Oracle® Application Server Integration

Adapter for Tuxedo Installation and User's Guide

10*g* (9.0.4)

Part No. B10459-01

September 2003

Oracle Application Server Integration Adapter for Tuxedo Installation and User’s Guide, 10*g* (9.0.4)

Part No. B10459-01

Primary Author: Julian Weiss

Contributors: Seshu Adunuthula, Deanna Bradshaw, Yishai Hadas, Dror Harari, Arvind Jain, Maneesh Joshi, Mark Kennedy, Adeeb Massad, Bo Stern, Costi Zaboura

Contents

2 Installing the Oracle Application Server Integration Adapter for Tuxedo

3 Modeling Interactions for Oracle Application Server Integration Adapter for Tuxedo

4 Using Interactions in a Business Process

5 Oracle Application Server Integration Adapter for Tuxedo Features and Restrictions

6 Troubleshooting the Oracle Application Server Integration Adapter for Tuxedo

7 Configuring the Working Environment

A The Oracle Connect Tuxedo Interaction Metadata Definitions

B Datatype Support

C Daemon Configuration Settings

D National Language Settings

Index

List of Tables

Send Us Your Comments

Oracle Application Server Integration Adapter for Tuxedo Installation and User's Guide, 10*g* (9.0.4)

Part No. B10459-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: appserverdocs_us@oracle.com
- FAX: (650) 506-7407 Attn: Oracle Application Server Documentation Manager
- Postal service:
  Oracle Corporation
  Oracle Application Server Documentation
  500 Oracle Parkway, Mailstop 2op3
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

This guide is the primary source of user and reference information on the Oracle Application Server Integration Adapter for Tuxedo, which enables client applications to access programs running under Tuxedo through the Sun 2JEE Connector Architecture (JCA) API.

This document describes the features of Oracle Application Server Integration Adapter for TuxedoS software that apply to the HP Tru64 UNIX, HP-UX, IBM AIX, Solaris Operating System (SPARC), Windows NT, Windows 2000, Windows XP, and Windows Server 2003 operating systems.

This preface covers the following topics:

- Audience
- Organization
- Related Documentation
- Conventions
- Documentation Accessibility

Audience

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

- Installing and configuring the Oracle Application Server Integration Adapter for Tuxedo
- Diagnosing errors
- Using Oracle Application Server ProcessConnect to access Tuxedo programs

> **Note:** You should understand the fundamentals of Oracle Application Server ProcessConnect, the platform where the Tuxedo workstation runs (such as, the Solaris Operating System (SPARC) platform) and Microsoft Windows operating system before using this guide to install or administer the Oracle Application Server Integration Adapter for Tuxedo.

Organization

This document contains:

Chapter 1, "Overview of the Oracle Application Server Integration Adapter for Tuxedo"

This chapter introduces the Oracle Application Server Integration Adapter for Tuxedo.

Chapter 2, "Installing the Oracle Application Server Integration Adapter for Tuxedo"

This chapter describes how to install Oracle Connect for Tuxedo on the platform where the Tuxedo workstation runs and Oracle Studio on a Windows platform.

Chapter 3, "Modeling Interactions for Oracle Application Server Integration Adapter for Tuxedo"

This chapter describes how to model interactions in Oracle Studio.

Chapter 4, "Using Interactions in a Business Process"

This chapter describes how to include Oracle Connect for Tuxedo interactions in a business process in Oracle Application Server ProcessConnect.

Chapter 5, "Oracle Application Server Integration Adapter for Tuxedo Features and Restrictions"

This chapter describes specific features of the Oracle Application Server Integration Adapter for Tuxedo.

Chapter 6, "Troubleshooting the Oracle Application Server Integration Adapter for Tuxedo"

This chapter describes how to troubleshoot problems both during modeling and runtime.

Appendix A, "The Oracle Connect Tuxedo Interaction Metadata Definitions"

This appendix describes the metadata for Oracle Connect for Tuxedo.

Appendix B, "Datatype Support"

This appendix describes how Tuxedo datatypes are converted to Oracle Connect for Tuxedo datatypes and Oracle datatypes.

Appendix C, "Daemon Configuration Settings"

This appendix describes daemon configuration used to control client/server communication.

Related Documentation

For more information, see these Oracle resources:

- *Oracle Application Server ProcessConnect User's Guide*

Printed documentation is available for sale in the Oracle Store at

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

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

`http://otn.oracle.com/membership/`

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

`http://otn.oracle.com/documentation/`

Conventions

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

- Conventions in Text
- Conventions in Code Examples
- Conventions for Windows Operating Systems

Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
Bold	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an **index-organized table**.
Italics	Italic typeface indicates book titles or emphasis.	*Oracle9i Database Concepts* Ensure that the recovery catalog and target database do *not* reside on the same disk.
`UPPERCASE monospace (fixed-width) font`	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, usernames, and roles.	You can specify this clause only for a `NUMBER` column. You can back up the database by using the `BACKUP` command. Query the `TABLE_NAME` column in the `USER_TABLES` data dictionary view. Use the `DBMS_STATS.GENERATE_STATS` procedure.

Convention	Meaning	Example
lowercase monospace (fixed-width) font	Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values. **Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter `sqlplus` to open SQL*Plus. The password is specified in the `orapwd` file. Back up the datafiles and control files in the `/disk1/oracle/dbs` directory. The `department_id`, `department_name`, and `location_id` columns are in the `hr.departments` table. Set the `QUERY_REWRITE_ENABLED` initialization parameter to `true.` Connect as `oe` user. The `JRepUtil` class implements these methods.
lowercase italic monospace (fixed-width) font	Lowercase italic monospace font represents placeholders or variables.	You can specify the *parallel_clause*. Run U*old_release*.SQL where *old_release* refers to the release you installed prior to upgrading.

Conventions in Code Examples

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

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

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

Convention	Meaning	Example
[ ]	Brackets enclose one or more optional items. Do not enter the brackets.	DECIMAL (*digits* [, *precision*])
{ }	Braces enclose two or more items, one of which is required. Do not enter the braces.	{ENABLE | DISABLE}
|	A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar.	{ENABLE | DISABLE} [COMPRESS | NOCOMPRESS]

Convention	Meaning	Example

. . .	Horizontal ellipsis points indicate either:
	■ That we have omitted parts of the code that are not directly related to the example
	■ That you can repeat a portion of the code
		CREATE TABLE ... AS *subquery*;
		SELECT *col1*, *col2*, ... , *coln* FROM employees;

. . .	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	SQL> SELECT NAME FROM V$DATAFILE; NAME ------------------------------------ /fsl/dbs/tbs_01.dbf /fs1/dbs/tbs_02.dbf . . . /fsl/dbs/tbs_09.dbf 9 rows selected.

Other notation	You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown.	acctbal NUMBER(11,2);
		acct CONSTANT NUMBER(4) := 3;

Italics	Italicized text indicates placeholders or variables for which you must supply particular values.	CONNECT SYSTEM/*system_password* DB_NAME = *database_name*

UPPERCASE	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. However, because these terms are not case sensitive, you can enter them in lowercase.	SELECT last_name, employee_id FROM employees;
		SELECT * FROM USER_TABLES;
		DROP TABLE hr.employees;

lowercase	Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files.
	Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.
		SELECT last_name, employee_id FROM employees;
		sqlplus hr/hr
		CREATE USER mjones IDENTIFIED BY ty3MU9;

Conventions for Windows Operating Systems

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

Convention	Meaning	Example
Choose Start >	How to start a program.	To start the Database Configuration Assistant, choose Start > Programs > Oracle - *HOME_NAME* > Configuration and Migration Tools > Database Configuration Assistant.

File and directory names	File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (<), right angle bracket (>), colon (:), double quotation marks ("), slash (/), pipe (|), and dash (-). The special character backslash (\\) is treated as an element separator, even when it appears in quotes. If the file name begins with \\\\, then Windows assumes it uses the Universal Naming Convention.	`c:\winnt"\"system32 is the same as`
		`C:\WINNT\SYSTEM32`

`C:\>`	Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the *command prompt* in this manual.	`C:\oracle\oradata>`

Special characters	The backslash (\\) special character is sometimes required as an escape character for the double quotation mark (") special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.	`C:\>exp scott/tiger TABLES=emp QUERY=\"WHERE job='SALESMAN' and sal<1600\"`
		`C:\>imp SYSTEM/`*`password`*` FROMUSER=scott TABLES=(emp, dept)`

HOME_NAME	Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.	`C:\> net start Oracle`*`HOME_NAME`*`TNSListener`

Convention	Meaning	Example
ORACLE_HOME and *ORACLE_BASE*	In releases prior to Oracle8*i* release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level *ORACLE_HOME* directory. For Windows NT, the default location was `C:\orant`. This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level *ORACLE_HOME* directory. There is a top level directory called *ORACLE_BASE* that by default is `C:\oracle`. If you install the latest Oracle release on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is `C:\oracle\ora`*nn*, where *nn* is the latest release number. The Oracle home directory is located directly under *ORACLE_BASE*. All directory path examples in this guide follow OFA conventions. Refer to *Oracle9i Database Getting Started for Windows* for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.	Go to the *ORACLE_BASE*`\`*ORACLE_HOME*`\rdbms\admin` directory.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation 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 additional information, visit the Oracle Accessibility Program Web site at

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

Accessibility of Code Examples in Documentation JAWS, a Windows screen reader, 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, JAWS 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 Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.

1

Overview of the Oracle Application Server Integration Adapter for Tuxedo

This chapter introduces the Oracle Application Server Integration Adapter for Tuxedo. The Oracle Application Server Integration Adapter for Tuxedo is a plug-in component of Oracle Application Server ProcessConnect, enabling Oracle Application Server ProcessConnect to access Tuxedo services on the platform where the Tuxedo workstation runs.

This chapter contains the following sections:

- Overview of Application Integration
- Overview of Tuxedo and Oracle Application Server Integration Adapter for Tuxedo
- Integrating Oracle Application Server ProcessConnect with the Adapter for Tuxedo
- Modeling the Tuxedo Adapter Using Oracle Studio

Overview of Application Integration

When business applications need to communicate with each other, the output from one application is not always understandable by the other application. Application adapters enable different applications to interoperate.

Adapters for application integration address a mismatch between the integrating interface of a source or target application and the integration infrastructure, such as Oracle Application Server ProcessConnect. Simple adapters only exchange data using an API, but complex adapters can also do transformations, as well as deal with exception conditions. Therefore, adapters are a combination of modeling tools and runtime software that act as "glue" to link applications, considered as sources or targets (or both), to the integration infrastructure that transports, translates, transforms, and routes data between systems. On the one hand, an adapter deals with a group of touchpoints (one or more entry/exit points, collectively an interface) for a source or target; on the other hand, an adapter links to the integration infrastructure. This means that adapters generally have two interfaces - one that is native to the source or target application and the other that is native to the integrating infrastructure. Adapters perform a variety of key tasks, including recognizing events, collecting and transforming data, and exchanging data with the integration infrastructure. They also handle exception conditions, and can often dynamically (or with minor reconfiguration changes) accommodate new revisions of back-end applications.

Each task performed by the adapter is defined as an interaction. Interactions that are initiated by the source or target application are referred to as inbound interactions. Interactions that are initiated by the integrating infrastructure are referred to as outbound interactions.

Overview of Tuxedo and Oracle Application Server Integration Adapter for Tuxedo

The Oracle Application Server Integration Adapter for Tuxedo is a plug-in component of Oracle Application Server ProcessConnect.

The Oracle Application Server Integration Adapter for Tuxedo models services running under BEA Tuxedo. The adapter is native to the platform where the Tuxedo workstation runs, providing direct access that results in improved performance.

Tuxedo

Tuxedo is an application server from BEA Systems, Inc. The Tuxedo system provides an API called ATMI (Application to Transaction Manager Interface) to interact with back-end Tuxedo servers. Tuxedo defines a Service concept that is equivalent to the Oracle Application Server ProcessConnect interaction concept. Services are written in 3GL programs which are registered with Tuxedo.

In Oracle Application Server Integration Adapter for Tuxedo, Tuxedo is modeled as a collection of interactions, where each outbound interaction is mapped to a specific Tuxedo service along with its particular input record and output record, corresponding to the structure of the communication area used for communicating with the service.

Tuxedo defines a metadata language to describe the inputs and outputs of a service. The Oracle Application Server Integration Adapter for Tuxedo captures and maintains a metadata schema for Tuxedo by importing Tuxedo metadata and transforming it into mapping definitions for Oracle Connect for Tuxedo.

The Oracle Application Server Integration Adapter for Tuxedo 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. Moreover, the Oracle Application Server Integration Adapter for Tuxedo maps data structures, facilitating access to Tuxedo services from within Oracle Application Server ProcessConnect.

