace**. The New Daemon Workspace screen is displayed.

6. Specify a name for the new workspace and optionally, provide a description.
7. Specify whether you want it to have default settings or copy the properties of an existing workspace.

   To copy the properties of an existing workspace, click **Ellipsis** and select the workspace from which you want to copy the properties.
8. Click **Next**. The Select Scenario screen is displayed.
9. Select **Application Server using connection pooling** and click **Next**.
10. Continue through the wizard, specifying the required values for the workspace.
11. Click **Finish**.

The workspace is displayed under the **IRPCD** daemon node.

8.1.2 Editing the Workspace

You edit a workspace by using the tabs described in the following table:

Table 8–1 Workspace Properties tabs

Tab	Description
General	Specifies general information including the server type, the command procedure used to start the workspace, the binding configuration associated with this workspace (which dictates the data sources and applications that can be accessed) the timeout parameters, and logging information (which dictates the data sources and applications that can be accessed), the timeout parameters, and logging information.
Server Mode	Contains the workspace server information including features that control the operation of the servers started up by the workspace and allocated to clients.
Security	Specifies administration privileges, user access, ports available to access the workspace and workspace account specifications.

Use Oracle Studio to access these tabs, as follows:

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. In the Design perspective Configuration view, expand the **Machines** folder and then expand the machine where you want to edit the workspace.
3. Expand the **Daemons** folder. The daemon available on this computer are listed.
4. Expand the **IRPCD** daemon. The daemon workspaces are listed.
5. Right-click the workspace you are editing and select **Open**.
6. Click the tab that contains the information you want to edit. For full details of the tabs and the fields in these tabs, see Workspaces.
7. After editing the workspace, click **Save**.

8.1.3 Configuring the Server Mode

The server mode dictates how the daemon starts new processes. The daemon supports the following server modes:

- **singleClient**: Each client receives a dedicated server process. The account in which a server process runs is determined either by the client login information or by the specific server workspace.

  This mode enables servers to run under a particular user account and isolates clients from each other (because each receives its own process). However, this server mode incurs a high overhead due to process startup times and may use a lot of server resources (because it requires as many server processes as concurrent clients).

- **multiClient**: Clients share a server process and are processed serially. This mode has low overhead because the server processes are initialized. However, because clients share the same process, they may impact one another, especially if they issue lengthy queries.

  The number of clients that share a process is determined by the Clients per server limit (the maximum number of concurrent clients a server process for the current workspace accepts).

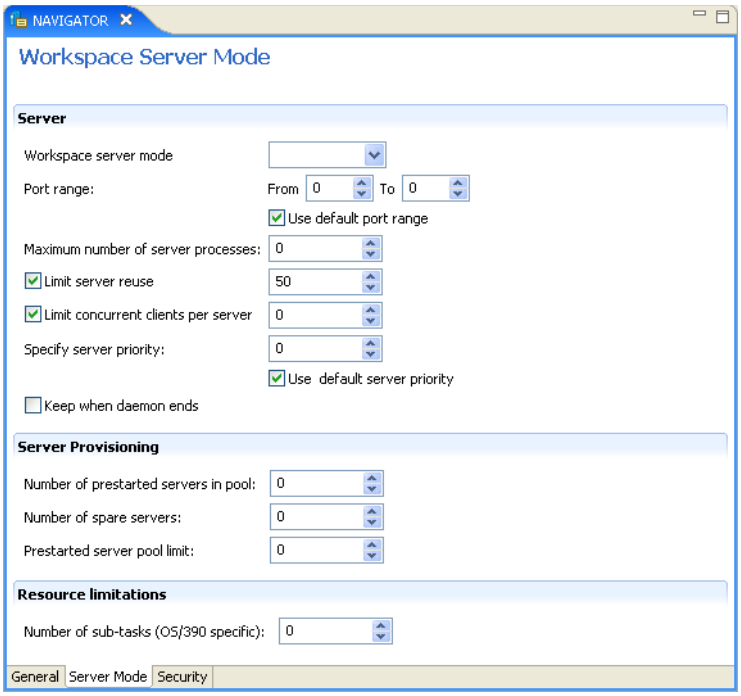
- **reusable**: This is an extension of the single client mode. Once the client processing finishes, the server process does not die and can be used by another client, reducing startup times and application startup overhead.

  This mode does not have the high overhead of single client mode because the servers are initialized. However, this server mode may use a lot of server resources (because it requires as many server processes as concurrent clients).

  The other modes can be set so that the server processes are reusable by setting the number of times a process can be reused with the `Reuse limit` value (the maximum number of times a particular server process can be reused or how many clients it can serve before it is retired). Reuse of servers enhances performance because it eliminates the need to repeat initializations. However, reuse runs a risk of higher memory leakage over time. The default value for the `Reuse limit` field is `None`, indicating that no reuse limit is enforced.

Set the server mode in the **Server Mode** tab of the daemon workspace editor as shown in the following figure:

Figure 8–1 The Server Mode Tab



When using any of the server modes you can specify a pool of server processes. These server processes are started when the daemon starts and are maintained in a pool. The server processes are available for use by new client requests from the pool, saving initialization time. Instead of starting a new server process each time one is requested by a client, the client receives a process immediately from the pool of available processes. When the client finishes processing, this server process either dies, or if reusable servers have been specified, it is returned to the pool.

You set up a pool of server processes by specifying the following parameters in the **Server Mode** tab.

- **Port Range**: Select the range for specific firewall ports through which you access the workspace. Determines the range of ports available for this workspace when starting server processes. Use this option when you want to control the port number, so that Oracle Connect can be accessed through a firewall.

  Enter the port range in the following fields:

  - **From**: Enter the highest numbered port in the range
  - **To**: Enter the lowest numbered port in the range

- **Use Default Port Range**: Select this to use the port range that is defined in the daemon.
- **Maximum number of server processes**: Enter the maximum number of server processes that can run at the same time.
- **Limit server reuse**: Select this to limit the number of servers that can be reused. If this is selected, the **Reuse limit** parameter is available.

  If **Limit server reuse** is selected, in the field next to the check box, enter the maximum number of times a server can be reused. Select the maximum of clients accepted in a server process.

A one-client server can be reused after its (single) client has disconnected. Reuse of servers enhances startup performance because it avoids the need to repeat initialization.

This parameter is not available if the Limit server reuse parameter is not selected.

This parameter is not available if the server mode value is **singleClient**.

- **Limit Concurrent clients per server**: Select this to limit the number of clients that a server can accept for the current workspace process.

  If this is not selected, the number of clients is unlimited.

  If **Limit concurrent clients per server** is selected, in the field next to the check box, enter the maximum number of clients that a server process for the current workspace accepts. The default for this field is **None**, indicating that the number of clients for each server is unlimited. This field is available if the server mode value is **multiClient** or **multiThreaded**.

- **Specify Server Priority**: Enter the server priority. For example, a workspace for applications with online transaction processing can be assigned a higher priority than a workspace that requires only query processing. The lower the number, the higher the priority. For example, workspaces with a priority of `1` are given a higher priority than workspaces with a priority of `2`.

  Note: This is unavailable if **Use default server priority** is selected.

- **Use default server priority**: Select this to set the priority to `0`. There is no specific priority for this workspace. Clear this check box to set a priority in the **Specify Server Priority** parameter.

- **Keep when daemon ends**: Select this to stop all servers started by that daemon when a daemon is shutdown, even if they are active. Select this if you want the servers for the workspace to remain active, even after the daemon has been shut down. If selected, it is the responsibility of the system operator or manager to ensure that the servers are eventually stopped. This must be done at the system level.

- **Number of prestarted servers in pool**: Enter the number of server processes that are prestarted for this workspace when the daemon starts. These are available for use by new client processes with minimal initialization time. Instead of starting a new server process each time one is requested by a client, the daemon immediately allocates (to the client) a server from a pool of available servers. When the number of available server processes drops lower than the value specified in the Minimum number of available servers field, the daemon again starts server processes until the specified number of available servers is reached. The default for this parameter is `0`, meaning that no servers are prestarted for this workspace.

- **Number of spare servers**: Enter the minimum number of server processes in the prestarted server's pool before the Oracle Connect daemon resumes creating new server processes (up to the number specified in the Initial number of servers field value, described earlier). If this parameter is set to a value greater than the Initial number of servers field value, the daemon considers the value to be the same as the value specified in the Initial number of servers field. In this case, a new server process is started and added to the pool each time a server process is removed from the pool and allocated to a client). The default for this parameter is `0`, which means that new servers are created only when there are no other available servers.

- **Number of sub-tasks**: Enter the number of sub-tasks for a server that are prestarted for this workspace when the daemon starts. In addition to setting up a

pool of server processes as described earlier, you can set additional server processes as sub-tasks by specifying this parameter. If you set 10 servers and 10 prestarted sub-tasks then 100 tasks are started (10 sub-tasks for each process).

8.2 Configuring a Binding Environment

Each binding configuration includes the following:

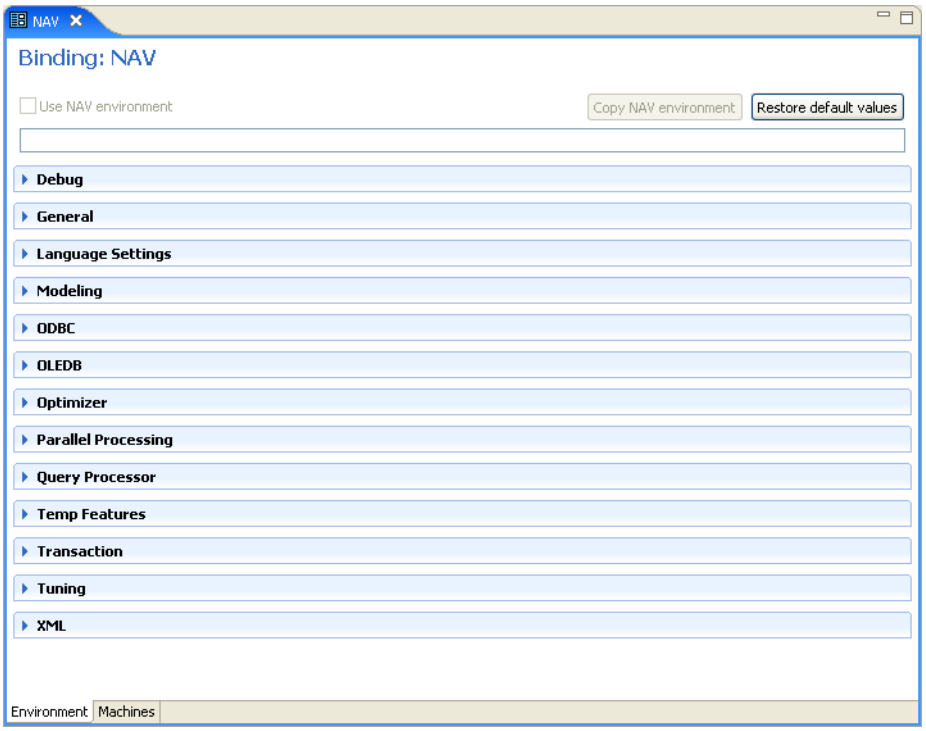
- Environment settings, which are used to configure the environment used by any of the adapters defined in the binding.
- Application adapters on the current computer.

To configure environmental settings in Oracle Studio perform the following steps:

1. From the **Start** menu, select **Programs**, **Oracle** and then select **Studio**.
2. In the Configuration Explorer, expand the node of the computer defined in Setting Up the IBM z/OS Platform in Oracle Studio on page 2-1.
3. Expand the **Bindings** node. The available binding configurations are listed.
4. Right-click **NAV** and select **Edit Binding**.
5. In the Properties tab edit the required environment settings as needed. To edit an environment setting, click the property category node, and then click the required value to edit.