Architecture of the Oracle Application Server Integration Adapter for Tuxedo

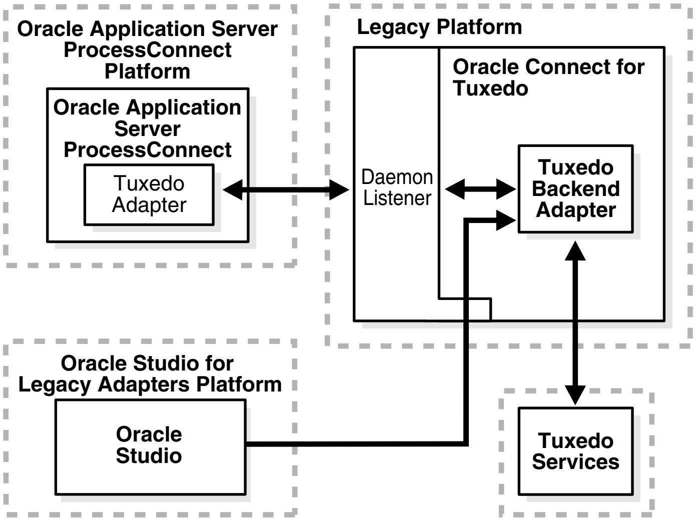
The Oracle Application Server Integration Adapter for Tuxedo includes the following components:

- The Tuxedo adapter, which is part of Oracle Application Server ProcessConnect.
- Oracle Connect for Tuxedo, which resides on the platform where the Tuxedo workstation runs.
- A GUI-based configuration tool, called Oracle Studio for Legacy Adapters, which resides on a Windows platform.

Note: Throughout this manual Oracle Studio for Legacy Adapters is referred to as Oracle Studio.

The components of the Oracle Application Server Integration Adapter for Tuxedo interact with each other, as shown in Figure 1–1 on page 1-4.

Figure 1–1 Architecture of the Oracle Application Server Integration Adapter for Tuxedo


Oracle Application Server ProcessConnect Platform
Oracle Application Server ProcessConnect
Tuxedo Adapter
Legacy Platform
Oracle Connect for Tuxedo
Daemon Listener
Tuxedo Backend Adapter
Oracle Studio for Legacy Adapters Platform
Oracle Studio
Tuxedo Services

The Tuxedo adapter enables communication with Oracle Connect for Tuxedo on the platform where the Tuxedo workstation runs.

Oracle Connect for Tuxedo is comprised of two main components

- **Daemon (a listener)** - The daemon is responsible for managing and allocating server processes for client requests coming from Oracle Application Server ProcessConnect. Server processes can be started per client or kept in a pool for reuse.

  > **See Also:** "Troubleshooting the Daemon" on page 6-4 and Appendix C, "Daemon Configuration Settings" for details about the daemon

- **Tuxedo backend adapter** - Provides the interaction definitions and input and output record schemas required to access a Tuxedo service

Oracle Studio is used to model the adapter interactions and manage the back-end component.

Inbound Interactions

Tuxedo services do not have a built-in model for inbound services. Applications that run under Tuxedo can interact with Oracle Application Server ProcessConnect by sending it a native event. The native event is sent to a queue running on the platform where the Tuxedo workstation runs and the adapter subscribes to that queue for inbound event handling. Typically, each application associated with Oracle Connect for Tuxedo will subscribe to a dedicated queue.

The Oracle Application Server Integration Adapter for Tuxedo provides an API which can be called from a C program to send an inbound interaction.

Integrating Oracle Application Server ProcessConnect with the Adapter for Tuxedo

Oracle Application Server Integration Adapter for Tuxedo is fully integrated with Oracle Application Server ProcessConnect, both during the modeling and runtime phases.

The Modeling Phase

The Oracle Application Server Integration Adapter for Tuxedo is integrated with Oracle Application Server ProcessConnect as follows:

- Modeling the Oracle Application Server Integration Adapter for Tuxedo is performed using Oracle Studio. This step includes modeling inbound and outbound interactions as required.

  See Also: "Modeling the Tuxedo Adapter Using Oracle Studio" on page 1-6

- After the Oracle Application Server Integration Adapter for Tuxedo has been modeled in Oracle Studio, the integration between the adapter and the application in Oracle Application Server ProcessConnect is modeled. Within Oracle Application Server ProcessConnect you define a delivery channel for the adapter and then add interactions from the list modeled using Oracle Studio.

  See Also: *Oracle Application Server ProcessConnect User's Guide*

Runtime

At runtime, the interactions specified in the Oracle Application Server ProcessConnect application are executed.

For outbound interactions, the interactions for the Oracle Application Server Integration Adapter for Tuxedo are handled by Oracle Connect for Tuxedo on the platform where the Tuxedo workstation runs. The Tuxedo service specified in the interaction is executed and any output returned to Oracle Application Server ProcessConnect.

For inbound interactions, the interactions are initiated on the platform where the Tuxedo workstation runs and sent to Oracle Application Server ProcessConnect.

Modeling the Tuxedo Adapter Using Oracle Studio

Oracle Studio is used to model and manage the Oracle Application Server Integration Adapter for Tuxedo.

Oracle Studio uses perspectives enabling you to focus specific information necessary to model the Oracle Application Server Integration Adapter for Tuxedo.

The following perspectives are available:

- The Configuration Perspective
- The Metadata Perspective
- The Metadata Import Perspective
- The Runtime Manager Perspective

The Configuration Perspective

Use the Configuration perspective to define access to Tuxedo services via the Tuxedo workstation machine (such as, the Solaris Operating System (SPARC) platform).

The Metadata Perspective

Use the Metadata perspective to model the Oracle Application Server Integration Adapter for Tuxedo interactions and the expected inputs and output.

The Metadata Import Perspective

Use the Metadata Import perspective to import information to create outbound interactions for Oracle Connect for Tuxedo based on existing Tuxedo services.

The Runtime Manager Perspective

Use the Runtime Manager perspective to troubleshoot Oracle Connect for Tuxedo by monitoring daemon activity on the platform where the Tuxedo workstation runs.

2

Installing the Oracle Application Server Integration Adapter for Tuxedo

This chapter explains how to install Oracle Connect for Tuxedo and Oracle Studio software from the distribution CD-ROM.

> **Note:** In addition to the installation procedures described in this chapter, the Tuxedo adapter component of the Oracle Application Server Integration Adapter for Tuxedo software must be installed with Oracle Application Server ProcessConnect. Installing the Tuxedo adapter component is described in the *Oracle Application Server ProcessConnect Installation Guide*.

This chapter includes the following sections:

- Before You Begin
- Installing Oracle Connect for Tuxedo on a UNIX Platform
- Installing Oracle Connect for Tuxedo on a Windows Platform
- Installing Oracle Studio
- Configuring Oracle Connect for Tuxedo
- Securing Access to Oracle Connect for Tuxedo

Before You Begin

Confirm that you have met all hardware and software requirements. Refer to the following sections:

- Determining the UNIX Hardware and Software Requirements
- Determining the Hardware and Software Requirements for Windows

Determining the UNIX Hardware and Software Requirements

This section describes the following requirements for installing Oracle Connect for Tuxedo:

- Hardware Requirements
- Software Requirements

Hardware Requirements

The following table summarizes the hardware requirements for Oracle Connect for Tuxedo.

Hardware Component	Requirements
Processor	An HP Alpha workstation running the required version of HP Tru64 UNIX Or An HP-UX workstation running the required version of HP-UX Or An IBM AIX workstation running the required version of IBM AIX Or A Sun Solaris workstation running the required version of Solaris Operating System (SPARC)
CD-ROM Drive	An internal or external CD-ROM drive
Disk Space	50MB of free disk space

Software Requirements

The system software configuration described in this section is supported by Oracle Corporation as long as the underlying system software products are supported by their respective vendors. Verify the latest support status with your system software vendors.

The following table summarizes the software requirements for Oracle Connect for Tuxedo.

Software Component	Requirements
Operating System	HP Tru64 UNIX version 5.1 and up Or HP-UX version 11 and up. **Note**: The operating system must run in its 32 bit mode. Or IBM AIX version 4.33 and up. **Note**: The operating system must run in its 32 bit mode. Or Solaris Operating System (SPARC) version 2.8 and up. **Note**: The operating system must run in its 32 bit mode.
BEA Tuxedo	Version 8.0 and up
Oracle Application Server ProcessConnect	Oracle Application Server 10*g* (9.0.4) Oracle Application Server ProcessConnect can reside on any supported platform

Determining the Hardware and Software Requirements for Windows

This section describes the following requirements for installing Oracle Studio or Oracle Connect for Tuxedo when the Tuxedo workstation runs on the Windows platform:

- Hardware Requirements
- Software Requirements

Hardware Requirements

The following table summarizes the hardware requirements for Oracle Studio or Oracle Connect for Tuxedo when the Tuxedo workstation runs on the Windows platform.

Hardware Component	Requirements
Processor	An Intel or 100% compatible personal computer (PC), based on a Pentium processor
Memory	128 MB of RAM
CD-ROM Drive	An internal or external CD-ROM drive
Disk Space for Oracle Studio	75MB of free disk space
Disk Space for Oracle Connect for Tuxedo	30MB of free disk space

Software Requirements

The system software configuration described in this section is supported by Oracle Corporation as long as the underlying system software products are supported by their respective vendors. Verify the latest support status with your system software vendors.

The following table summarizes the software requirements for Oracle Studio or Oracle Connect for Tuxedo when the Tuxedo workstation runs on the Windows platform.

Software Component	Requirements
Operating System	Microsoft Windows NT Server Version 4.0 with service pack 5 or higher, Microsoft Windows 2000 with service pack 2 or higher, or Microsoft Windows XP
Microsoft	Network transport protocol software, TCP/IP, included with Microsoft Windows

Installing Oracle Connect for Tuxedo on a UNIX Platform

This section explains how to install Oracle Connect for Tuxedo software on a UNIX platform from the distribution CD-ROM. This section includes the following:

- Preinstallation Instructions
- Installing the Kit

- Installation Instructions
- Postinstallation Instructions

Preinstallation Instructions

Before starting the installation, make sure that you have the following information available:

- The root directory where Oracle Connect for Tuxedo software is to be installed.

> **Note:** The root directory cannot be a system root directory or /var or /tmp.

- The account name where Oracle Connect for Tuxedo will run.

Oracle Connect for Tuxedo software is contained in the following operating system dependent files:

- To run under HP Tru64 UNIX: `ocl904-alphaunx.taz`
- To run under HP-UX: `ocl904-hpux.taz`
- To run under IBM AIX: `ocl904-ibmaix.taz`
- To run under Solaris Operating System (SPARC): `ocl904-sunsol.taz`

These files are provided on a CD-ROM in the directory `Oracle_Connect\Tuxedo_Legacy_Adapter`.

Installing the Kit

1. Using FTP, copy the relevant Oracle Connect for Tuxedo software installation file in binary mode from the installation CD to the UNIX machine.
2. On the UNIX machine, rename the taz file, as in the following example:

   ```
   mv ocl904-sunsol.taz ocl904-sunsol.tar.Z
   ```

3. Decompress the file, as in the following example:

   ```
   uncompress ocl904-sunsol.tar.Z
   ```

4. Run the tar command, as in the following example:

   ```
   tar xvf ocl904-sunsol.tar nav_install
   ```

A message similar to the following is returned:

```
x nav_install, nnnn bytes, mmmm tape blocks
```

5. Make sure the directory used to run the installation has write privileges.

Installation Instructions

Complete the following steps to install Oracle Connect for Tuxedo:

1. Run the following command:

   ```
   ./nav_install
   ```

   Follow the instructions in the Response column in Table 2–1 for each entry in the Screen column.

Table 2–1 Installation Prompts and Responses

Screen	Response
ORA01A: Please enter the name of the .tar file containing the Oracle Connect software [./*tarfile*]:	Enter the full path of the disk archive file
ORA02A: The Oracle Connect software will be installed in a fixed directory (named "navroot"), under the root directory of your choice. You need at least 16000 kbytes of free disk space to install the software. Please enter the root directory name [/*home*/*username*]:	Enter the root directory for the installation. You must have write permission to this directory The default directory is the users home directory **Note**: The root directory cannot be a system root directory or `/var` or `/tmp`
The Oracle Connect software installation directory (NAVROOT): /*home*/*username*/navroot Please confirm (Yes/No/Quit) [Yes]:	Confirm the root directory for the installation
ORA04A: Supply the account name where you want Oracle Connect to run. This is the default that you can change after the installation. Please enter the account name [*username*]:	The account name used for anonymous access by clients to the server

Table 2–1 (Cont.) Installation Prompts and Responses

Screen	Response
The Oracle Connect server will be running as "*username*" Please confirm (Yes/No/Quit) [Yes]:	Confirm the account name for used for client access
ORA05A: Please choose the shell which you want the Oracle Connect server to run under: 1. C-shell (/bin/csh) 2. Korn-shell (/bin/ksh) 3. Bourne-shell (/bin/sh) Please enter your choice (1/2/3/Quit) [1]:	Enter the shell used at the site The installation creates a startup file based on the shell specified
ORA06A: Adding and updating Oracle Connect for Tuxedo's configuration on this machine, from a remote Oracle administration console, can only be done by someone defined as an administrator for Oracle Connect on this machine. Enter a valid user name for an Oracle Connect administrator [All]:	In order to manage Oracle Connect on this machine from Oracle Studio, you need to enter a user account of a user who will have administrative authorization, or press Enter to enable any user to administer Oracle Connect on this machine. The administrative rights can be changed from within Oracle Studio after the installation.

The installation completes.

Postinstallation Instructions

The following postinstallation tasks must be done to work with Oracle Connect for Tuxedo:

- Setting Up the Oracle Connect for Tuxedo Environment
- Setting the Tuxedo Environment for Oracle Connect for Tuxedo
- Setting the Oracle Connect for Tuxedo Script
- Starting the Oracle Connect for Tuxedo Daemon

Setting Up the Oracle Connect for Tuxedo Environment

When Oracle Connect for Tuxedo is installed on a UNIX platform, using FTP, copy the `brand.bin` file, in binary mode, from the `Oracle Connect\Tuxedo Legacy Adapter` directory in the installation CD to the Tuxedo workstation machine, to *`NAVROOT`*`/bin`.

NAVROOT is the directory where Oracle Connect for Tuxedo is installed.

Setting the Tuxedo Environment for Oracle Connect for Tuxedo

Check the following Tuxedo environment variables are correctly set:

TUXDIR is set to the Tuxedo root directory.

WSNADDR is set to the Tuxedo workstation network address.

Check the shared library environment variable (LD_LIBRARY_PATH, or SHLIB_PATH under HP-UX) includes the path to the Tuxedo lib directory, as in the following example:

```
LD_LIBRARY_PATH = /disk2/users/tuxedo/tuxedo8.0/lib
```

Setting the Oracle Connect for Tuxedo Script

The program that manages Oracle Connect for Tuxedo server processes (nav_server) is accessed by a symbolic link to a file for the C-shell, Bourne and Korn shells. To set up nav_server, perform the following procedure:

1. In the bin directory, under the directory where Oracle Connect for Tuxedo is installed, delete the existing link to nav_server as follows:

   ```
   rm nav_server
   ```

2. In the bin directory, under the directory where Oracle Connect for Tuxedo is installed, link to the appropriate version of nav_server as follows:

   C-shell: `ln -s nav_server.csh nav_server`

   Bourne: `ln -s nav_server.sh nav_server`

   Korn: `ln -s nav_server.ksh nav_server`

Note: Use a symbolic link as shown instead of renaming files.

Oracle Connect for Tuxedo’s nav_login procedure defines the default environment when the adapter runs. If you want site-dependent variables to be included in the environment, create a file called site_nav_login and save this file in the bin directory under the Oracle Connect for Tuxedo root directory. nav_login runs site_nav_login automatically.

nav_login must be invoked to run Oracle Connect for Tuxedo. It can be invoked from the user’s login script.

> **Note:** It is recommended to add the TUXDIR and WSNADDR environment variables to the site_nav_login file, so that they are always available when new server processes are started to handle client requests.

The command line for invoking `nav_login` varies according to the shell the user is running. The following table shows the different options:

Shell	nav_login Command
CSH	source *root*/bin/nav_login
Bourne	. *root*/bin/nav_login.sh
Korn	. *root*/bin/nav_login.sh

where *root* represents the root directory of the Oracle Connect for Tuxedo installation. After the login procedure is executed, the environment variable NAVROOT points to this root directory.

> **Note:** For the Bourne and Korn shells, nav_login.sh requires a period and space (as shown in the above table).

Make sure that users have read and execute permission on the Oracle Connect for Tuxedo files (if necessary, use the chmod command to change the permissions).

Starting the Oracle Connect for Tuxedo Daemon

The Oracle Connect for Tuxedo daemon must run on a server for client/server access to Oracle Connect for Tuxedo. To start the daemon at system boot time, add the following command invoking the daemon to the end of the `/etc/inittab` file.

```
nv:3:once:navroot/bin/irpcd -l ip:2552 start >/dev/console 2>&1
```

where the symbol *navroot* should be replaced with the directory where Oracle Connect for Tuxedo is installed and *ip* replaced by the ip address of the machine.

Note: To allow automatic client/server access to Oracle Connect for Tuxedo, start the daemon at system boot time from a super user account. Otherwise, make sure that all users have access to the account that started the daemon.

Installing Oracle Connect for Tuxedo on a Windows Platform

This section explains how to install Oracle Connect for Tuxedo software from the distribution CD-ROM.

Assuming that the CD-ROM drive is D:, the installation file is located in the `D:\Oracle_Connect\Tuxedo_Legacy_Adapter` directory. Install the software from the CD-ROM by running the installation batch file, `OCL904-win32.bat`, either using the Run option in the Windows Start menu or through Windows Explorer.

Caution: If you are installing Oracle Connect for Tuxedo on a Windows XP machine, you cannot specify a logical drive as the Destination folder for the installation.

Installing Oracle Studio

Note: If you have Oracle Studio already installed because you are also using another legacy adapter, you do not need to reinstall it.

The other legacy adapters are:

- Oracle Application Server Integration Adapter for CICS
- Oracle Application Server Integration Adapter for IMS/DB
- Oracle Application Server Integration Adapter for Tuxedo
- Oracle Application Server Integration Adapter for VSAM

This section explains how to install Oracle Studio software from the distribution CD-ROM.

Installing Oracle Studio from the CD-ROM

Assuming that the CD-ROM drive is D:, the installation file is located in the `D:\Oracle_Studio` directory. Install the software from the CD-ROM by running the self-extracting executable installation file, `OSL904-win32.exe`, either using the Run option in the Windows Start menu or through Windows Explorer.

> **Caution:** If you are installing Oracle Studio on a Windows XP machine, you cannot specify a logical drive as the Destination folder for the installation.

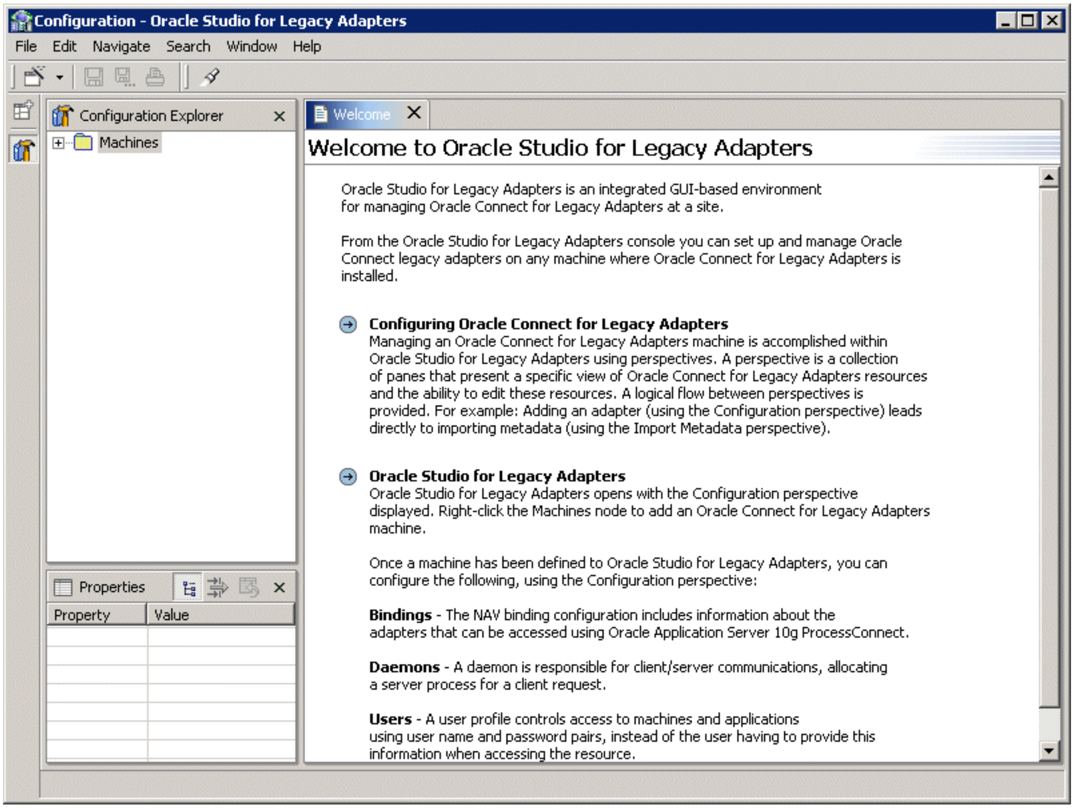
Configuring Oracle Connect for Tuxedo

All modeling of Oracle Connect for Tuxedo is performed using Oracle Studio. To use Oracle Studio you first configure it to enable access to the platform where the Tuxedo workstation runs.

> **Note:** The following tasks assume you have permission to access the platform where the Tuxedo workstation runs and that the Oracle Connect for Tuxedo daemon is running on this machine.
>
> Check with the system administrator to ensure these requirements are fulfilled.

Task: Setting Up the Tuxedo Workstation Machine

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**. Oracle Studio opens, displaying the Configuration perspective.


Configuration - Oracle Studio for Legacy Adapters
File Edit Navigate Search Window Help
Configuration Explorer
Machines
Welcome
Properties
Property
Value

Welcome to Oracle Studio for Legacy Adapters

Oracle Studio for Legacy Adapters is an integrated GUI-based environment for managing Oracle Connect for Legacy Adapters at a site.

From the Oracle Studio for Legacy Adapters console you can set up and manage Oracle Connect legacy adapters on any machine where Oracle Connect for Legacy Adapters is installed.

Configuring Oracle Connect for Legacy Adapters
Managing an Oracle Connect for Legacy Adapters machine is accomplished within Oracle Studio for Legacy Adapters using perspectives. A perspective is a collection of panes that present a specific view of Oracle Connect for Legacy Adapters resources and the ability to edit these resources. A logical flow between perspectives is provided. For example: Adding an adapter (using the Configuration perspective) leads directly to importing metadata (using the Import Metadata perspective).

Oracle Studio for Legacy Adapters
Oracle Studio for Legacy Adapters opens with the Configuration perspective displayed. Right-click the Machines node to add an Oracle Connect for Legacy Adapters machine.

Once a machine has been defined to Oracle Studio for Legacy Adapters, you can configure the following, using the Configuration perspective:

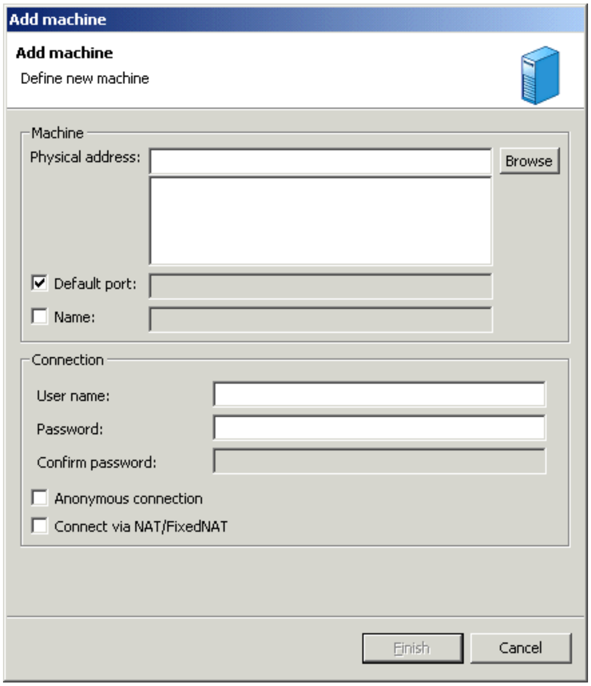
Bindings - The NAV binding configuration includes information about the adapters that can be accessed using Oracle Application Server 10g ProcessConnect.

Daemons - A daemon is responsible for client/server communications, allocating a server process for a client request.

Users - A user profile controls access to machines and applications using user name and password pairs, instead of the user having to provide this information when accessing the resource.

2. Right-click **Machines** in the **Configuration Explorer** and choose **Add Machine** from the popup menu.

   The Add Machine window appears.

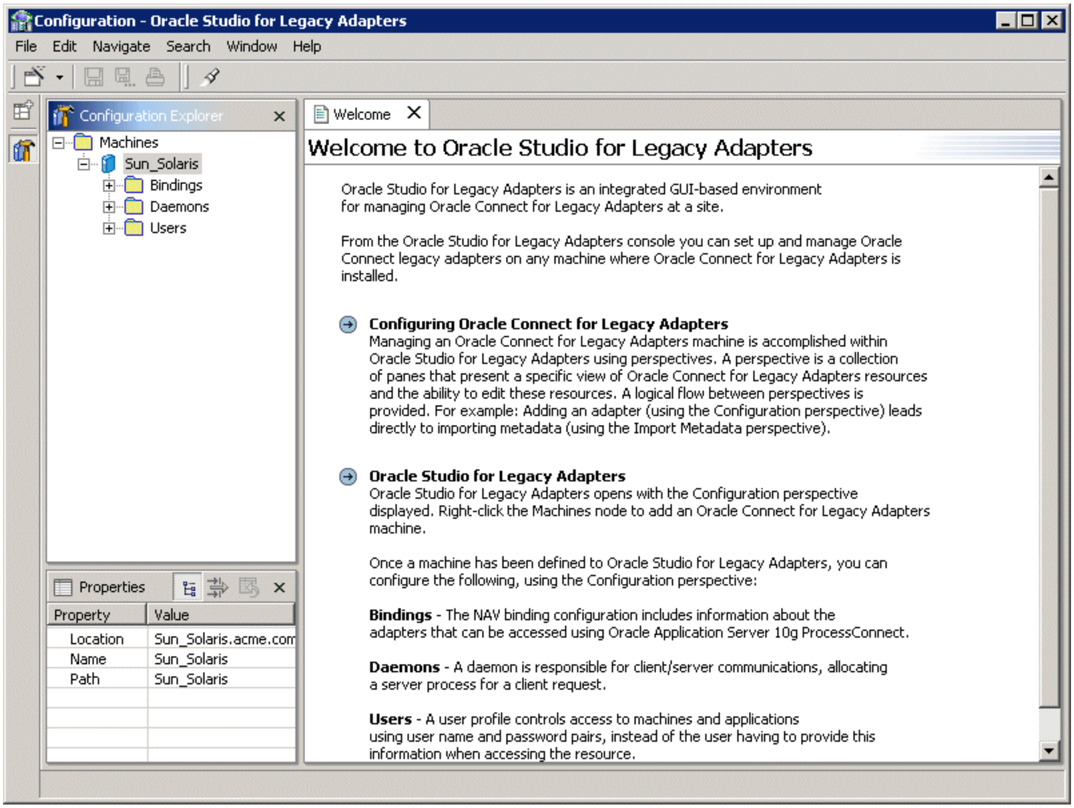


3. In the Add Machine window, uncheck the Default port check box and enter 2552 as the port number.
4. Enter the name of the machine you want to connect to, or click **Browse** to choose the machine from the list of machines that is displayed.
5. Specify the username and password of the user who was specified as the administrator when Oracle Connect for Tuxedo was installed.