   The binding **Properties** tab is shown in the following figure:

Figure 8–2 The Binding Properties tab



The binding environment is divided into the following categories:

- Debug
- General

- Language
- Modeling
- ODBC
- OLEDB
- Optimizer
- Query Processor
- Parallel Processing
- Transactions
- Tuning
- XML

8.2.1 Debug

The following table lists the parameters that define debugging and logging operations:

Table 8–2 Debug Parameters

Parameter	Description
ACX trace	Select this for the input xml sent to the back-end adapter and the output xml returned by the back-end adapter to be written to the log.
GDB Trace	This parameter is not applicable for use with OracleAS Adapter for CICS.
General trace	Select this to log general trace information. The default writes only error messages to the log. **Note**: Changing the default setting can degrade performance.
Query warnings	This parameter is not applicable for use with OracleAS Adapter for CICS.
Add timestamp to traced events	Select this to add a The properties for the column are displayed at the bottom of the tab on each event row in the log.
Query Processor trace	This parameter is not applicable for use with OracleAS Adapter for CICS.
Binary XML Log Level	Select the binary XML log level from the list. The following logging levels are available: ■ None ■ API ■ Info ■ Debug
Log file	The high-level qualifier of the log file for messages. The following type of message are written to the log: ■ Error messages ■ Trace information and information about the query optimization strategy, if `generalTrace` is set to `true`.
Trace Directory	This parameter is not applicable for use with OracleAS Adapter for CICS.

Table 8–2 (Cont.) Debug Parameters

Parameter	Description
Optimizer trace	This parameter is not applicable for use with OracleAS Adapter for CICS.
Transaction extended logging	Select this for the transaction manager to write additional information about transactions to the log.

8.2.2 General

The following table lists the parameters that define general operations where temporary files are written:

Table 8–3 General Parameters

Parameter	Description
NAV_UTIL editor	This parameter is not applicable for use with OracleAS Adapter for CICS.
Temporary Dir	The directory where temporary files are written, including the temporary files created for use by hash joins and for sorting files. The default is the current high-level qualifier.
Year 2000 policy	Determines the way two-digit years are converted into four-digit years. When the `year2000Policy` parameter is not set, or when it is set to a value outside the range of values defined for the policy, as described in the following paragraphs, a default value of `5` and the Sliding Base Year policy is used. Two policies are provided: Fixed Base Year: `year2000Policy` is set to a value greater than, or equal to 1900. In this case, the value of `year2000Policy` is the first four-digit year after 1900 that can be represented by a two-digit year. For example, if `year2000Policy` is set to 1905, the years 2000->2004 is represented by 00->04. All other two digits are mapped to 19xx. This solution is most required if there is live data at the low end (close to the year 1900), which the user wants to keep with the current two-digit format. The user probably changes the base date only after ensuring that these old dates have been deleted from the data source. Sliding Base Year: `year2000Policy` is set to a positive value less than 100. In this case, the value of `year2000Policy` represents the number of years ahead of the current year that can be represented by a two-digit number. With each passing year the earliest year that can be represented by a two-digit number changes to a year later.
Cache buffer size	Enter the number of bytes to be used for a memory buffer on a client computer, which is used by the Oracle Connect client/server to store read-ahead data. The default is `200000`.

8.2.3 Language

The following table lists the parameters that define globalization support:

Table 8–4 Language Settings Parameters

Parameter	Description
Language	Identifies the application language. A default code page is selected based on the value specified for this parameter. For more information, see Appendix D, "Globalization Settings".
Code Page	For use with globalization support to identify the code page for the workspace. For more information, see Appendix D, "Globalization Settings".
NLS String	Specifies the code page used by a field whose data type is defined as `nlsString`. Use this for a field whose code page is other than that of the computer code page. This parameter includes the following values: ■ The name of the code page. ■ Whether the character set reads from right to left (as in middle eastern character sets). The default is `false`.

8.2.4 Modeling

The **Modeling** parameters are not applicale with OracleAS Adapter for CICS.

8.2.5 ODBC

The **ODBC** parameters are not applicable for use with OracleAS Adapter for CICS.

8.2.6 OLEDB

The **OLEDB** parameters are not applicable for use with OracleAS Adapter for CICS.

8.2.7 Optimizer

The **Optimizer** parameters are not applicable for use with OracleAS Adapter for CICS.

8.2.8 Query Processor

The **Query Processor** parameters are not applicable for use with OracleAS Adapter for CICS.

8.2.9 Parallel Processing

The **Parallel Processing** parameters are not applicable for use with OracleAS Adapter for CICS.

8.2.10 Transactions

The following table lists the parameters that define transaction support:

Table 8–5 Transaction Parameters

Parameter	Description
Transaction extended logging	Select this to write extended information about transactions to the transaction manager log files.

Table 8–5 (Cont.) Transaction Parameters

Parameter	Description
Commit on destroy	Select this to commit all single-phase commit transactions opened for a data source, if a connection closes while the transaction is still open.
Disable 2PC	Select this to disable two-phase commit capabilities, even in drivers that support two phase commit.
User commit confirm table	This parameter is not applicable for use with OracleAS Adapter for CICS.
Transaction log file	The high-level qualifier and name of the log file that logs activity when using transactions. The `logfile` parameter can also include the keyword `NORRS` after a comma (so that the format is `log,NORRS`) when RRS is not running on the z/OS platform.
Recovery Delay	The number of minutes from the start of a transaction before any recovery operation on that transaction can be attempted. The default is `15 minutes`.
Time limit	Enter the time to wait for a transaction to complete before an error is returned. This parameter is also used when performing a RECOVERY, and it then indicates the number of minutes to wait before a forced activity can be performed, since the last transaction activity.
Conversions Select one:	
	No conversion: Select this if you want all transactions to remain as sent. This is selected by default **Convert all to distributed**: Select this to convert all simple transactions into distributed transactions. **Convert all to simple**: Select this to convert all distributed transactions into simple transactions.

8.2.11 Tuning

The **Tuning** parameters are not applicable for use with OracleAS Adapter for CICS.

8.2.12 XML

The following table lists the parameters that define XML support:

Table 8–6 XML Category Parameters

Parameter	Description
COM maximum XML in memory	Specifies the maximum size of an XML document held in memory. The default is `65535` bytes.
COM maximum XML size	Specifies the maximum size of an XML document passed to another computer. The default is `65535` bytes. **Note**: When you increase this value for this property, you may need to increase the value for the **Maximum XML in memory** property in the daemon. For more information on daemons, see "Control".
COM XML transport buffer size	Enter the maximum size of the internal communications buffer. The default value (`-1`) indicates there is no size limit.

Table 8–6 (Cont.) XML Category Parameters

Parameter	Description
XML date format	Enter the date format to use for XML. The options are: ■ `ISO` (the default): The date format is: `YY-MM-DDThh:mm:ss[.ss..]` ■ `ODBC`: The date format is: `YYYY-MM-DD HH:MM:SS[.NNN...]`
Replace invalid XML characters	Select this to replace invalid XML characters with a `'?'`. This property is used for diagnostic and troubleshooting purposes.
XML trim char columns	Select this to enable padded spaces to be trimmed from XML string columns when the record format is fixed. By default this is selected, and padded spaces are trimmed for fixed size character columns. If you do not want this behavior, clear this check box.

8.3 Migration Considerations

You can migrate an adapter configuration from one platform to another. The configuration information is stored in the Oracle Connect repository on the source platform and is exported to an XML file which can then be imported to the target platform.

Note that when migrating a configuration, file names and paths that are specific to the source platform must be changed to valid files on the target platform.

Perform the following steps to migrate an adapter configuration using Oracle Studio.

1. From the **Start** menu, select **Programs**, **Oracle** and then select **Studio**.
2. In the Configuration Explorer, right-click the computer defined in Setting Up the IBM z/OS Platform in Oracle Studio on page 2-1 and select **Export XML definitions**.
3. Specify the path and name of the XML file, which stores the XML representation and complete configuration.
4. Edit any paths in the XML definition to the paths required on the target platform. For example, the setting for the `serverLogFile` parameter may need changing, depending on the platform.
5. Set up the target platform in Oracle Studio in the same way you set up the source platform, as described in Setting Up the IBM z/OS Platform in Oracle Studio on page 2-1.
6. In the Configuration Explorer, right-click the target computer and select **Import XML definitions**.
7. Import the XML file to the target platform.

8.4 Security Considerations

Oracle Connect works within the confines of the legacy platform security system. For example, on an z/OS computer with RACF installed, and with the workspace server mode set to `multi-tasking`, a `RACROUTE VERIFY` is performed for each task in the address space, according to the client connection.

In addition, Oracle Connect provides the following security components:

- A binary XML encryption mechanism, which is activated as follows:

1. The client's first message to the server includes a pre-defined shared key, including the key name and value in the connection string. The server gets the key value for the key name passed from the client from the native object store (NOS).
2. The server generates a random 128-bit RC4 session key which is returned encrypted to the client, using the shared key. If no predefined shared key is provided, then a predefined, hardcoded key is used (this key is hardcoded on the client and on the server).
3. Passwords are always encrypted when passed over the wire, using an RC4, 128-bit session key, regardless of whether the entire session is encrypted or not.
4. If a predefined shared key was provided, then the entire session is encrypted. Otherwise, only the password exchange is encrypted (using the hardcoded key).

- Credentials: Passwords and user names exchanged over the network are encrypted using a pre-defined, hardcoded, 128-bit RC4 session key.
- Design Time: Security within Oracle Studio to grant access to Oracle Studio itself and to grant access to computers, user profiles and workspaces.
- Run-time: Security used to access resources from an application, including encryption over the network and controlling the daemon for the access.

8.4.1 Setting Design Time Security

Set the design time security as described in the following sections:

- Securing access to Oracle Studio is described in Setting Password Access to Oracle Studio on page 2-2.
- Securing rights to configure a computer in Oracle Studio is described in Specifying Users with Administrative Rights on page 2-3.
- Securing access to user profiles is accomplished by right-clicking the relevant user profile in Oracle Studio and selecting **Change Master Password**. In the dialog box that is displayed, specify a password that must be provided in the future to edit the specific user profile.
- Securing access to workspaces is accomplished by right-clicking the relevant workspace in Oracle Studio and selecting **Set Authorization**. In the dialog box that opens, specify a valid user and password that must be provided in the future to edit the specific workspace.

8.4.2 Setting Run-time Security

During run-time, security considerations are implemented as follows:

- When the client request accesses the legacy platform through the daemon, either anonymous access is allowed or a valid user name and password must be provided for the computer in the user profile. The `userName` and `password` properties in the J2CA 1.5 CICS adapter are used at this stage to access the daemon.

> **Note:** The user name used to access the daemon must also be the name of a user profile used.

- Access by the client must be through a valid port, according to the list of ports specified in the Workspace Access section of the Workspace Security tab in Oracle Studio.

  Note: Access to the legacy platform through a firewall using the NAT protocol is specified when the computer is added to Oracle Studio.

- To be allocated a server process, the client must be granted anonymous access to the workspace or be listed in the Workspace Users section of the Workspace Security tab in Oracle Studio.
- To access a server through a firewall, the daemon must be specified with a range of port numbers that are recognized by Oracle Connect. This is specified in the Enable Port Range section of Workspace Security tab in Oracle Studio.
- The ability to run commands on the daemon, such as starting or stopping a daemon or ending server processes is available only to administrators who have been registered in Oracle Connect as a daemon administrator. A client is registered as a valid daemon administrator in the **Daemon Security** tab in Oracle Studio, as described in "Security" on page B-5.

  Note: You can also specify administrators who can run commands only at the level of the workspace. Specify these administrators in the Workspace Security tab.

8.5 Transaction Support

OracleAS Adapter for CICS supports global transactions and can fully participate in a distributed transaction.

Note: To use OracleAS Adapter for CICS with 2PC, you must have RRS installed and configured.

To work with global transactions, select **Convert all to distributed** in the Transaction section of the binding environemtal properties. For more information, see Transactions in the Configuring a Binding Environment section.

To use the global transaction capability to access data on the z/OS computer, define every library in the ATTSRVR JCL as an APF-authorized library.

To define a DSN as APF-authorized, in the SDSF screen enter the command:

```
"/setprog apf,add,dsn=instroot.library,volume=ac002"
```

Where ac002 is the volume where you installed Oracle Connect and *INSTROOT* is the high level qualifier where Oracle Connect is installed.

If the Oracle Connect installation volume is managed by SMS, then when defining APF-authorization, enter the following command in the SDSF screen:

```
"/setprog apf,add,dsn=instroot.library,SMS"
```

Ensure that the library is APF-authorized, even after an IPL (restart) of the computer.

If RRS is not running, OracleAS Adapter for CICS can participate in a distributed transaction, as the only one-phase commit resource, if the T**ransaction log file** environment property includes the keyword `NORRS`. For more information, see Transactions in the Configuring a Binding Environment section.

Note: If a log file is not specified, then the format for the `logFile` parameter when RRS is not running is: `,NORRS`.

A

Advanced Tuning of the Metadata

Oracle Studio enables you to define outbound adapter interactions. In addition, Oracle Studio defines input and output structures used for these interactions. The interactions and input and output structures are maintained as metadata by Oracle Studio in the **Metadata** tab of the Design perspective.

This appendix contains the following sections:

- Adapter Metadata

A.1 Adapter Metadata

Use Oracle Studio to maintain the metadata for your database adapter.

1. From the **Start** menu, select **Programs**, **Oracle**, and then select **Studio**.
2. Expand the **Machines** folder, then expand the machine you are working with.
3. Expand the **Bindings** folder.
4. Expand the **NAV** binding.
5. Expand the **Adapters** folder to display the adapters list.
6. Right-click the adapter, and select **Show Metadata View** to open the Metadata view.

   You can create and edit the adapter metadata as follows:

   - Adapter Metadata General Properties: Enter and edit information about the adapter, such as the adapter name and the way in which you connect to the adapter. You make these changes in the Design perspective, Metadata view.
   - Adapter Metadata Interactions: Enter details of an interaction. The interaction Advanced tab is displayed for some adapters only, such as the Database adapter and includes more details about the interaction.
   - Adapter Metadata Schema Records: The input and output record structure for a record in the adapter definition.

A.1.1 Adapter Metadata General Properties

You can enter and edit information about the adapter, such as the adapter name and the way in which you connect to the adapter. You make these changes in the Design perspective, Metadata view. The following describes how to open the Adapter General Properties editor.

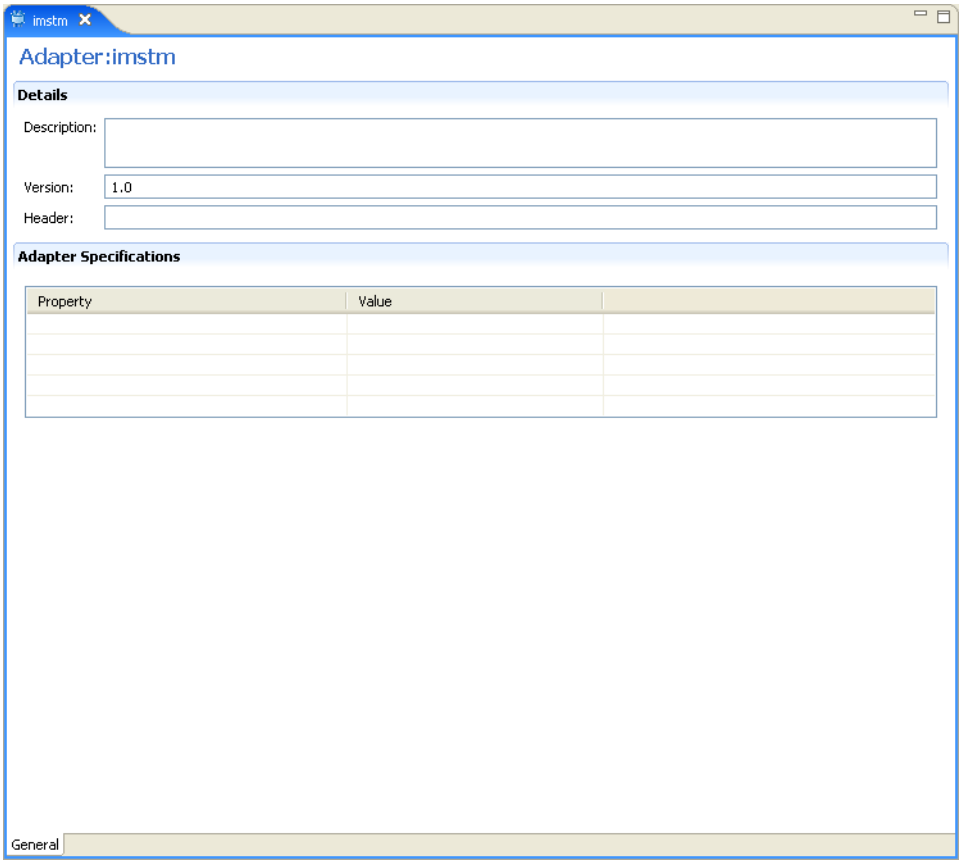
1. In Oracle Studio Design perspective, Metadata view, expand the **Adapters** folder.

2. Right-click the adapter to edit, and select **Open**.

   The General properties editor is displayed.

The Adapter General Properties editor is shown in the following figure:

Figure A–1 Adapter Metadata General Properties



The **General** properties are listed in the following table:

Table A–1 General tab Components

Field	Description
Description	Specifies an identifying description of the adapter.
Authentication mechanism	Specifies the authentication to access the adapter. The available mechanisms are: ▪ kerbv5 ▪ none ▪ basic password
Max request size	Specifies the maximum size in bytes for an XML request or reply. Larger messages are rejected with an error.
Max active connections	Specifies the maximum number of simultaneous connections for an adapter (per process).
Max idle timeout	Specifies the maximum time, in seconds, that an active connection can stay idle. After that time, the connection is closed.

Table A–1 (Cont.) General tab Components

Field	Description
Adapter Specifications	Specifies the adapter-specific properties for an interaction. The CICS adapter does not have any adapter-specific properties.

A.1.2 Adapter Metadata Interactions

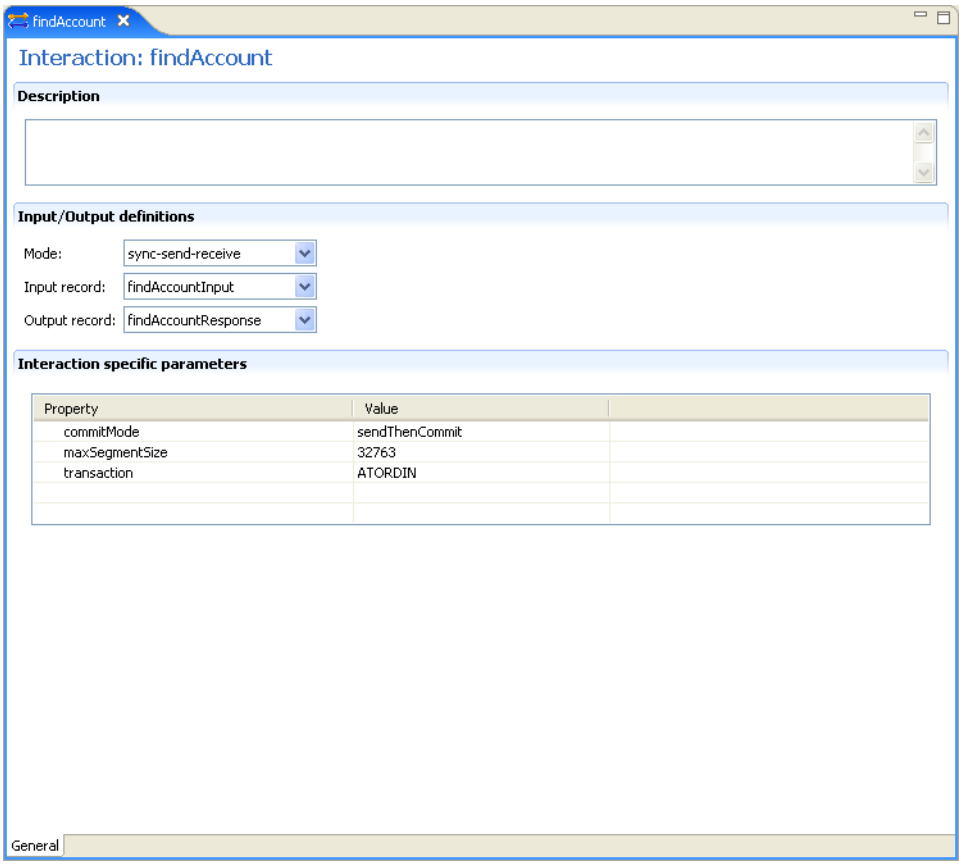
The Adapter Metedata Interactions editor defines an interaction and its input and output definitions. The following describes how to open the Adapter Metadata editor.

1. In Oracle Studio Design perspective, Metadata view, expand the **Adapters** folder.
2. Expand the adapter with the interaction to edit.
3. Expand the **Adapter** folder.
4. Right-click the adapter you want to edit and select **Open**.

   The Adapter Metadata Interactions editor is displayed.

The Adapter Metadata Interactions editor is shown in the following figure:

Figure A–2 Adapter Metadata Interactions



The Adapter Metadata Interaction properties are listed in the following table:

Table A–2 Interaction General Editor Components

Field	Description
Description	Provides a descriptive identifier for the interaction.
Mode	Determines the interaction mode. The following interaction modes are available: ■ sync-send-receive: The interaction sends a request and expects to receive a response. ■ sync-send: The interaction sends a request and does not expect to receive a response. ■ sync-receive: The interaction expects to receive a response.
Input record	Identifies an input record.
Output record	Identifies an output record for the results of an interaction.
Interaction Specific Parameters	Specific properties for the interaction. When an Interaction **Advanced** tab is used, this section is not displayed.

A.1.3 Adapter Metadata Schema Records

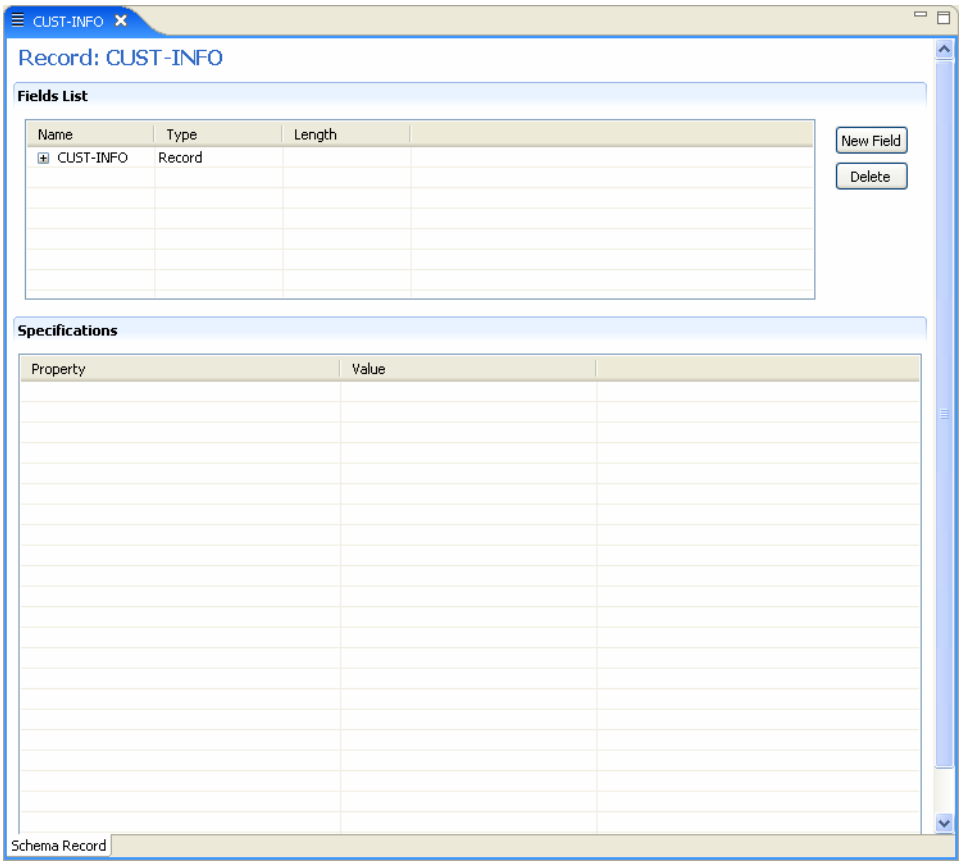
The Adapter Metadata Schema Records editor defines the general details of the input and output record structures for the interaction.The following describes how to open the Adapter Metadata Schema Records editor:

1. In Oracle Studio Design perspective, Metadata view, expand the **Adapters** folder.
2. Expand the adapter with the schema records to edit.
3. Expand the **Schemas**.
4. Right-click the schema you want to edit and select **Open**.