> **Note:** Checking the **Anonymous connection** checkbox enables anyone having access to the machine to be an administrator, if this was defined for the machine.

6. Click **Finish**.

   The machine is displayed in the **Configuration Explorer**.



Property	Value
Location	Sun_Solaris.acme.com
Name	Sun_Solaris
Path	Sun_Solaris

Welcome to Oracle Studio for Legacy Adapters

Oracle Studio for Legacy Adapters is an integrated GUI-based environment for managing Oracle Connect for Legacy Adapters at a site.

From the Oracle Studio for Legacy Adapters console you can set up and manage Oracle Connect legacy adapters on any machine where Oracle Connect for Legacy Adapters is installed.

Configuring Oracle Connect for Legacy Adapters
Managing an Oracle Connect for Legacy Adapters machine is accomplished within Oracle Studio for Legacy Adapters using perspectives. A perspective is a collection of panes that present a specific view of Oracle Connect for Legacy Adapters resources and the ability to edit these resources. A logical flow between perspectives is provided. For example: Adding an adapter (using the Configuration perspective) leads directly to importing metadata (using the Import Metadata perspective).

Oracle Studio for Legacy Adapters
Oracle Studio for Legacy Adapters opens with the Configuration perspective displayed. Right-click the Machines node to add an Oracle Connect for Legacy Adapters machine.

Once a machine has been defined to Oracle Studio for Legacy Adapters, you can configure the following, using the Configuration perspective:

Bindings - The NAV binding configuration includes information about the adapters that can be accessed using Oracle Application Server 10g ProcessConnect.

Daemons - A daemon is responsible for client/server communications, allocating a server process for a client request.

Users - A user profile controls access to machines and applications using user name and password pairs, instead of the user having to provide this information when accessing the resource.

Securing Access to Oracle Connect for Tuxedo

Oracle Studio includes mechanisms to secure access to Oracle Connect for Tuxedo both during modeling and at runtime.

During modeling the following security mechanisms can be applied:

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

At runtime client access to Oracle Connect for Tuxedo is provided by the user profile:

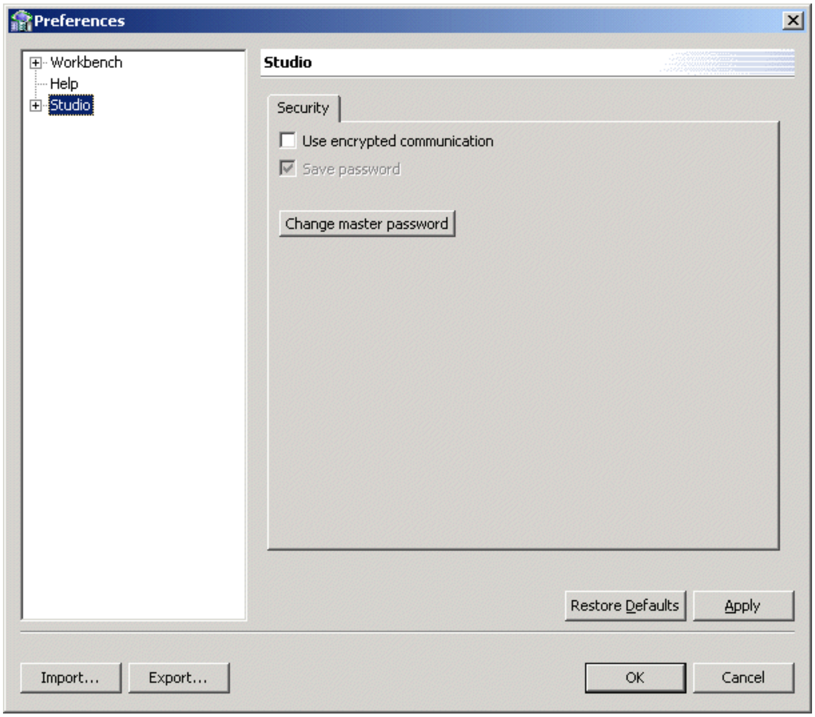
- Setting Up Runtime User Access to the Tuxedo Workstation Machine

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.

Task: Setting Up a Master Password to Access Oracle Studio

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**. Oracle Studio opens.
2. Choose **Window > Preferences** from the menu bar. to open the Preferences window.
3. Select the **Studio** node.



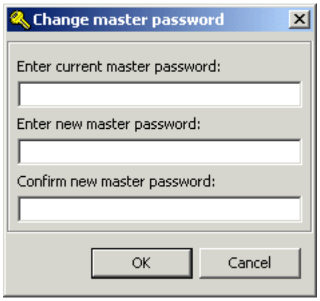
□ Use encrypted communication

☑ Save password

Note: The **Save password** checkbox is for future use.

---

4. Click **Change master password** to open the Change master password window.



Change master password

Enter current master password:

Enter new master password:

Confirm new master password:

5. Leave the **Enter current master password** field empty and type a new master password.
6. Confirm the password.
7. Click **OK**.

Specifying Users with Administrative Rights

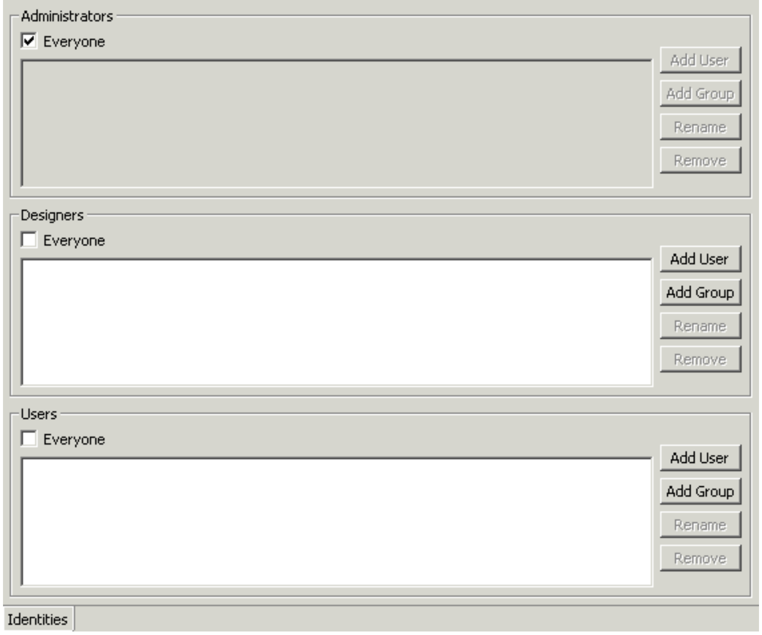
By default, only the user who was specified during the installation as an administrator has the authorization to modify settings on that machine from Oracle Studio. This user can then authorize other users to make changes or to view the definitions for a selected machine. Adding a machine to Oracle Studio is described in "Task: Setting Up the Tuxedo Workstation Machine" on page 2-11.

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

Task: Setting Administrative Rights to a Machine

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**. Oracle Studio opens, displaying the Configuration perspective.
2. Right-click the machine in the **Configuration Explorer** and choose **Administration Authorization** from the popup menu.

   The Administration Authorization window opens.



The window has three sections:

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

Designers - Designers can view all the definitions for the machine in Oracle Studio and can modify any of the definitions under the **Bindings** and **Users** nodes for the selected machine. 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 machine in Oracle Studio for the selected machine. 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 machine.

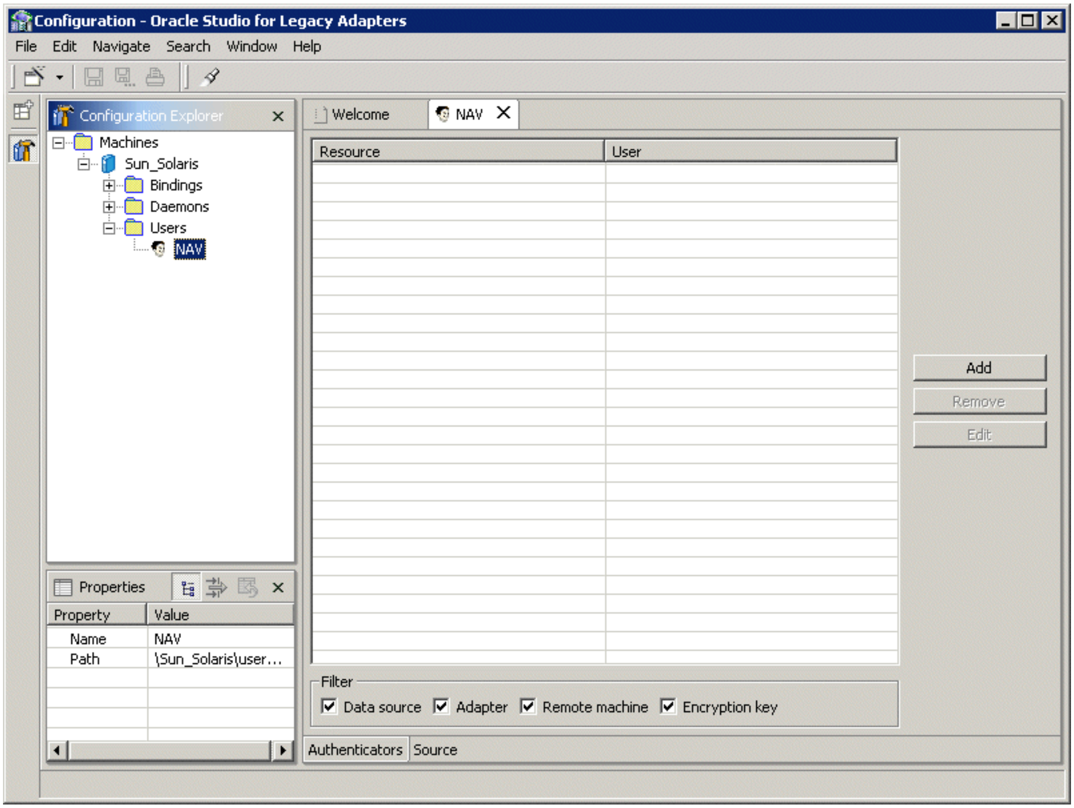
   Once a name has been added to a section, only the user or group who logs on with that user name has the relevant authorization.

Setting Up Runtime User Access to the Tuxedo Workstation Machine

At runtime, client access to Oracle Connect for Tuxedo is provided by the user profile. A user profile contains name and password pairs that are used to access a machine, data source or application at runtime, when anonymous access is not allowed.

1. In the **Configuration Explorer**, click the + next to the machine node of the machine for which you want to set the user name and password.
2. Click the + next to the **Users** node.
3. Right-click the **NAV** user profile and choose **Edit User**.

Caution: You must not delete the NAV user profile since it also contains Oracle Connect for Tuxedo system settings.



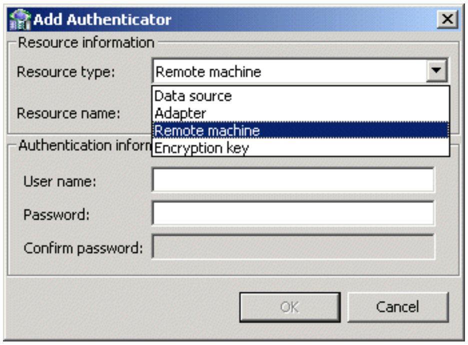
Property	Value
Name	NAV
Path	\Sun_Solaris\user...

Filter

☑ Data source ☑ Adapter ☑ Remote machine ☑ Encryption key

4. In the User editor, click **Add** to open the Add authenticator window.

5. Select the **Remote Machine** resource type.



Resource type: Remote machine

6. Enter the name of the Tuxedo workstation machine defined in the previous task.
7. Enter the name and password used to access the machine and confirm the password.
8. Click **OK**.

3 Modeling Interactions for Oracle Application Server Integration Adapter for Tuxedo

Modeling interactions for Oracle Application Server Integration Adapter for Tuxedo involves defining an Oracle Connect for Tuxedo backend adapter using Oracle Studio. All the definitions specified in Oracle Studio are written to the platform where the Tuxedo workstation runs.

This chapter contains the following sections:

- Setting Up Oracle Connect for Tuxedo
- Generating Outbound Interactions
- Generating Inbound Interactions

Setting Up Oracle Connect for Tuxedo

To work with the Oracle Connect for Tuxedo, you need to set up an adapter definition on the platform where the Tuxedo workstation runs to handle the interactions to and from a Tuxedo service. You use Oracle Studio to specify these definitions.

Task: Setting Up the Backend Adapter

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. In the Configuration Explorer, click the + next to the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Click the + next to the **Bindings** node. The binding configurations available on this machine are listed.
4. Click the + next to the **NAV** binding node. The NAV binding configuration includes branches for data sources and adapters that are located on the machine.

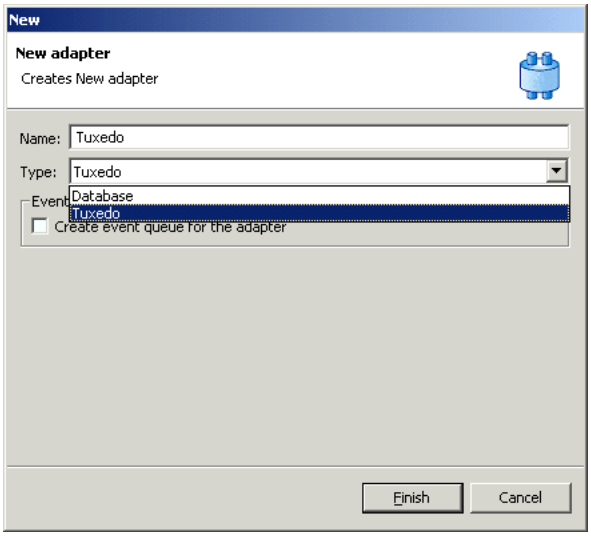
> **Caution:** You must not delete the NAV binding configuration since it also contains Oracle Connect for Tuxedo system settings.

5. Right-click **Adapters** and choose **New Adapter** from the popup menu, to open the New Adapter wizard.
6. Enter a name for the backend adapter.

> **Note:** The name specified for the backend adapter is also the name specified setting up the delivery channel in Oracle Application Server ProcessConnect. For details see, "Setting Up the Delivery Channel" on page 4-2.

> **Note:** The word *event* is a reserved word and cannot be used when naming an adapter.

7. Select `Tuxedo` as the backend adapter type from the `Type` drop-down list:



New

New adapter

Creates New adapter

Name: Tuxedo

Type: Tuxedo

Event

□ Create event queue for the adapter

8. Check the **Create event queue for the adapter** checkbox.

> **Note:** Checking this box enables you to generate inbound interactions, as described in Generating Inbound Interactions on page 3-27.

9. Click **Finish**. The backend adapter is added to the tree and the definition opened for editing.

Generating Outbound Interactions

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

Use the Metadata perspective in Oracle Studio to generate interaction metadata.

- Task 1: Defining an Import Procedure
- Task 2: Generating Outbound Interactions

Task 1: Defining an Import Procedure

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. In the Configuration Explorer, click the + next to the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Click the + next to the **Bindings** node. The binding configurations available on this machine are listed.
4. Click the + next to the **NAV** binding node.
5. Click the + next to the **Adapters** node.
6. Right-click the Tuxedo backend adapter defined in "Setting Up Oracle Connect for Tuxedo" on page 3-2.
7. Choose **Import** from the popup menu, to open the Metadata Import perspective, with the Tuxedo backend adapter displayed under the Adapters list.
8. Right-click the Tuxedo backend adapter and choose **New Import** from the popup menu, to open the New Metadata Import wizard.
9. Enter a name for the import. The name can contain letters and numbers and the underscore character only.
10. Select the metadata import type. You can import the metadata either from a BEA Jolt file or from Tuxedo record definition files (FML or VIEW files).

    > **Note:** Jolt files are files where definitions of Tuxedo services are bulked into one file along with the metadata using the BEA Jolt Bulk Loader. FML and VIEW files are metadata files used by Tuxedo in conjunction with a configuration file that includes the Tuxedo services.

    For Jolt files, select `Tuxedo Import Manager Using Jolt Bulk Loader File`. For FML or VIEW files, select `Import Manager Using VIEW Files`.
11. Click **Finish**. The metadata import wizard opens in Oracle Studio.

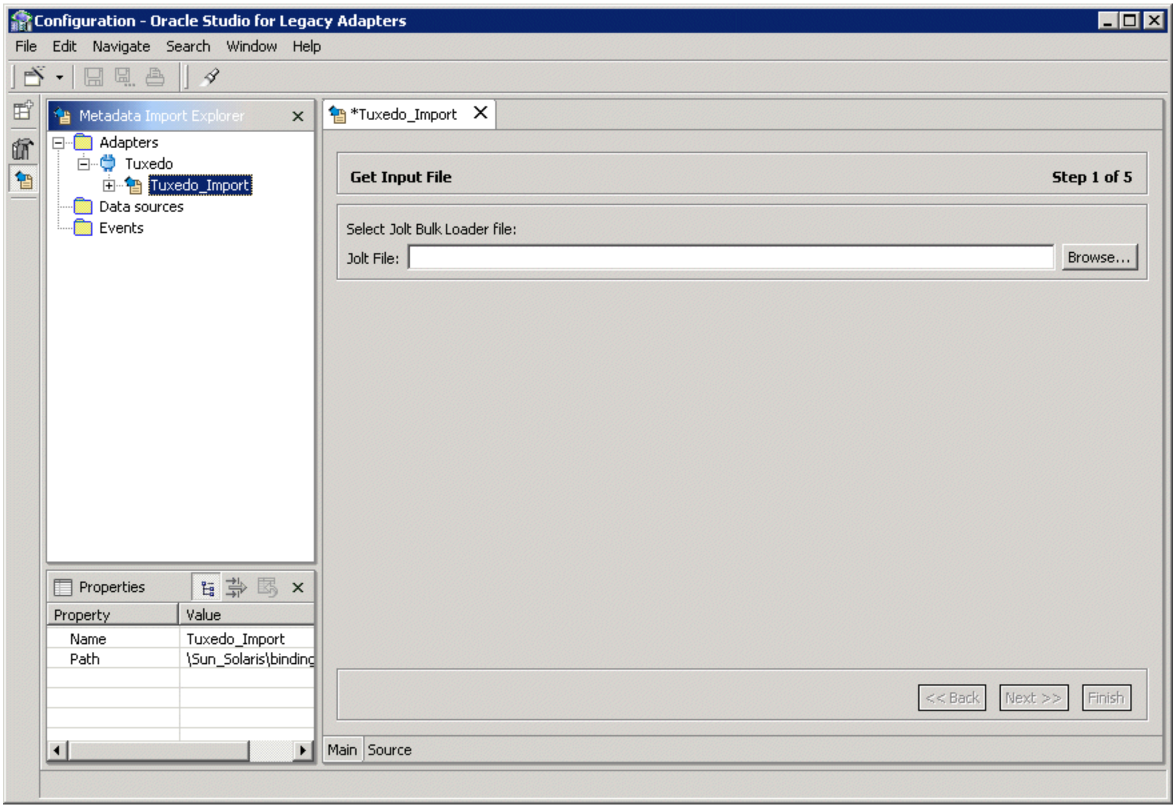
Task 2: Generating Outbound Interactions

After defining an import as described in the preceding task, the metadata import wizard opens in Oracle Studio. The input files are used to create the metadata. The import wizard generates record structures, which are used for the input and resulting output for outbound interactions. The following options are available:

- Generating Outbound Interactions Using a Jolt File
- Generating Outbound Interactions Using View Files

Generating Outbound Interactions Using a Jolt File

A Jolt file is used to generate record structures, which are used for the input and resulting output for outbound interactions. The import wizard opens with a panel where you specify the Jolt file to use.



Get Input File Step 1 of 5

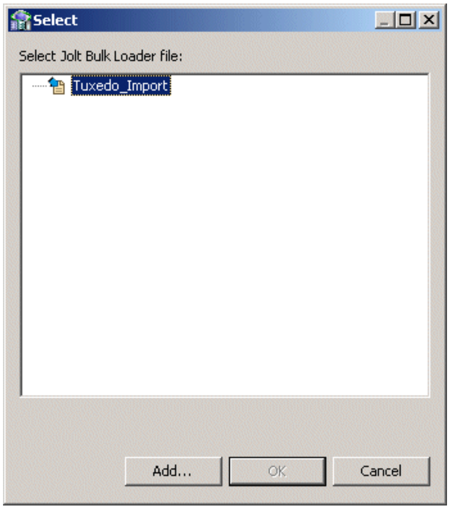
Select Jolt Bulk Loader file:

Jolt File:

Property	Value
Name	Tuxedo_Import
Path	\Sun_Solaris\binding

1. Click **Add** to add the Jolt file.

   A Select window opens. This window lists any files selected.



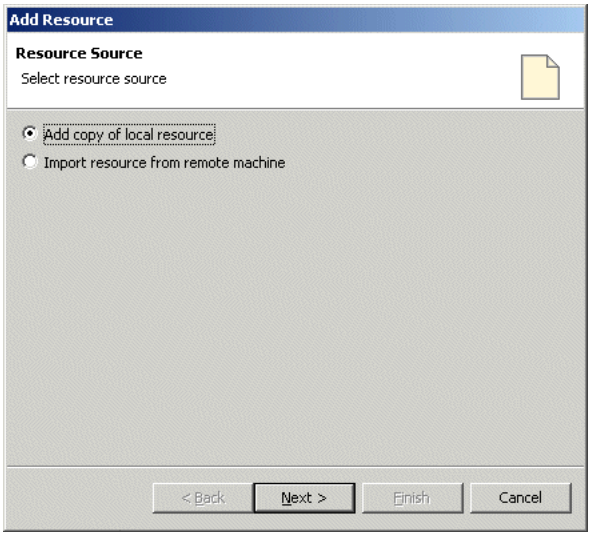
Select

Select Jolt Bulk Loader file:

Tuxedo_Import

2. Click **Add**.

   The Add Resource window opens, which provides the option to select files from the local machine or FTP the files from another machine.



Add Resource

Resource Source

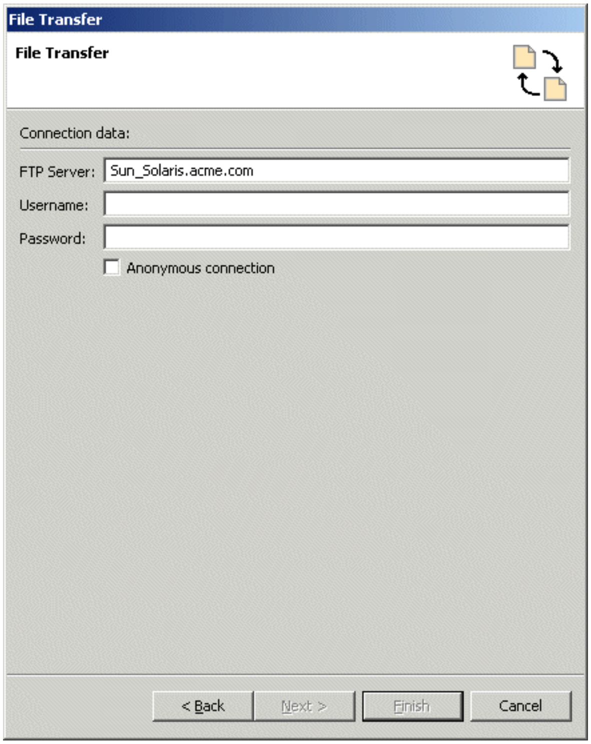
Select resource source

● Add copy of local resource

○ Import resource from remote machine

3. If the files are on another machine, select the **Import resource from remote machine** option to start a file transfer process from the machine.

4. Click **Next** to open the File Transfer wizard.



File Transfer

File Transfer

Connection data:

FTP Server: Sun_Solaris.acme.com

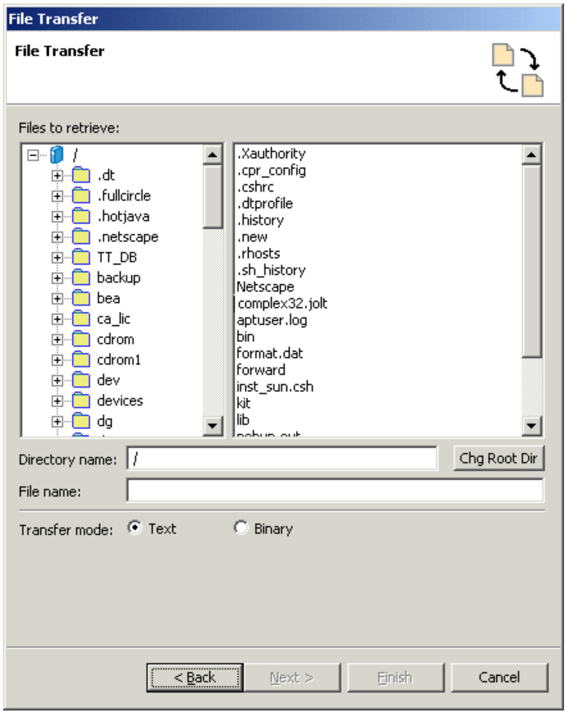
Username:

Password:

□ Anonymous connection

5. Enter the server name where the Tuxedo workstation resides and, if not using anonymous access, enter a username and password to access the machine.

6. Click **Next**. The machine is accessed and the directory structure displayed, enabling you to browse to the files to be transferred.