   The Adapter Metadata Schema Records editor is displayed.

The Adapter Metadata Schema Records editor is shown in the following figure:

Figure A–3 Adapter Metadata Schema Records



The Adapter Metadata Schema Records properties are listed in the following tables:

Table A–3 Schema Record Tab

Field	Description
Fields list	Defines the single data item within a record. This section has a table with the following three columns: ▪ **Name**: The name of the field ▪ **Type**: The data type of the field. For more information, see the Valid Data Types table for a list of the valid data types. ▪ **Length**: The size of the field including a null terminator, when the data type supports null termination (such as the string data type).
Specifications	Defines specific field properties. To display the properties, select the specific field in the Fields list.

The following table describes the valid data types that can be used when defining these specifications in the Schema Record editor.

Table A–4 Valid Data Types

Binary	Boolean	Byte
Date	Double	Enum

Table A–4 (Cont.) Valid Data Types

Binary	Boolean	Byte
Float	Int	Long
Numeric[(p[,s])]	Short	String
Time	Timestamp	

A.1.4 Adapter Metadata XML

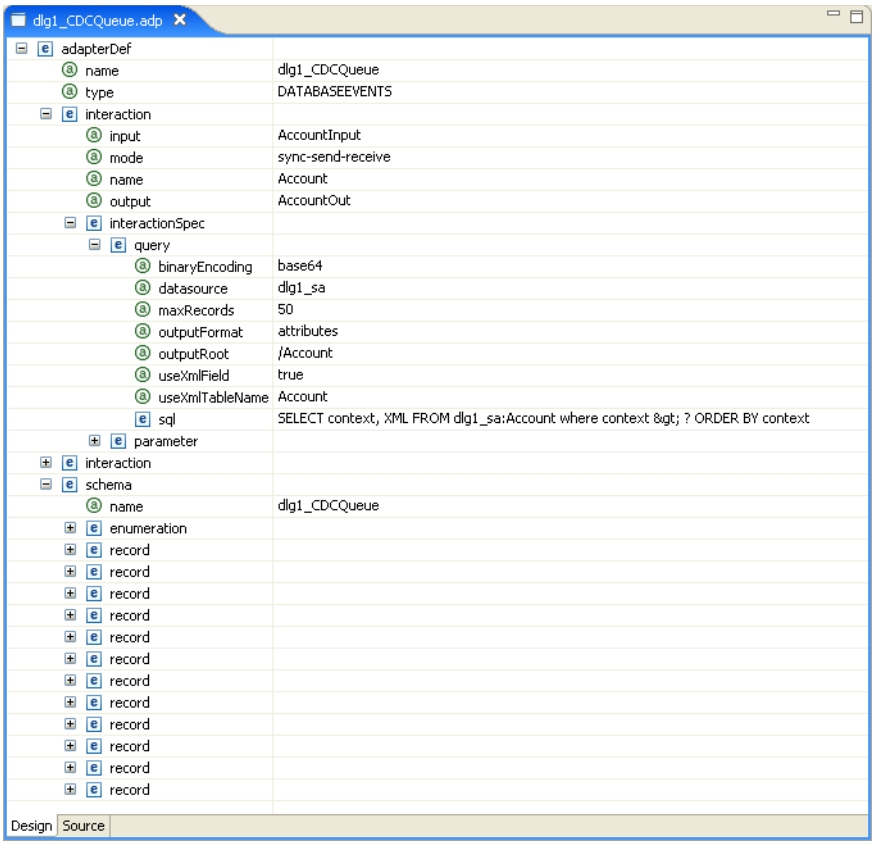
You can also edit the adapter metadata by viewing its XML schema. The following describes how to open the adapter metadata XML editor:

1. In Oracle Studio Design perspective, Metadata view, expand the **Adapters** folder.
2. Right-click the adapter to edit, and select **Open as XML**.

   The adapter XML editor is displayed in the Design view.

The adapter XML editor is shown in the following figure:

Figure A–4 Adapter Metadata XML Editor



For information on how to edit the properties in the XML editor Design view, see .

B

Advanced Tuning of the Daemon

The daemon configuration is managed using Oracle Studio. Daemon configuration is divided into the following groups:

- Control
- Logging
- Security
- Workspaces

B.1 Control

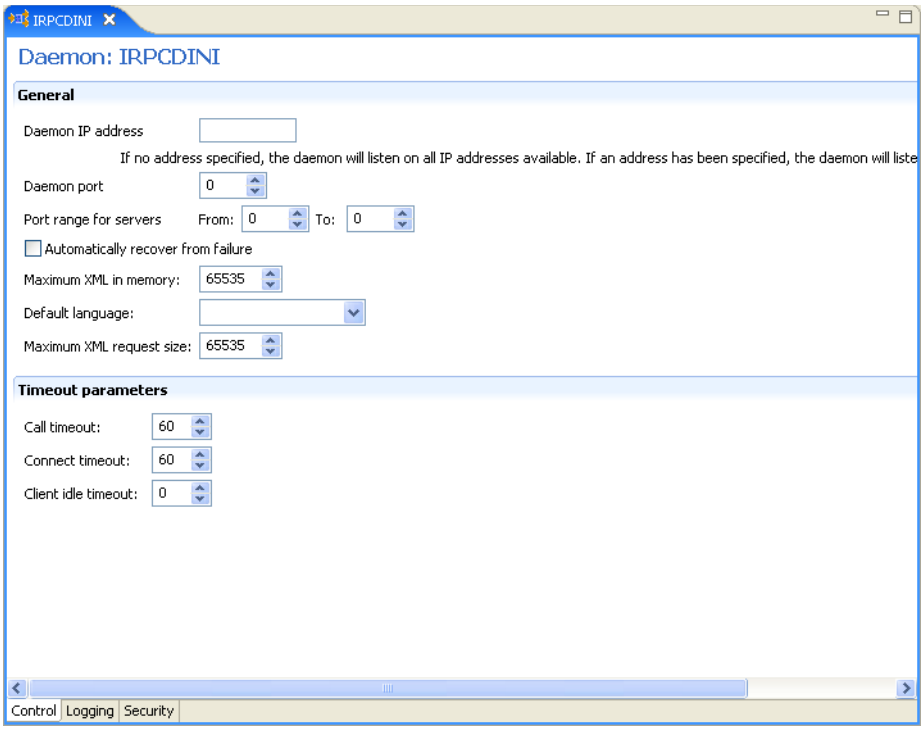
Using the **Control** tab for the daemon, you define various daemon control options. The **Daemon Control** tab is accessed as follows:

1. From the **Start** menu, select **Programs, Oracle**, and then select **Studio**.
2. From the Design perspective Configuration view expand the **Machines** folder.
3. Right-click the computer and select **Open Runtime Perspective**.
4. Right-click the required daemon in the Runtime Explorer and select **Edit Daemon Configuration**. The **Control** tab for the daemon is displayed in the editor.
5. After making changes to the daemon, right-click the daemon and select **Reload Configuration**.

> **Note:** You can also change daemon settings using the Configuration Explorer, by selecting a computer and scrolling the list to the required daemon. Right-click the daemon and select **Open**.
>
> Changes made to the daemon configuration are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager perspective.

The **Control** tab is shown in the following figure:

Figure B–1 The Control tab



The following table shows the parameters that can be set in the **Control** tab:

Table B–1 Daemon Control Tab Components

Field	Description
Daemon IP Address	Enter the IP address of the computer(s) where the daemon is listening. If no IP address is entered, the daemon listens on all available IP addresses.
Daemon port	Enter the port where the daemon is listening. If no port is entered, the daemon listens on all available ports.
Automatically recover from failure	The daemon restarts automatically if it fails for any reason (any error that causes the daemon process to terminate, such as network process lost or the CPU running the daemon crashes and the backup daemon is defined on another CPU). All available and unconnected servers are terminated and any connected servers are marked and terminated on release. Also the backup starts a backup for itself. The backup appends a new log file to the log of the original daemon, adding a line indicating that a backup daemon was started.
Maximum XML request size	The maximum number of bytes that the daemon handles for an XML document.
Default language	The language that the daemon supports. This setting is used when working with a client with a code page different from the server code page.
Maximum XML in memory	The maximum amount of space reserved for the XML in memory.

Table B–1 (Cont.) Daemon Control Tab Components

Field	Description
Call timeout	The timeout period for *short* calls for all daemons. The definition of a short call is a call that should be completed in a few seconds. For example, most calls to a database such as DESCRIBE should be completed in a few seconds as opposed to call like a GETROWS call, which can take a long time. In heavily loaded or otherwise slow systems, even short calls such as calls to open a file, may take a significant amount of time. If a short call takes more than the specified time to complete, then the connection is stopped. The default value for this parameter is 60 seconds. Values of less than 60 seconds are considered to be 60 seconds. Specifying the timeout in a workspace overrides the value set in this field for that workspace.
Connect timeout	The time the client waits for a daemon server to start. If the daemon server does not start within this period, then the client is notified that the server did not respond. The value specified for this parameter serves as the default timeout for all the workspaces listed in the daemon configuration. The default value for this parameter is 60 seconds. **Notes**: ■ Entering the timeout in a workspace overrides the value set in this field for that workspace. ■ Even if the XML source does not list this parameter in the workspace section, the workspace gets it using the default value. To prevent a workspace from using the default value, you must enter a value of zero for this parameter in the workspace section.
Client idle timeout	The maximum amount of time any daemon client may be idle before the connection with the server is closed. Specifying the timeout in a Workspace overrides this setting for that workspace.

B.2 Logging

Using the **Logging** tab, you define the daemon log file settings, the log file structure and the location where the log is saved. In addition, use it to define the data that is logged and traced in the file.

The following describes how to open the **Logging** tab.

1. From the **Start** menu, select **Programs, Oracle**, and then select **Studio**.
2. From the Design perspective Configuration view expand the **Machines** folder.
3. Right-click a computer and select **Open Runtime Perspective**.
4. Right-click the daemon in the Runtime Explorer and select **Edit Daemon Configuration**.
5. Click the **Logging** tab.
6. After making changes to the daemon, right-click the daemon and select **Reload Configuration**.

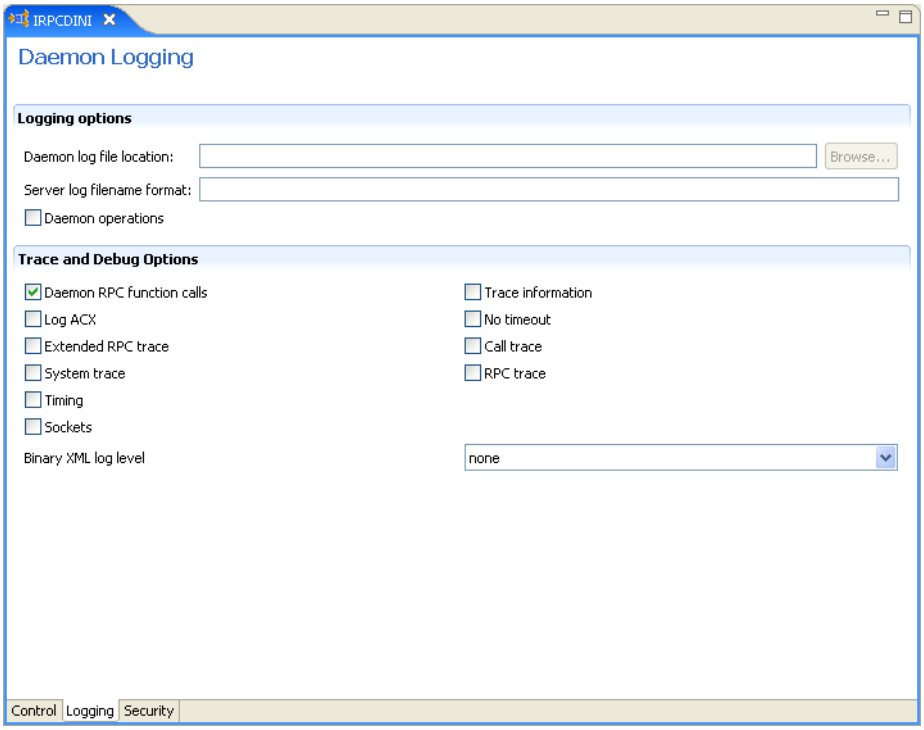
Note: You can also change daemon settings using the Configuration Explorer, by selecting a computer and scrolling the list to the required daemon. Right-click the daemon and select **Open**.