Files to retrieve:

/
- .dt
- .fullcircle
- .hotjava
- .netscape
- TT_DB
- backup
- bea
- ca_lic
- cdrom
- cdrom1
- dev
- devices
- dg

.Xauthority
.cpr_config
.cshrc
.dtprofile
.history
.new
.rhosts
.sh_history
Netscape
complex32.jolt
aptuser.log
bin
format.dat
forward
inst_sun.csh
kit
lib

Directory name: /

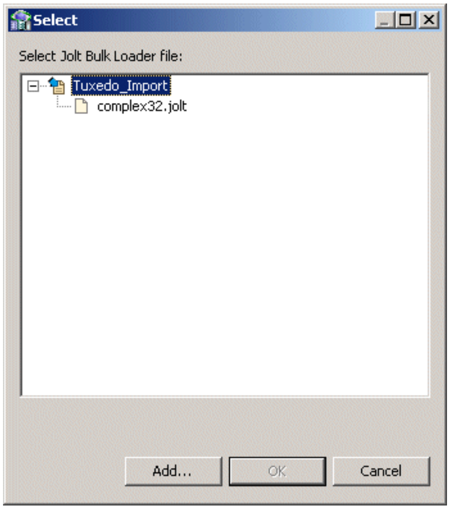
File name:

Transfer mode: ● Text ○ Binary

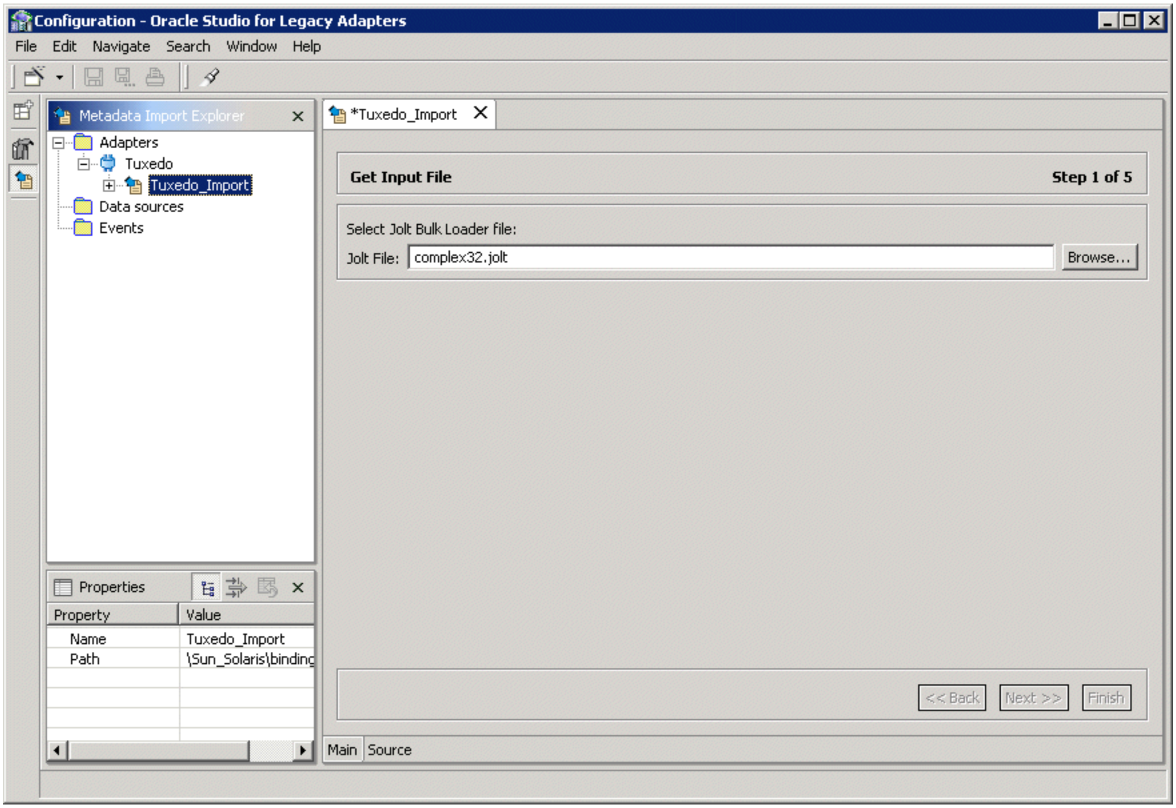
To change the root directory, enter the name of the root directory in the Directory name field and click **Chg Root Dir**.

7. Click **Next**. A summary screen is displayed with details of what is going to be transferred.

8. Click **Finish** to start the transfer. The **Select** window is redisplayed with the list of transferred files.



9. Select the file you want to use in the import and click **OK**.



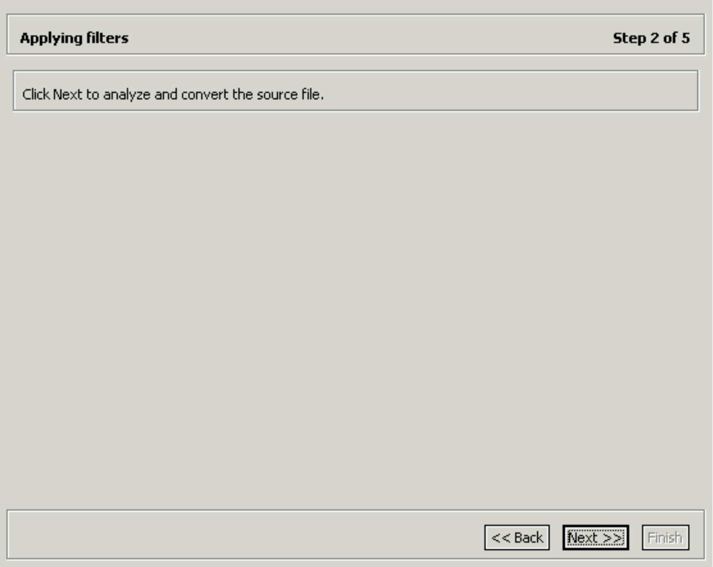
Get Input File — Step 1 of 5

Select Jolt Bulk Loader file:

Jolt File: complex32.jolt

Property	Value
Name	Tuxedo_Import
Path	\Sun_Solaris\binding

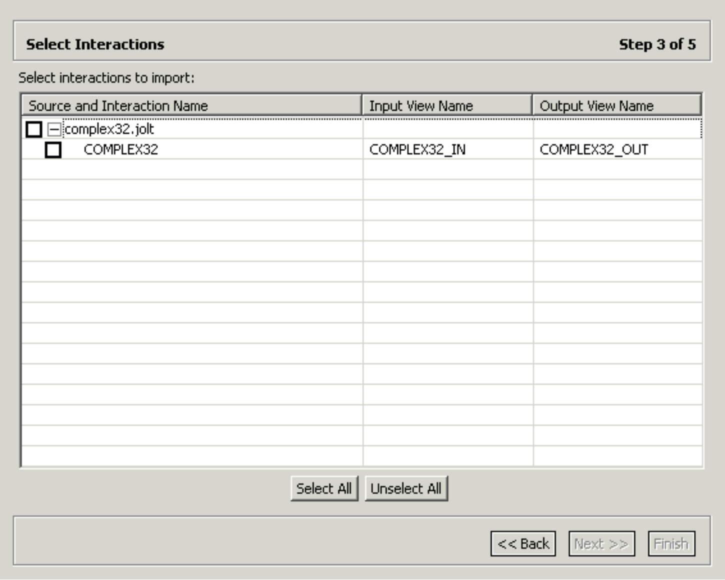
10. Click **Next**. The **Apply Filters** window opens.



Applying filters **Step 2 of 5**

Click Next to analyze and convert the source file.

11. Click **Next**. The **Select Interactions** window opens.



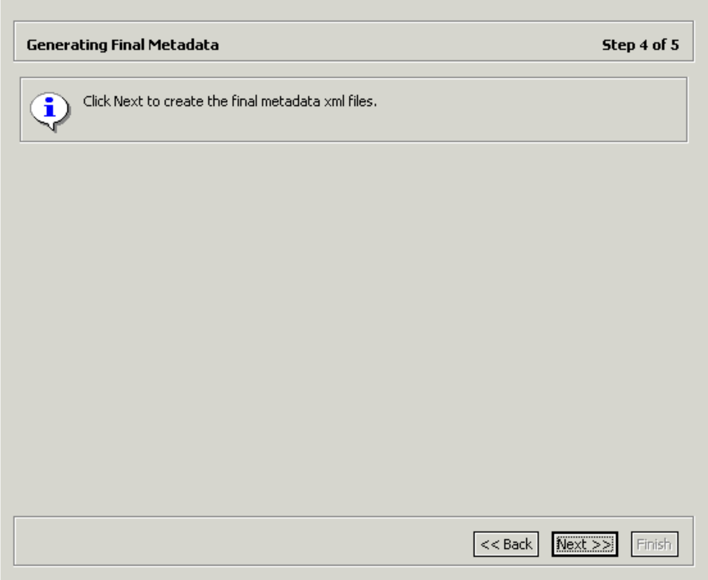
Select Interactions **Step 3 of 5**

Select interactions to import:

Source and Interaction Name	Input View Name	Output View Name
□ ⊟ complex32.jolt		
□ COMPLEX32	COMPLEX32_IN	COMPLEX32_OUT

The window lists the Services from the Jolt file as interactions, along with the input and output structures to use with each interaction. Select the interactions you want to implement from the list.

12. Click **Next**.

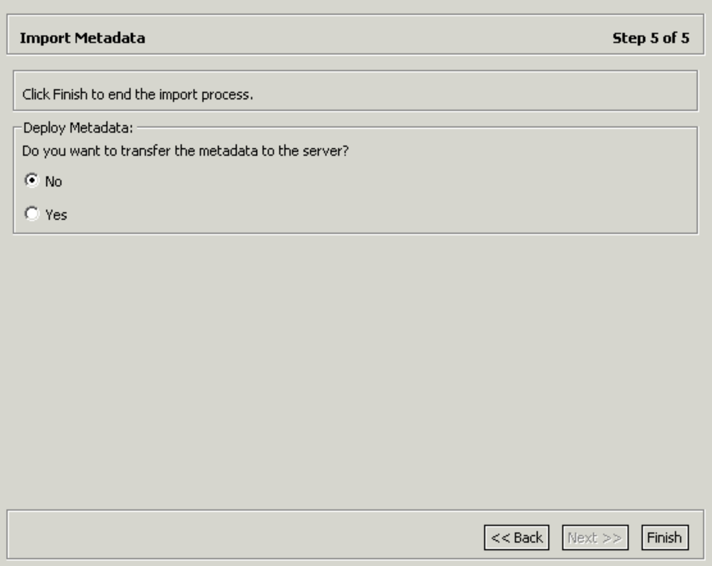


Generating Final Metadata

Step 4 of 5

Click Next to create the final metadata xml files.

13. Click **Next** to generate the metadata definitions for the adapter.



Import Metadata — Step 5 of 5

Click Finish to end the import process.

Deploy Metadata:

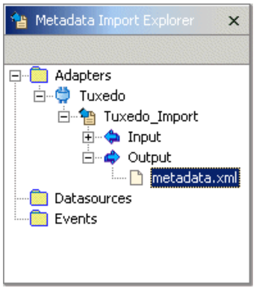
Do you want to transfer the metadata to the server?

● No

○ Yes

14. Specify that you want to transfer the metadata from the Windows machine to the Tuxedo workstation machine and click **Finish**.

The metadata is imported based on the options specified and it is stored on the machine where the Tuxedo workstation runs. An XML representation of the metadata is generated. This XML file can be viewed by expanding the Output node.



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

> **See Also:** "Interaction Metadata for the Adapter" on page A-2 for details about the adapter metadata

Generating Outbound Interactions Using View Files

A configuration file is used to get the information necessary for starting application servers and initializing the bulletin boards in an orderly sequence in Tuxedo. A configuration file can have a number of sections, including a SERVICES section which provides information on services used by the application, from which interactions are generated during the import procedure.

The view files contain metadata used by Tuxedo and are used to provide the data structures for the interactions. The import wizard opens with a panel where you specify the files to use.

Both the configuration files and the view files are binary files, compiled from text files. The import wizard uses the text files to generate the metadata. If the text files are no longer available, you can recreate them from the binary files as follows:

For configuration files, run the following Tuxedo command:

```
tmunloadcf config_file
```

where *config_file* is the name of the configuration file.