Changes made to the daemon configuration are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager perspective.

7. Right-click the daemon and select **Recycle Servers**. Any servers in the connection pool are closed and new servers start with the new configuration.

The **Logging** tab for the daemon is shown in the following figure:

Figure B–2 The Logging tab



The **Daemon Logging** tab comprises fields, as listed in the following table:

Table B–2 Daemon Logging Tab Components

Field	Description
Logging options	
Daemon log file location	Specifies the daemon produces its log data. The full path must be specified.
Server log filename format	Defines the name and location of the server log file. The field must specify the full path name. If no directory information is provided for the log file, then it is located in the login directory of the account running Oracle Connect workstation.
Daemon operations	Select this to log all of the daemon operations.

Table B–2 (Cont.) Daemon Logging Tab Components

Field	Description
Tracing and debug options	
Daemon RPC function calls	Select this to log all daemon RPC function calls.
Log ACX	Select this to log requests and processes.
Extended RPC trace	Generates a verbose message in the server log file for each low-level RPC function called. This is useful for troubleshooting the server.
System trace	Generates system-specific tracing of various operations.
Timing	Generates a time stamp for every entry to the server log file.
Sockets	Generates a message in the server log file for each socket operation.
Trace information	Select this to log low-level RPC operations.
No timeout	Disables the standard RPC timeouts, setting them to a long duration (approximately an hour) to facilitate debugging.
Call trace	Generates a message in the server log file for each RPC function called. This is useful for troubleshooting the server.
RPC trace	Enables debugging messages on the server.
Binary XML log level	Sets the binary XML log level. Your options are: ■ debug ■ none (the default) ■ api ■ info

The following tokens can appear in the log file template and is replaced accordingly:

- `%A`: workspace name
- `%D`: date (yymmdd)
- `%I`: instance number of the given workspace server
- `%L`: server account login directory
- `%P`: server process ID
- `%T`: time (hhmmss)
- `%U`: server account name (username)

For example, `%L/server_%A%I.log` may produce a log file such as: `/usr/smith/server_sales15.log`.

The default log file template is `%L/server_%A%I.log`.

B.3 Security

The following **Security** tab for the daemon is used to:

- Grant administration rights for the daemon.
- Determine access to the computer.

The following shows how to open the Daemon **Security** tab:

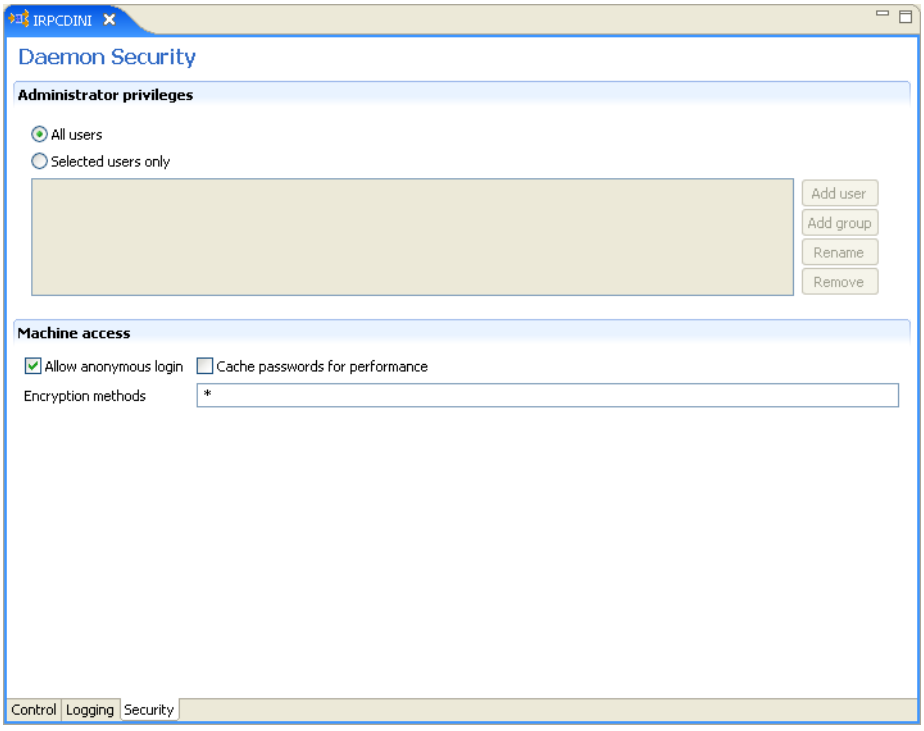
1. From the **Start** menu, select **Programs, Oracle**, and then select **Studio**.
2. From the Design perspective Configuration view expand the **Machines** folder.
3. Right-click the computer and select **Open Runtime Perspective**.
4. Right-click the daemon in the Runtime Explorer and select **Edit Daemon Configuration**.
5. Click the **Security** tab.
6. After making changes to the daemon, right-click the daemon and select **Reload Configuration**.

> **Note:** You can also change daemon settings using the Configuration Explorer, by selecting a computer and scrolling the list to the required daemon. Right-click the daemon and select **Open**.
>
> Changes made to the daemon configuration are not implemented. They are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager.

7. Right-click the daemon and select **Recycle servers**. Any servers in the connection pool are closed and new servers start with the new configuration.

The Daemon **Security** tab is shown in the following figure:

Figure B–3 The Daemon Security tab



The **Daemon Security** tab comprises fields, as listed in the following table:

Table B–3 Daemon Security Tab Components

Field	Description
Administrators privileges	Identifies the users (accounts) allowed to perform administrative tasks (tasks that require administrative login).
All users	Enables all users to access the daemon and change the settings.
Selected users only	Identifies the names of users (accounts) and groups that can be administrators.[1] If a user is not specified, the account from which the daemon was started is considered the administrator. Note that the daemon does not require the user to log in to the account on the system, but to log in to the daemon using the account name and password.
Machine access	Manages access to the computer.
Allow anonymous login	Whether workspaces allow anonymous logins (without user name/password entries). For the optimal level of security, keep this option unchecked and define a user name for the Daemon Administrators parameter. If unchecked, then no workspace can have an anonymous client. If checked, then a particular workspace allows anonymous clients.
Cached password	Enables login passwords to be cached. This enhances performance by reducing login times for future connections from the same client in a session.
Encryption methods	Specifies the encryption method used to send information across the network. The default is an asterisk (*), meaning that all methods are acceptable. If an encryption method is specified, it must be used. The RC4 and DES3 protocols are currently supported.

[1] The name is prefixed with '@', to use the operating system GROUP feature.

B.4 Workspaces

A daemon can include multiple workspaces. A workspace defines the server processes and environment that are used for the communication between the client and the server for the duration of the client request. Each workspace has its own definition. The workspace definition is divided into the following groups:

- General
- Server Mode
- Security

B.4.1 General

Using the **General** tab, you enter general information about the features that control the operation of the workspace, such as the server type, the command procedure used to start the workspace and the binding configuration associated with this workspace.

Do the following to open the **General** tab:

1. From the **Start** menu, select **Programs, Oracle**, and then select **Studio**.
2. From the Design perspective Configuration view, expand the **Machines** folder.

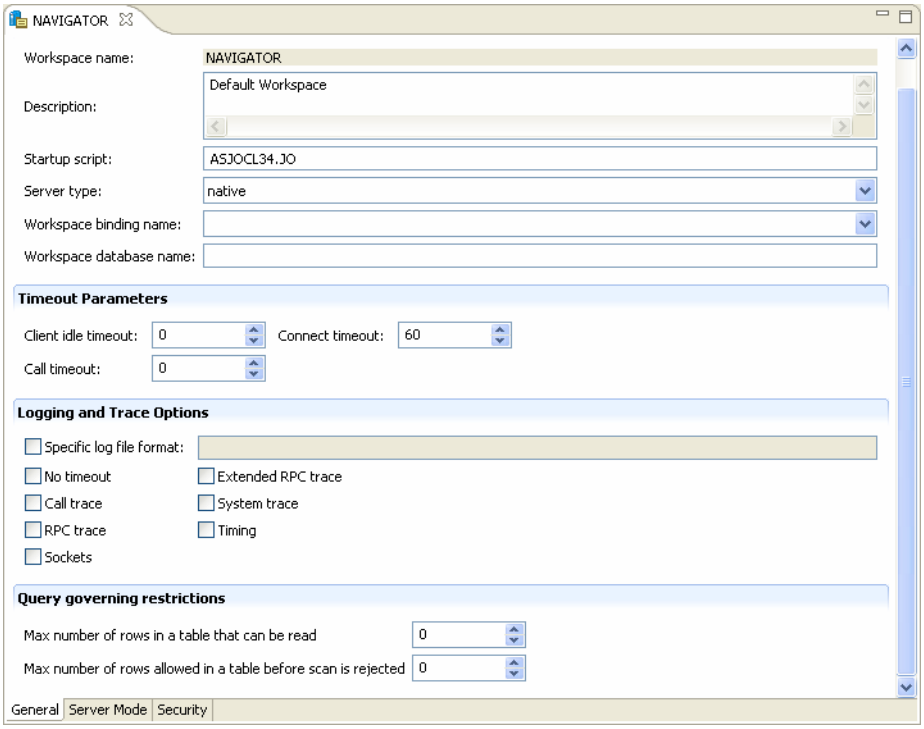
3. Right-click a computer and select **Open Runtime Perspective**.
4. Expand the Daemons node to display the workspaces in the Runtime Explorer.
5. Right-click the workspace and select **Edit Workspace Configuration**. The **WS Info**. tab opens.
6. After making changes to the workspace, right-click the daemon and select **Reload Configuration**.

> **Note:** You can also change daemon settings using the Configuration Explorer, by selecting a computer and scrolling the list to the required daemon. Right-click the daemon and select **Open**.
>
> Changes made to the daemon configuration are not implemented. They are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager.

7. Right-click the daemon and select **Recycle Servers**. Any servers in the connection pool are closed and new servers start with the new configuration.

The **General** tab is shown in the following figure:

Figure B–4 The General Tab



The **General** tab comprises fields, as listed in the following table:

Table B–4 General Tab Components

Field	Description
Workspace name	The name used to identify the workspace. **Note**: The default configuration includes the default Navigator workspace. This workspace is automatically used if a workspace is not specified as part of the connection settings.
Description	A description of the workspace.
Startup script	The full path name of the script that starts the workspace server processes. The script specified here must always activate the `nav_login` procedure and then run the server program (svc). If you do not specify the directory, the startup procedure is taken from the directory where the daemon resides. Oracle Connect includes a default startup script, which it is recommended to use.
Server type	This field is not applicable for use with OracleAS Adapter for CICS. *Do not change the default value*.
Workspace binding name	This field is not applicable for use with OracleAS Adapter for CICS.
Timeout parameters	The time the client waits for the workspace server to start. If the workspace server does not start within this period, then the client is notified that the server did not respond. Specifying the timeout here overrides the default setting, specified in the Control section. For more information, see Control for details about the Daemon Control section.
Client idle timeout	The maximum amount of time a workspace client can be idle before the connection with the server is closed.
Connect timeout	The time the client waits for a workspace server to start. If the workspace server does not start within this period, then the client is notified that the server did not respond.
Call timeout	The timeout period for *short* calls for all daemons. The definition of a short call is a call that should be completed in a few seconds. For example, most calls to a database such as `DESCRIBE` should be completed in a few seconds as opposed to call like a `GETROWS` call, which can take a long time. In heavily loaded or otherwise slow systems, even short calls such as calls to open a file, may take a significant amount of time. If a short call takes more than the specified time to complete, then the connection is stopped. The default value for this parameter is 60 seconds. Values of less than 60 seconds are considered to be 60 seconds. Specifying the timeout in a workspace overrides the value set in this field for that workspace.
Logging and Trace Options	

Table B–4 (Cont.) General Tab Components

Field	Description
Specific log file format	Defines the name and location of the server log file to write the data to a file instead of SYSOUT for the server process. The parameter must specify the name and the high level qualifier. The following tokens can appear in the log file template and are replaced accordingly: ■ `%A`: workspace name ■ `%D`: date (yymmdd) ■ `%I`: instance number of the given workspace server ■ `%L`: server account's login directory ■ `%P`: server's process ID ■ `%T`: time (hhmmss) ■ `%U`: server's account name (username)
Trace options	Specifies the type of tracing to be performed. The following tracing options are available: ■ **No timeout**: Select this to disable the standard RPC timeouts, setting them to a long duration (approximately an hour) to facilitate debugging. ■ **Call trace**: Select this to generate a message in the server log file for each RPC function called. This is useful for troubleshooting the server. ■ **RPC trace**: Select this to enable debugging messages on the server. ■ **Sockets**: Select this to generate a message in the server log file for each socket operation. This is useful for troubleshooting client/server communication - providing a detailed trace of every client/server communication. ■ **Extended RPC trace**: Select this to generate a verbose message in the server log file for each low-level RPC function called. This is useful for troubleshooting the server. ■ **System trace**: Select this to generate operating system-specific tracing. ■ **Timing**: Select this to generate a time stamp for every entry to the server log file.
Query governing restrictions	
Max number of rows in a table that can be read	Select the maximum number of table rows that are read in a query. When the number of rows read from a table exceeds the number stated the query returns an error.
Max number of rows allowed in a table before scan is rejected	Select the maximum number of table rows that can be scanned. This parameter has different behavior for query optimization and execution. ■ For query optimization, the value set is compared to the table cardinality. If the cardinality is greater than the value, the scan strategy is ignored as a possible strategy (unless it is the only available strategy). ■ For query execution, a scan is limited to the value set. When the number of rows scanned exceeds the number entered, the query returns an error.

B.4.2 Server Mode

The Server Mode tab lets you configure the features that control the operation of the servers started up by the workspace and allocated to clients.

For example, you can configure the workspace to start multiple servers for future use, before any client request, instead of starting each server when a request is received from a client.

Do the following to open the **Server Mode** tab:

1. From the **Start** menu, select **Programs, Oracle,** and then select **Studio**.
2. From the Design Perspective Configuration view, expand the **Machines** folder.
3. Right-click the computer and select **Open Runtime Perspective**.
4. Expand the Daemons node to display the workspaces in the Runtime Explorer.
5. Right-click the workspace and select **Edit Workspace Configuration**.
6. Click the **Server Mode** tab.
7. After making changes to the workspace, right-click the daemon and select **Reload Configuration**.

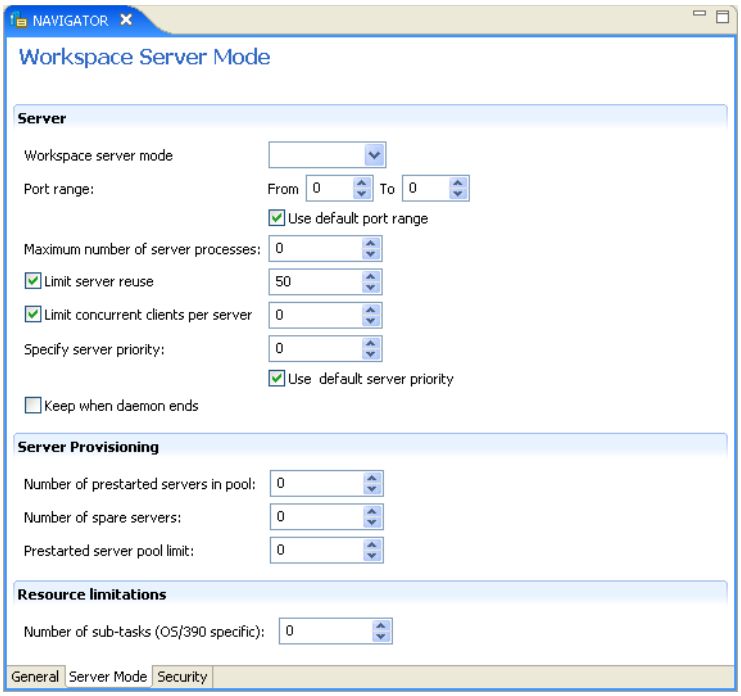
Notes:

- You can also change daemon settings using the Configuration Explorer, by selecting a computer and scrolling the list to the required daemon. Right-click the daemon and select **Open**.
- Changes made to the daemon configuration are not implemented. They are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager.

8. Right-click the daemon and select **Recycle Servers**. Any servers in the connection pool are closed and new servers start with the new configuration.

The **Server Mode** tab is shown in the following figure:

Figure B–5 The Server Mode tab



The **Server Mode** tab has the following fields:

Table B–5 Server Mode Tab Components

Field	Description
Workspace server modeCICS	Specifies the type of new server processes that the daemon starts. The daemon supports the following server modes: ■ **singleClient**: Each client receives a dedicated server process. The account in which a server process runs is determined either by the client login information or by the specific server workspace. This mode enables servers to run under a particular user account and isolates clients from each other, as each receives its own process. However, this server mode incurs a high overhead due to process startup times and can use a lot of server resources as it requires as many server processes as concurrent clients. ■ **multiClient**: Clients share a server process and are processed serially. This mode has low overhead because the server processes are initialized. However, because clients share the same process, they can impact one another, especially if they issue lengthy queries. The number of clients that share a process is determined by the Clients per server limit field. ■ **multiThreaded**: This mode is not applicable for use with OracleAS Adapter for CICS. ■ **reusable**: An extension of single-client mode. Once the client processing finishes, the server process does not die and can be used by another client, reducing startup times and application startup overhead. This mode does not have the high overhead of single-client mode because the servers are initialized. However, this server mode can use a lot of server resources as it requires as many server processes as concurrent clients. **Note**: The other modes can be set so that the server processes are reusable. The Reuse limit field value defines the number of times a process can be reused.
Reuse limit	Sets the maximum number of times a particular server can be reused. A one-client server can be reused after its (single) client has disconnected. Reuse of servers enhances startup performance because it avoids the need to repeat initialization. The default for this field is none (0), indicating that server reuse is unlimited. This parameter is disabled only if the server mode value is **singleClient**.
Clients per server limit	Sets the maximum number of clients a server process for the current workspace accepts. The default for this field is none (0), indicating that the number of clients for each server is unlimited. This field is enabled only if the server mode value is **multiClient** or **multiThreaded**.

Table B–5 (Cont.) Server Mode Tab Components

Field	Description
Server availability	Specifies the number of servers in a pool of servers, available to be assigned to a client. The following options are available: ■ **Initial number of servers**: The number of server processes that are prestarted for this workspace when the daemon starts. When the number of available server processes drops lower than the value specified in the Minimum number field, the daemon again starts server processes until this number of available server processes is reached. The default for this field is 0. ■ **Minimum number**: The minimum number of server processes in the prestarted pool before the daemon resumes creating new server processes (to the value specified in the Initial number of servers field). If this field is set to a value higher than the Initial number of servers field, the daemon uses the value specified in the Initial number of servers field. The default for this field is 0. ■ **Keep when daemon ends**: When a daemon is shutdown, all the servers started by that daemon are also stopped, even if they are active. Set this field to `true` for the workspace servers to remain active, even after the daemon has been shut down. If this field is set to `true`, it is the responsibility of the system operator or manager to ensure that the servers are eventually stopped. This must be done at the system level. ■ **Set maximum number of servers**: The maximum number of available server processes. Once this number is reached, no new nonactive server processes are created for the particular workspace. For example, if several server processes are released at the same time, so that there are more available server processes than specified by this field, the additional server processes higher than this value are terminated. The default for this field is zero, meaning that there is no maximum.
Port range	Determines the range of ports available for this workspace when starting server processes. Use this option when you want to control the port number, so that Oracle Connect can be accessed through a firewall. Enter the port range in the following fields: ■ **From**: enter the lowest numbered port in the range. ■ **To**: Enter the highest numbered port in the range.
Use default port range	Select this to use the port range that is defined in the daemon. This is defined in the Port range for servers field in the daemon Control tab.
Maximum number of server processes	Enter the maximum number of server processes that can run at the same time.
Limit server reuse	Select this to limit the number of servers that can be reused. If this is selected, the Reuse limit parameter is available.

Table B–5 (Cont.) Server Mode Tab Components

Field	Description
	If **Limit server reuse** is selected, in the field next to the check box, enter the maximum number of times a server can be reused. Select the maximum of clients accepted in a server process. A one-client server can be reused after its (single) client has disconnected. Reuse of servers enhances startup performance because it avoids the need to repeat initialization. This parameter is not available if the Limit server reuse parameter is not selected. This parameter is not available if the server mode value is **singleClient**.
Limit concurrent clients per server	Select this to limit the number of clients that a server can accept for the current workspace process. If this is not selected, the number of clients is unlimited. If Limit concurrent clients per server is selected, in the field next to the check box, enter the maximum number of clients that a server process for the current workspace accepts. The default for this field is **None**, indicating that the number of clients for each server is unlimited. This field is available if the server mode value is **multiClient** or **multiThreaded**.
Specify Server Priority	Set the priority for servers. For example, a workspace for applications with online transaction processing can be assigned a higher priority than a workspace that requires only query processing. The lower the number, the higher the priority. For example, workspaces with a priority of `1` are given a higher priority than workspaces with a priority of `2`. **Note**: This is unavailable if **Use default server priority** is selected.
Use default server priority	Sets the priority to `0`. There is no specific priority for this workspace. Clear this check box to set a priority in the **Specify server priority** parameter.
Keep when daemon ends	Select this to stop all servers started by that daemon when a daemon is shutdown, even if they are active. Select this for the workspace servers to remain active, even after the daemon has been shut down. If selected, it is the responsibility of the system operator or manager to ensure that the servers are eventually stopped. This must be done at the system level.
Server Provisioning	
Number of prestarted servers in pool	**Initial number of servers**: The number of server processes that are prestarted for this workspace when the daemon starts. When the number of available server processes drops lower than the value specified in the Minimum number field, the daemon again starts server processes until this number of available server processes is reached. The default for this field is 0.

Table B–5 (Cont.) Server Mode Tab Components

Field	Description
Number of spare servers	The minimum number of server processes in the prestarted pool before the daemon resumes creating new server processes (to the value specified in the Initial number of servers field). If this field is set to a value higher than the Initial number of servers field, the daemon uses the value specified in the Initial number of servers field. The default for this field is 0.
Prestarted server pool limit	The maximum number of available server processes. Once this number is reached, no new nonactive server processes are created for the particular workspace. For example, if several server processes are released at the same time, so that there are more available server processes than specified by this field, the additional server processes higher than this value are terminated. The default for this field is zero, meaning that there is no maximum.
Resource limitations	
Number of sub-tasks	The number of sub-tasks for a server that are prestarted for this workspace when the daemon starts. In addition to setting up a pool of server processes as described earlier, you can set additional server processes as sub-tasks by specifying this parameter. If you set 10 servers and 10 prestarted sub-tasks then 100 tasks are started (10 sub-tasks for each process).

B.4.3 Security

The **Security** tab lets you configure the security level for a workspace. this lets you set the security options for the workspace only. The Security tab is used to:

- Grant administration rights for the workspace
- Determine access to the workspace by a client

Do the following to open the **Security** tab:

1. From the **Start** menu, select **Programs, Oracle,** and then select **Studio**.
2. From the Design Perspective Configuration view, expand the **Machines** folder.
3. Right-click the computer and select **Open Runtime Perspective**.
4. Expand the Daemons node to display the workspaces in the Runtime Explorer.
5. Right-click the workspace and select **Edit Workspace Configuration**.
6. Click the **Security** tab.
7. After making changes to the workspace, right-click the daemon and select **Reload Configuration**.