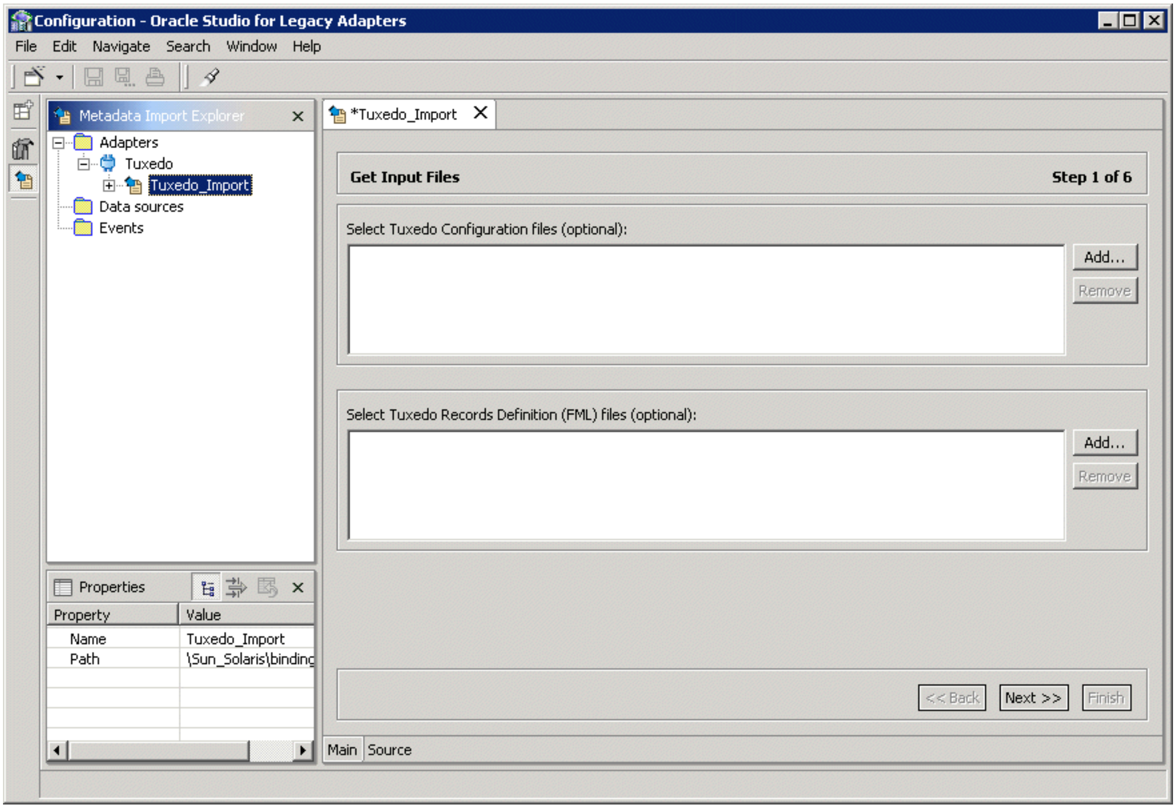
For view files, run either of the following Tuxedo command:

```
viewdis view_file
```

or, for 32bit view files:

```
viewdis32 view_file
```

where *view_file* is the name of the configuration file.



*Tuxedo_Import

Get Input Files — Step 1 of 6

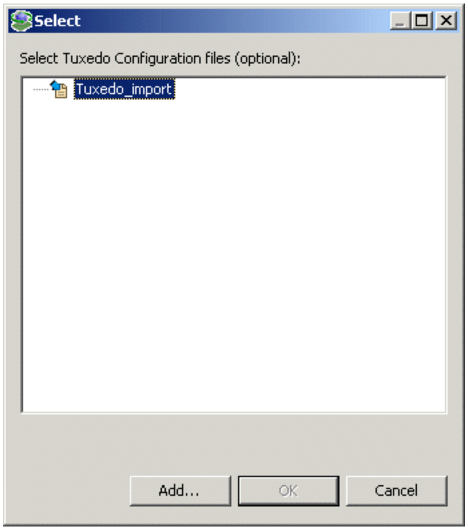
Select Tuxedo Configuration files (optional):

Select Tuxedo Records Definition (FML) files (optional):

Property	Value
Name	Tuxedo_Import
Path	\Sun_Solaris\binding

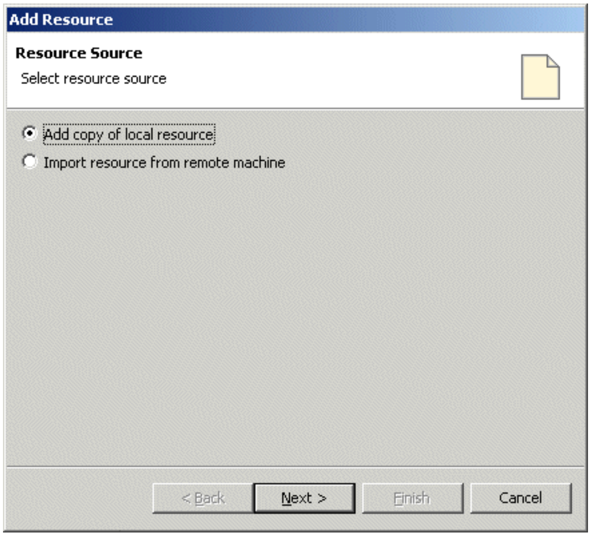
1. Click **Add** to add the Tuxedo configuration files.

   A Select window opens. This window lists any files selected.



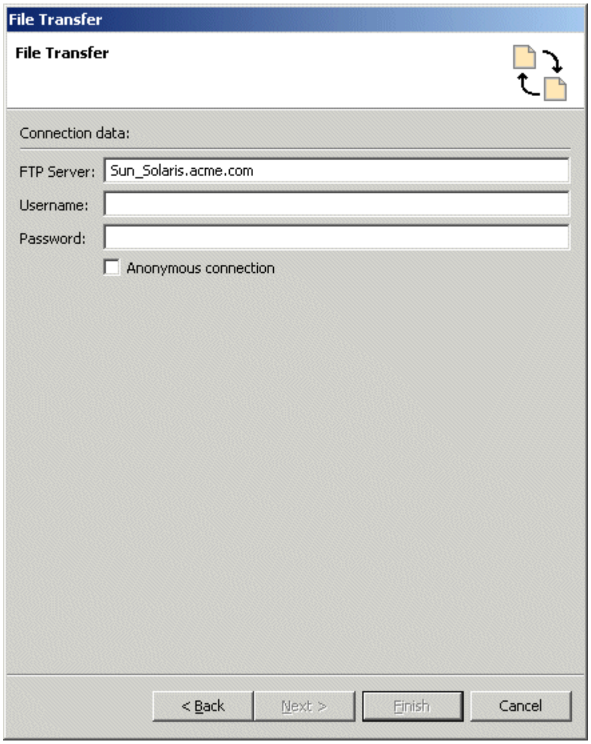
2. Click **Add**.

   The Add Resource window opens, which provides the option to select files from the local machine or FTP the files from another machine.



3. If the files are on another machine, select the **Import resource from remote machine** option to start a file transfer process from the machine.

4. Click **Next** to open the File Transfer wizard.



File Transfer

File Transfer

Connection data:

FTP Server: Sun_Solaris.acme.com

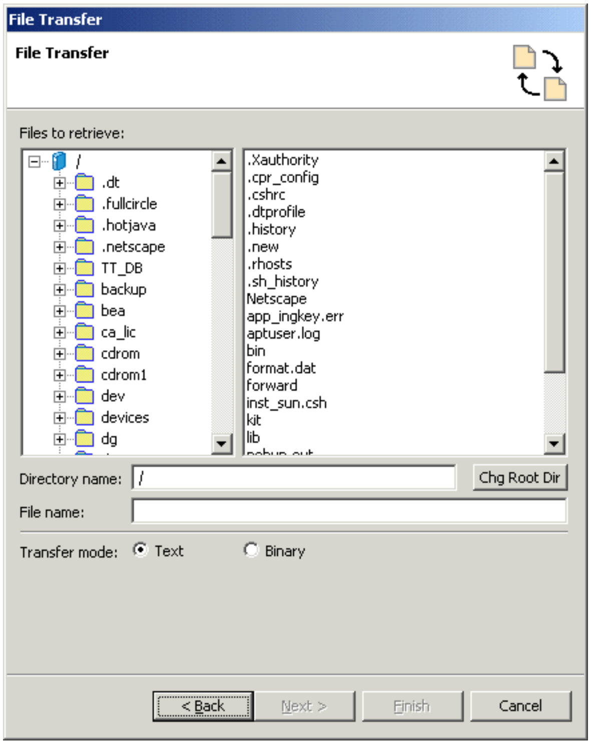
Username:

Password:

□ Anonymous connection

5. Enter the server name where the Tuxedo workstation resides and, if not using anonymous access, enter a username and password to access the machine.

6. Click **Next**. The machine is accessed and the directory structure displayed, enabling you to browse to the files to be transferred.



Directory name: /

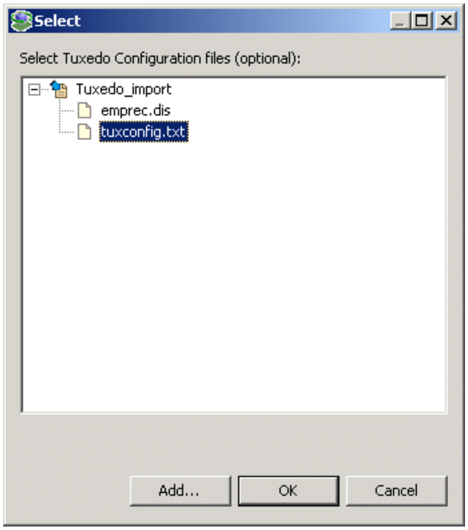
File name:

Transfer mode: ● Text ○ Binary

To change the root directory, enter the name of the root directory in the Directory name field and click **Chg Root Dir**.

7. Click **Next**. A summary screen is displayed with details of what is going to be transferred.

8. Click **Finish** to start the transfer. The **Select** window is redisplayed with the list of transferred files.

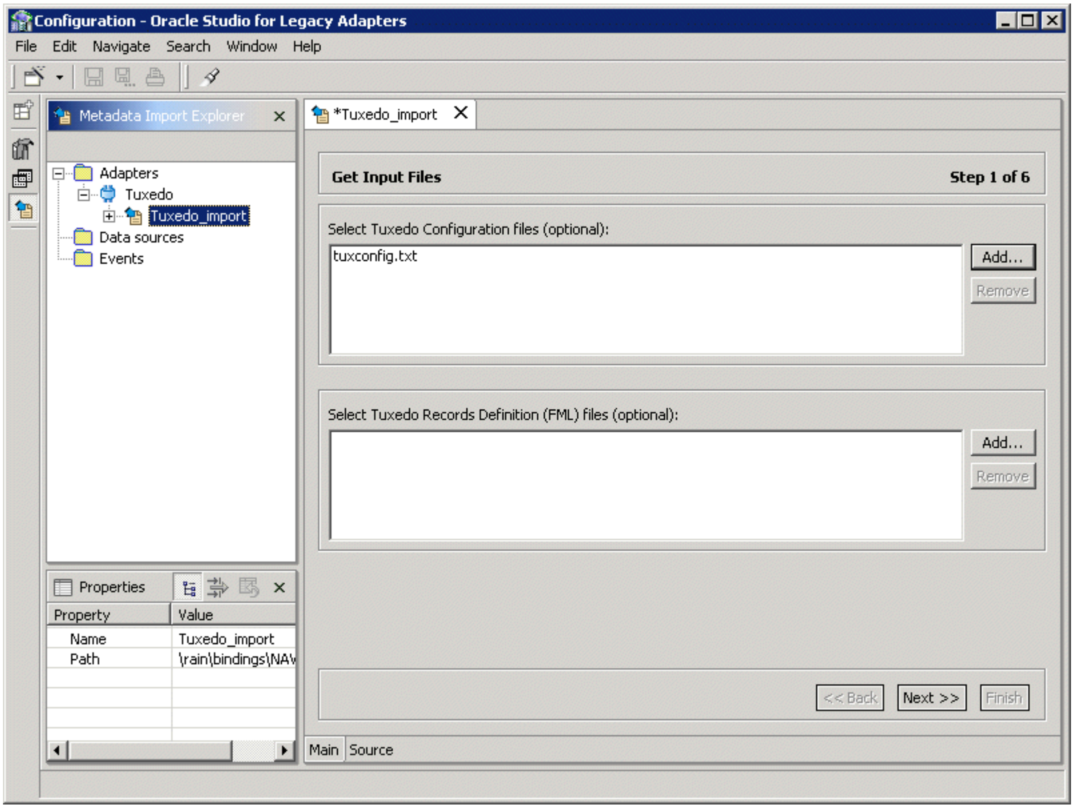


Select

Select Tuxedo Configuration files (optional):

Tuxedo_import
emprec.dis
tuxconfig.txt

9. Select the files you want to use in the import and click **OK**.



Get Input Files Step 1 of 6

Select Tuxedo Configuration files (optional):

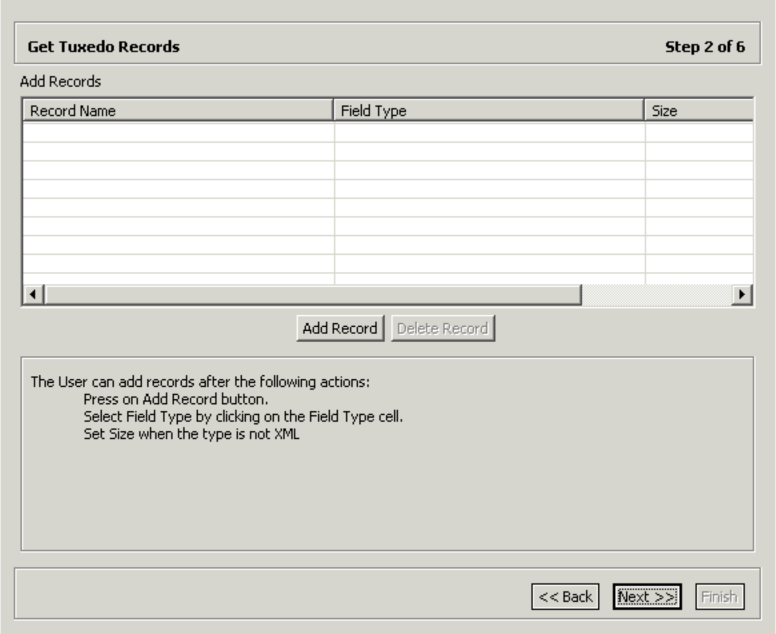
tuxconfig.txt

Select Tuxedo Records Definition (FML) files (optional):