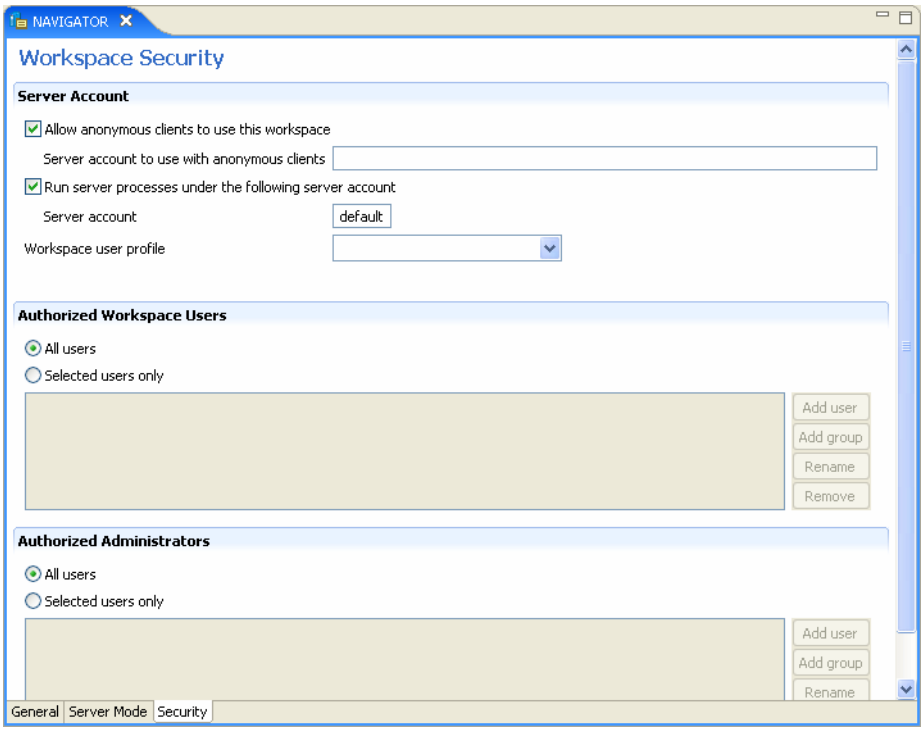
Notes:

- You can also change daemon settings using the Configuration Explorer, by selecting a computer and scrolling the list to the required daemon. Right-click the daemon and select **Open**.
- Changes made to the daemon configuration are not implemented. They are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager.

8. Right-click the daemon and select **Recycle Servers**. Any servers in the connection pool are closed and new servers start with the new configuration.

The **Security** tab is shown in the following figure:

Figure B–6 The Security tab



The **Security** tab has the following fields:

Table B–6 Security Tab

Field	Description
Server Account	This section defines the users (accounts) allowed to access the workspace, firewall access ports, workspace account, and anonymous login permissions.
Use specific workspace account	Select this to define the operating system account used for the workspace. If selected, enter the name of the workspace account in the **workspace account** field. If not selected, the account name that was provided by the client is used.
Allow anonymous clients to use this workspace	Select this to allow this workspace to be invoked without authentication. If selected, enter the name of the workspace account in the **Server account to use with anonymous clients** field.
Authorized Workspace users	Indicate which users have permission to use the workspace. Select one of the following ■ **All user**s: Any user who has logged on to the daemon may use the workspace ■ **Selected users only**: Select this to allow only users (or accounts) with specific permission to use the workspace. When this is selected, add the names of users (or accounts) and groups that can be use the workspace in the field below. **Note**: If no user is specified, any user who has logged on to the daemon may use the workspace.
Authorized Administrators	Identifies the users (accounts) with administrator privileges. Select one of the following: ■ **All users**: Indicates that anyone can access the workspace and change the settings. ■ **Selected users only**: Select this to allow only users (or accounts) with specific permission to be administrators. When this is selected, add the names of users (or accounts) and groups that can be workspace administrators. If no user is specified, any user who has logged on to the daemon may administrator this workspace.

C

Back-end Adapter Data Type Support

OracleAS Adapter for CICS supports several data types that are used when defining metadata in Oracle Studio. The data types are mapped from the COBOL data types during the import procedure.

> **Note:** The mapping of data types between OracleAS Adapter for CICS and Oracle Application Server is performed internally by Oracle Connect.

C.1 Data Type Mapping

The COBOL data type `COMP`, in the table is an abbreviation for, and synonymous with, COMPUTATIONAL. Square brackets ([ ]) denote optional qualifiers for some COBOL compilers, which may not be allowed for other COBOL compilers.

Table C–1 Data Type Mapping: COBOL and Oracle Connect Back-end Adapter

COBOL Data Type	Oracle Connect Data Type
`BINARY` (with fractional data)	`string`
`BINARY` (without fractional data)	`int`
`COMP` (with fractional data)	`string`
`COMP` (without fractional data)	`int`
`COMP-2`	`double`
`COMP-3`	`string`
`COMP-4` (with fractional data)	`string`
`COMP-4` (without fractional data)	`int`
`COMP-5` (with fractional data)	`string`
`COMP-5` (without fractional data)	`int`
`COMP-X` (with fractional data)	`string`
`COMP-X` (without fractional data)	`int`
`INDEX`	`int`
`[SIGN [IS]] LEADING`	`string`
`[SIGN [IS]] LEADING SEPARATE [CHARACTER]`	`string`
`NATIVE-2`	`int`

Table C–1 (Cont.) Data Type Mapping: COBOL and Oracle Connect Back-end Adapter

COBOL Data Type	Oracle Connect Data Type
NATIVE-4	int
NATIVE-8	string
PACKED-DECIMAL	string
POINTER	int
[SIGN [IS]] TRAILING	string
[SIGN [IS]] TRAILING SEPARATE [CHARACTER]	string

D

Globalization Settings

The OracleAS CDC Adapter for CICS provides the globalization support for the following languages:

- Arabic
- English (the default)
- French
- German
- Greek
- Hebrew
- Italian
- Japanese
- Korean
- Portuguses
- Simple Chinese
- Spanish
- Traditional Chinese
- Turkish

This appendix describes how to define the language support.

D.1 Defining the Language and Code Page

The language and code page parameters are accessed from the computer where Oracle Studio is installed.

Perform the following steps to define the required language and code page:

1. From the **Start** menu, select **Programs, Oracle**, and then select **Studio**.
2. In the Design perspective **Configuration** view, expand the **Machines** folder.
3. Expand the machine for which you want to set the language.
4. Expand the Bindings folder and right-click the **NAV** binding.
5. Select **Open**.
6. Expand the **Language Settings** and do the following:

- From the **Language** list, select the NLS supported language to use in this binding. Valid values are are listed in the Language Name column of NLS Language Codes table.
- From the **Codepage** list, select the code page to use with this language. The code pages available are determined by the Language that is selected. If you have additional code pages available, you can manually enter them in this field.

  Note: If you change the language, the code page also changes. Use the selected code page with the language you selected.

  If no code page is selected, the default code page for the selected language is used.
- From the **NLS string** list, select the NLS string for this language and code page. The NLS strings available are determined by the code page that is selected. If you have additional NLS strings available, you can manually enter them in this field.

  The code page is used by a field with a data type defined as `nlsString`. This parameter is used for a field with a code page that is different than the computer's code page. This property includes values for the name of the code page and whether the character set reads from right to left (as in middle-eastern character sets).

  For example, the following specifies a Japanese EUC 16-bit code page:

  ```
  <misc nlsString="JA16EUC,false"/>
  ```

7. Save the change. New servers use the language selected.

The following table lists the code pages:

Table D–1 NLS Language Codes

Language Name	Language Code	Windows Default	ASCII Platforms (Default)	EBCDIC Platforms (Default)	Alternative Codepages (EBCDIC based unless noted otherwise)
English US	ENUS	Windows-1252	ISO-8859-15	IBM1140	IBM285, IBM500, IBM1148, IBM1047, ISO-8859-1 (ASCII based)
English UK	ENUK	Windows-1252	ISO-8859-15	IBM1146	IBM285, IBM037, IBM500, IBM1140, IBM1148, IBM1047, ISO-8859-1 (ASCII based)

Table D–1 (Cont.) NLS Language Codes

Language Name	Language Code	Windows Default	ASCII Platforms (Default)	EBCDIC Platforms (Default)	Alternative Codepages (EBCDIC based unless noted otherwise)
French	FRE	Windows-1252	ISO-8859-15	IBM1147	IBM297, IBM037, IBM500, IBM1140, IBM1148, IBM1047, ISO-8859-1 (ASCII based)
Latin International	LAT	Windows-1252	ISO-8859-15	IBM1148	IBM500, IBM037, IBM1140, IBM1047, ISO-8859-1 (ASCII based)
Spanish	SPA	Windows-1252	ISO-8859-15	IBM1145	IBM284, IBM037, IBM500, IBM1140, IBM1148, IBM1047, ISO-8859-1 (ASCII based)
German	GER	Windows-1252	ISO-8859-15	IBM1141	IBM273, IBM037 ,IBM500, IBM1140, IBM1148, IBM1047, ISO-8859-1 (ASCII based)
Portuguese	POR	Windows-1252	ISO-8859-15	IBM1140	IBM037, IBM500, IBM1148, IBM1047, ISO-8859-1 (ASCII based)
Italian	ITL	Windows-1252	ISO-8859-15	IBM1144	IBM280, IBM037, IBM500, IBM1140, IBM1148, IBM1047, ISO-8859-1 (ASCII based)
Greek	GRK	Windows-1253	ISO-8859-7	IBM875	-
Russian[1]	RUS	Windows-1251	ISO-8859-5	IBM1154	IBM1025

Table D–1 (Cont.) NLS Language Codes

Language Name	Language Code	Windows Default	ASCII Platforms (Default)	EBCDIC Platforms (Default)	Alternative Codepages (EBCDIC based unless noted otherwise)
Turkish[2]	TUR	Windows-1254	ISO-8859-9	IBM1155	IBM1026
Hebrew	HEB	Windows-1255	ISO-8859-8	IBM424	IBM 862
Arabic	ARA	Windows-1256	ISO-8859-6	IBM420	
Japanese	JPN	SJIS	SJIS	IBM939	EUC-JP=EUC, VMS-JP
Chinese - Simplified	SCHI	GBK	GBK	IBM935	-
Chinese - Traditional	TCHI	BIG5	BIG5	IBM937	-
Korean	KOR	MS949	EUC-KR	IBM933	MS949

[1] Russian users who use ANSI 1251 Cyrillic as their Windows code page must edit the `RUS.TXT` file and compile it to RUS.CP using the NAV_UTIL CODEPAGE.

[2] To work with solutions in Oracle Studio, when using Turkish, add the `-nl en` switch to the Target path in the Oracle Studio shortcut properties. For example: `"C:\Program Files\Oracle\Studio1\studio.exe -nl en"`

E

Editing XML Files in Oracle Studio

In many cases you must manually edit the metadata to configure parts of a solution or composition. Metadata is created in XML format. You define aspects of a solution by changing the values of the elements and attributes of the XML files that belong to the solution. Oracle Studio provides a graphical interface where you can define the various aspects of a solution. This interface lets you make changes easily without having to manually edit the XML file.

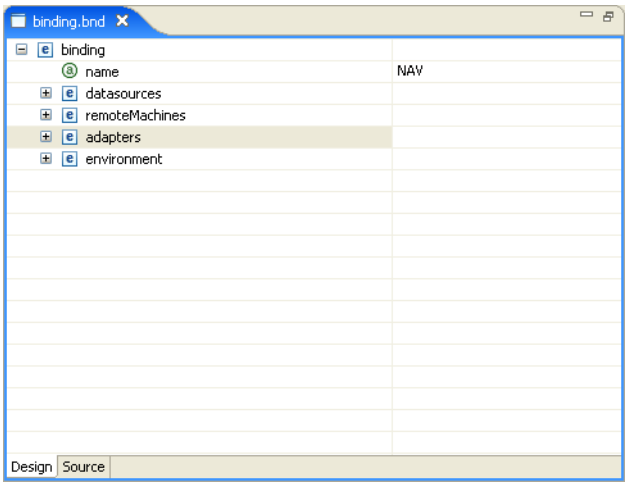
E.1 Preparing to Edit XML Files in Oracle Studio

You can edit XML files for the following items in Oracle Studio:

- Machines
- Bindings.
- Daemons
- Users

When you open an XML file, a graphical representation of the file is opened in the editor. The editor displays the elements and attributes in the file in the first column and their corresponding values in the second column. Each entry has an icon that indicates whether the entry is an element or an attribute. Click the **Source** tab to view the file in its native format. The following figure is an example of the editor's view of an XML file.

Figure E–1 XML Graphical Display



To edit an XML file in Oracle Studio

1. In the Design perspective, open the Navigator view.
2. In the Navigator view, find the item with the XML file to edit. This can be a machine, binding, daemon, or user.
3. Right-click the item and select **Open as XML**. A graphical list of the file's elements and attributes opens in the editor.
4. Find the element or attribute (property) to change.
5. Click in the right column next to the property you are changing and edit or add the value.
6. Save the file, then select it again in the Project Explorer and press F5 to refresh.

   The XML file is updated automatically.

E.2 Making Changes to the XML File

You can also make the following changes to XML files in Oracle Studio:

- Remove Objects
- Add DTD Information
- Edit Namespaces
- Add Elements and Attributes
- Replace an Element

E.2.1 Remove Objects

You can delete an element, attribute, or other object from the XML file.

To remove an object

1. Right-click an object from the list in the editor.
2. Select **Remove**.

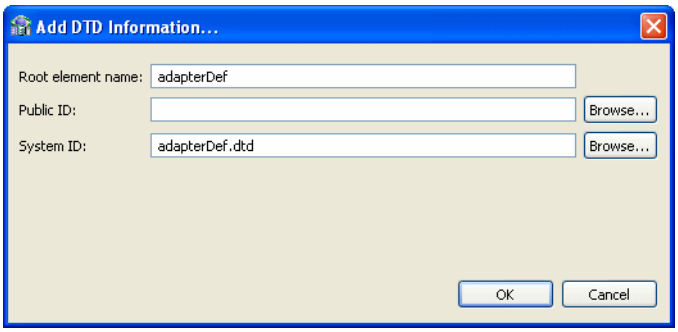
E.2.2 Add DTD Information

You can add DTD information to an element or attribute.

To add DTD Information

1. Right-click an element or attribute and select **Add DTD Information**. The Add DTD Information dialog box opens.

Figure E–2 Add DTD Information Dialog Box



2. Enter the information requested in the dialog box. The following table describes the Add DTD Information dialog box.

Table E–1 Add DTD Information

Field	Description
Root element name	The name of the XML root element.
Public ID	The value in this field is the Public Identifier. It is used to associate the XML file (using an XML catalog entry) with a DTD file by providing a hint to the XML processor. Click **Browse** to select an XML catalog entry from a list. An XML Catalog entry contains two parts, a Key (which represents a DTD or XML schema) and a URI (which contains information about a DTD or XML schema's location). Select the catalog entry you want to associate with your XML file.
System ID	The value in this field is the DTD the XML file is associated with. You can change the DTD the file is associated with by editing this field. The XML processor tries to use the Public ID to locate the DTD, and if this fails, it uses the System ID to find it. Click **Browse** to select a system ID. You can this in two ways: ■ Select the file from the workbench. In this case, update the with the import dialog box. ■ Select an XML catalog entry.

3. Save the file, then select it again in the Project Explorer and press F5 to refresh.

   The XML file is updated automatically.

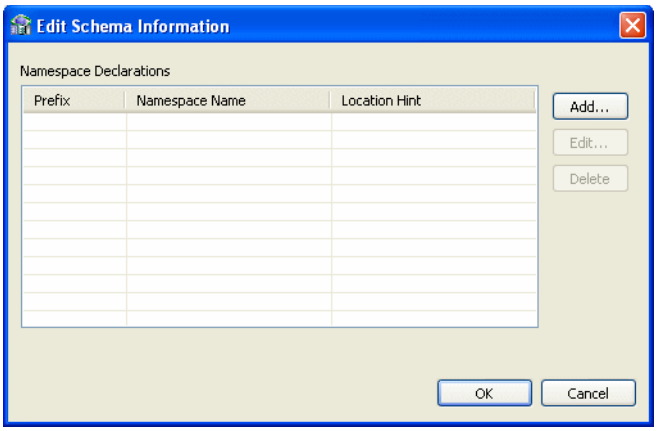
E.2.3 Edit Namespaces

You can make changes to the namespaces associated with an element or attribute.

To edit namespaces

1. Right-click an element or attribute and select **Edit namespaces**. The Edit Schema Information dialog box opens.

Figure E–3 Edit Schema Information



2. Click a button to make any changes to this information.

To add a new namespace

1. From the Schema Information dialog box, click **Add**.
2. The Add Namespace Definitions dialog box opens. Select one of the following:
   - **Select from registered namespaces**. This selection is available when the dialog box opens. Select from the list of registered namespaces and then click OK. If no registered namespaces are available, the list is empty.
   - **Specify new namespace**. Enter the information described in the following table:

Table E–2 New Namespace

Field	Description
Prefix	The prefix is added to all qualified elements and attributes in the XML file.
Namespace Name	The namespace of the XML file.
Location Hint	The location of the XML schema of the XML file. An XML Catalog ID or a URI can be entered in this field. Click **Browse** to search for the schema you want You can this in two ways: ▪ Select the schema from the workbench. In this case, update the with the import dialog box. ▪ Select an XML catalog entry. The **Namespace Name** and **Prefix** fields are be filled with the appropriate values from the schema (you must leave the fields blank for this to occur). **Note**: If you are creating an XML file from an XML schema, you cannot change the Namespace Name or Location Hint values.

To edit a namespace

1. From the Schema Information dialog box, click **Edit**.
2. Enter the information in the fields.

E.2.4 Add Elements and Attributes

You can add additional elements and attributes to the XML file.

To add Elements and Attributes

1. Right-click an element.
2. Select one of the following:
   - **Add Attribute** to add an attribute under the selected element.
   - **Add Child** to add another element under the selected element
   - **Add Before** to add another element above the selected element
   - **Add After** to add another element below the selected element

   Note: The Oracle Studio XML editor is Context sensitive to Oracle schemas. In this case, when adding elements and attributes to an XML file with an Oracle schema, you can select an element or attribute from a list of the possible values (depending on the schema definition). This list is available as a submenu.
3. Provide a name for the element or attribute if required. You may also select the element from a submenu. The element or attribute is added to the file.
4. Save the file, then select it again in the Project Explorer and press F5 to refresh.

   The XML file is updated automatically.

E.2.5 Replace an Element

You can replace an element with another legal element.

To replace an element

1. Right-click an element from the list in the editor.
2. Select **Replace with**.
3. Select an element from the submenu. Only legal elements are available.
4. The original element is replaced with the selected element.

Index

A

ACX trace parameter, 8-7
add timestamp to traced events parameter, 8-7

B

binary XML log level parameter, 8-7

C

cache buffer size parameter, 8-8
CICS adapter configuration, 3-2
CICS queue adapter configuration, 3-3
code page parameter, 8-9
COM maximum XML size in memory parameter, 8-10
COM maximum XML size parameter, 8-10
COM XML transport buffer parameter, 8-10
commit on destroy, 8-10
configuring JCA configuration files for inbound, 4-3
configuring the CICS adapter for outbound, 3-2
configuring the CICS queue adapter for inbound, 3-3
configuring the JCA configuration files for outbound, 4-2
creating JCA configuration files, 4-1

D

daemon
- logging, B-3
- security, B-5
- server modes, 8-2
- shutting down, 7-2
- starting, 7-1
- timeout, 7-6

data types
- atomic metadata, C-1
- NLS string parameter, 8-9

debug parameters
- ACX trace, 8-7
- add timestamp to traced events, 8-7
- environment, 8-7
- GDB trace, 8-7
- general trace, 8-7
- lbinary XML log level, 8-7
- log file, 8-7
- optimizer trace, 8-8
- query processor trace, 8-7
- query warnings, 8-7
- trace directory, 8-7
- Transaction extended logging, 8-8

disable 2PC, 8-10

E

environment parameters
- ACX trace, 8-7
- add timestamp to traced events, 8-7
- cache buffer size, 8-8
- code page, 8-9
- COM maximum XML size, 8-10
- COM maximum XML size in memory, 8-10
- COM XML transport buffer, 8-10
- commit on destroy, 8-10
- debug, 8-7
- disable 2PC, 8-10
- GDB trace, 8-7
- general trace, 8-7
- language, 8-9
- log file, 8-7
- miscellaneous, 8-8
- nav utility editor, 8-8
- NLS string, 8-9
- odbc, 8-9
- oledb, 8-9
- optimizer, 8-9
- optimizer trace, 8-8
- parallel processing, 8-9
- query processor trace, 8-7
- query warnings, 8-7
- queryProcessor, 8-9
- recovery delay, 8-10
- Replace invalid XML characters, 8-11
- time limit, 8-10
- trace directory, 8-7
- transaction conversions parameters, 8-10
- Transaction extended logging, 8-8
- transaction extended logging, 8-9
- transaction log file, 8-10
- transactions, 8-9
- ttemporary directory, 8-8

tuning, 8-10
user commit confirm table, 8-10
XML, 8-10
XML date format parameter, 8-11
XML trim char column, 8-11
year 2000 policy, 8-8
error log, binary XML log level parameter, 8-7
error log, log file parameter, 8-7

G

GDB trace parameter, 8-7
general parameters
cache buffer size, 8-8
nav utility editor, 8-8
temporary directory, 8-8
year 2000 policy, 8-8
general trace parameter, 8-7
generating interactions
inbound, 2-14
outbound, 2-27

I

inbound configuration, 3-3
inbound interactions, generating, 2-14
interactions
inbound, 2-14
outbound, 2-27

J

JCA configuration files
configuring inbound interactions, 4-3
configuring outbound interations, 4-2
creating, 4-1

L

language parameter, 8-9
language parameters
code page, 8-9
environment, 8-9
language, 8-9
NLS string, 8-9
log file parameter, 8-7
log files
binary XML log level parameter, 8-7
daemon options, B-3
log file parameter, 8-7
logging
daemon configurations, B-3
optimizer strategy, 8-8

M

metadata
atomic data types, C-1
miscellaneous parameters
environment, 8-8

N

nav utility editor parameter, 8-8
nav utility editor, text editor, 8-8
NLS string parameter, 8-9

O

odbc environment parameters, 8-9
oledb environment parameters, 8-9
optimizer
environment parameters, 8-9
trace directory parameter, 8-7
optimizer trace parameter, 8-8
outbound configuration, 3-2
outbound interactions, generating, 2-27

P

parallel processing environment parameters, 8-9

Q

query optimizer
logging strategy, 8-8
trace directory parameter, 8-7
query processor trace parameter, 8-7
query warnings parameter, 8-7
queryProcessor environment parameters, 8-9

R

recovery delay, 8-10
Replace invalid XML characters parameter, 8-11

S

security
daemon configurations, B-5
servers
configuring modes, 8-2
reusable, 8-3
Reuse limit daemon parameter, 8-3, B-13

T

temporary directory parameter, 8-8
temporary files, 8-8
time limit parameter, 8-10
timeout
client idle, 7-6
daemon, 7-6
trace directory parameter, 8-7
trace information, logging, 8-7
transaction conversions parameters, 8-10
transaction extended logging, 8-9
Transaction extended logging parameter, 8-8
transaction log file, 8-10
transactions environment parameters, 8-9
transactions parameters
commit on destroy, 8-10

disable 2PC, 8-10
recovery delay, 8-10
time limit, 8-10
transaction conversions parameters, 8-10
transaction extended logging, 8-9
transaction log file, 8-10
user commit confirm table, 8-10
tuning environment parameters, 8-10

U

user commit confirm table, 8-10

W

WebLogic
integration, 3-1
WebLogic integration, 3-1
Workspace server mode, B-13

X

XML
environment parameters, 8-10
XML date format parameter, 8-11
XML parameters
COM maximum XML size, 8-10
COM maximum XML size in memory, 8-10
COM XML transport buffer, 8-10
Replace invalid XML characters, 8-11
XML date format parameter, 8-11
XML trim char column, 8-11
XML trim char column parameter, 8-11

Y

Y2K
See year 2000 policy parameter
year 2000 policy parameter, 8-8