Property	Value
Name	Tuxedo_import
Path	\rain\bindings\NAV

10. Repeat the previous steps (from step 1) for the Tuxedo record definition (FML) files.

11. In the metadata import wizard, click **Next** after you have completed adding the list of files to use.



Get Tuxedo Records **Step 2 of 6**

Add Records

Record Name	Field Type	Size

The User can add records after the following actions:
Press on Add Record button.
Select Field Type by clicking on the Field Type cell.
Set Size when the type is not XML

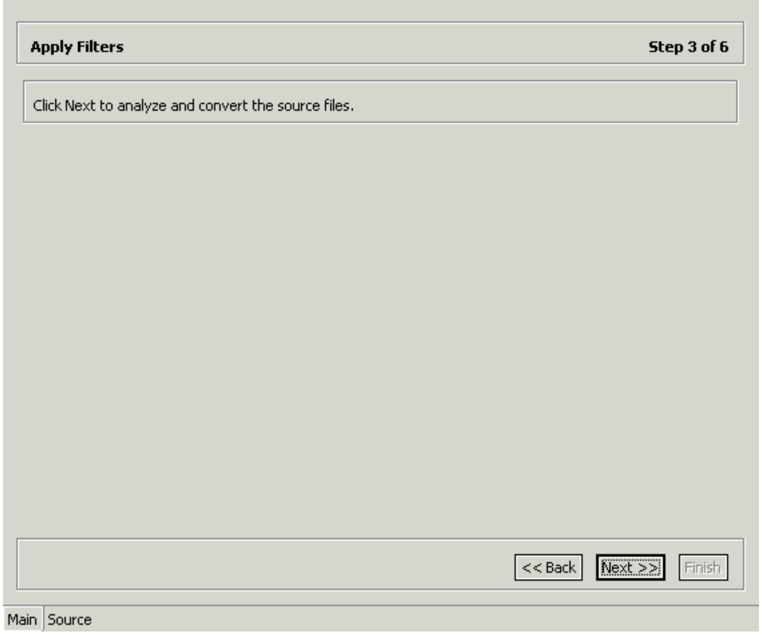
12. Add additional simple record definitions, if necessary.

    In addition to view files, you can include information stored in the following Tuxedo buffer types:

    - XML data
    - String data
    - Carrays

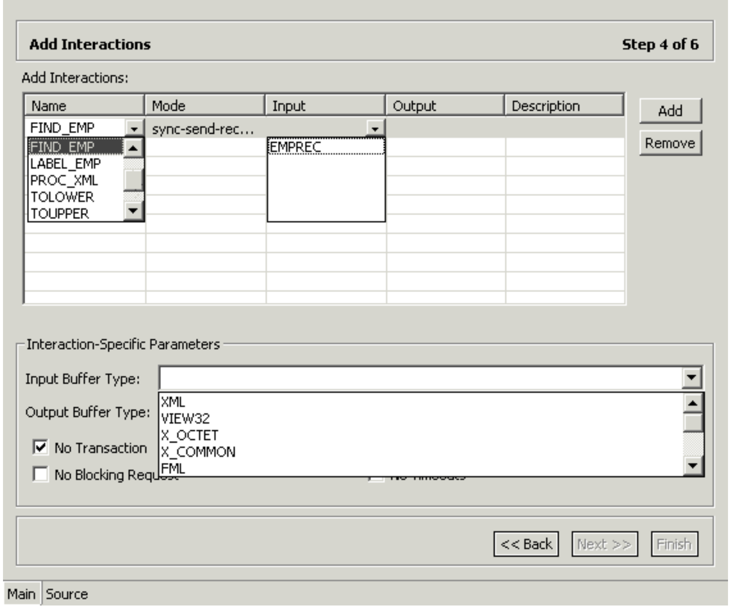
    Specify the name of the record and select the buffer type from the list in the `Field Type` field.

13. Click **Next**. The **Apply Filters** window opens.



Click Next to analyze and convert the source files.

14. Click **Next**. The **Add Interactions** window opens.



Add Interactions **Step 4 of 6**

Add Interactions:

Name	Mode	Input	Output	Description
FIND_EMP	sync-send-rec...	EMPREC		

Interaction-Specific Parameters

Input Buffer Type:

Output Buffer Type:

☑ No Transaction

□ No Blocking Request

15. Click **Add** to add an interaction for the Tuxedo adapter.

    Interaction name - The name of the interaction. Select the interaction from the list. The list is taken from the list of services in the configuration file.

    Mode - The interaction 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.
    - **sync-send-receive** - The interaction sends a request and expects to receive a response.

    Input - Identifies an input record. The input record is the data structure for the outbound interaction. The records generated from the FML programs specified at the beginning of the procedure are listed in a drop-down list. 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, the record specified here is ignored.

Output - Identifies an output record. The output record is the data structure for the results of the outbound interaction. The records generated from the FML programs specified at the beginning of the procedure are listed in a drop-down list. 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.

Interaction-Specific Parameters - Tuxedo specific parameters.

Input Buffer Type - The type of data used for the input.

Output Buffer Type - The type of the buffer to use for the results of an outbound interaction.

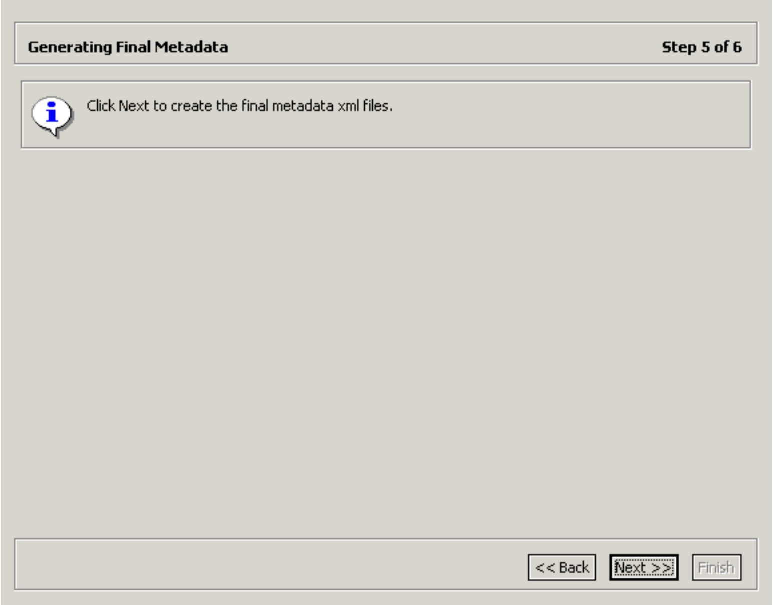
No Transaction - Enables a service to be executed, regardless of transaction context. This parameter should always be checked.

No Reply Expected - For future use.

No Blocking Request - Avoids a FROM request submission if a blocking condition exists.

No Timeouts - Ignores blocking timeouts.

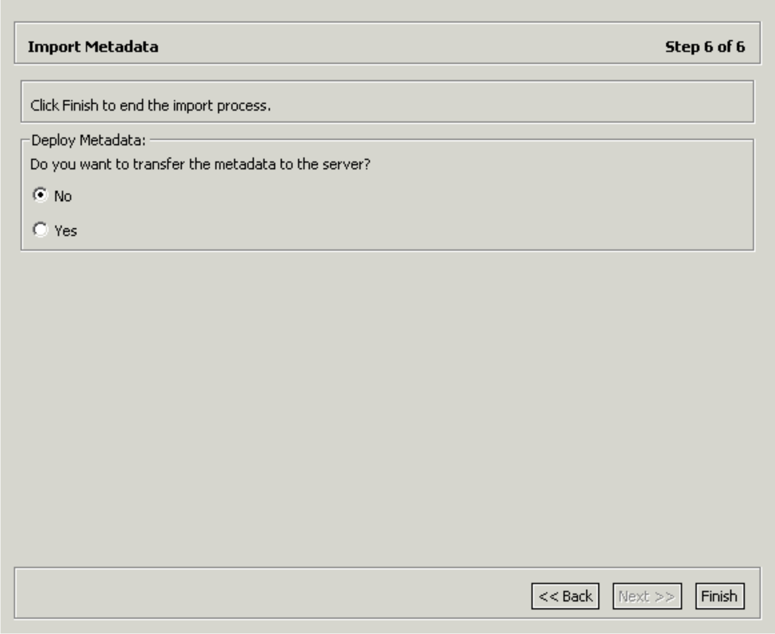
16. Add as many interactions as necessary and then click **Next**.



Generating Final Metadata **Step 5 of 6**

Click Next to create the final metadata xml files.

17. Click **Next** to generate the metadata definitions for the adapter.



Import Metadata **Step 6 of 6**

Click Finish to end the import process.

Deploy Metadata:

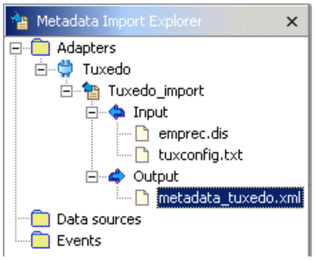
Do you want to transfer the metadata to the server?

● No

○ Yes

18. Specify that you want to transfer the metadata from the Windows machine to the Tuxedo workstation machine and click **Finish**.

The metadata is imported based on the options specified and it is stored on the machine where the Tuxedo workstation runs. An XML representation of the metadata is generated. This XML file can be viewed by expanding the Output node.



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

See Also: "Interaction Metadata for the Adapter" on page A-2 for details about the adapter metadata

Generating Inbound Interactions

Inbound interactions are defined as events in Oracle Studio. When you defined Oracle Connect for Tuxedo's backend adapter with the **Create event queue for the adapter** box checked, as described in "Task: Setting Up the Backend Adapter" on page 3-2, an event adapter was defined automatically. The event adapter is defined with the same name as the backend adapter with the word *event* appended to it. The backend adapter and the event adapter are linked by Oracle Studio. You can jump from one adapter definition to the other by right-clicking the adapter or event in the **Configuration Explorer** list and choosing the Linked Event or Linked Adapter option respectively.

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

Use the Import Metadata perspective in Oracle Studio to generate interaction metadata for the adapter.

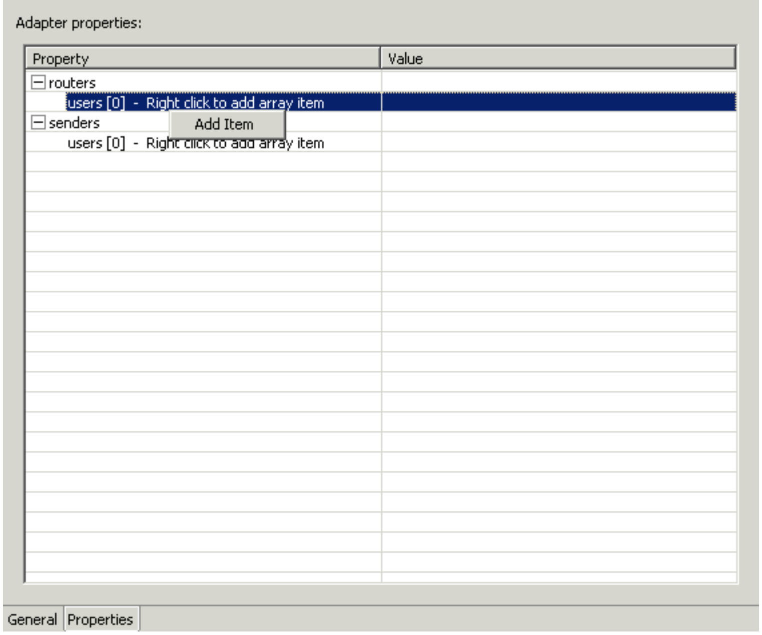
- Task 1: Defining an Import Procedure
- Task 2: Generating Inbound Interactions

Task 1: Defining an Import Procedure

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. In the Configuration Explorer, click the + next to the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Click the + next to the **Bindings** node. The binding configurations available on this machine are listed.
4. Click the + next to the **NAV** binding node.
5. Click the + next to the **Adapters** node.
6. Right-click the Tuxedo adapter defined in "Setting Up Oracle Connect for Tuxedo" on page 3-2.
7. Choose **Linked Event** from the popup menu, to jump to the event adapter.
8. Right-click the event adapter and choose **Edit Event** from the popup menu.

9. Select the **Properties** tab to add the names of Oracle Application Server ProcessConnect users who can retrieve inbound interactions and Tuxedo users who can send inbound interactions.

10. To add Oracle Application Server ProcessConnect users, click the + next to the **routers** node and right click the users property to add the user.


Adapter properties:

Property	Value
⊟ routers	
users [0] - Right click to add array item	
⊟ senders	
users [0] - Right click to add array item	

11. Enter the name of the Oracle Application Server ProcessConnect user in the Value column for the item added.

12. To add Tuxedo users, click the + next to the **senders** node and right click the users property to add the user.

13. Enter the name of the Tuxedo user in the Value column for the item added.

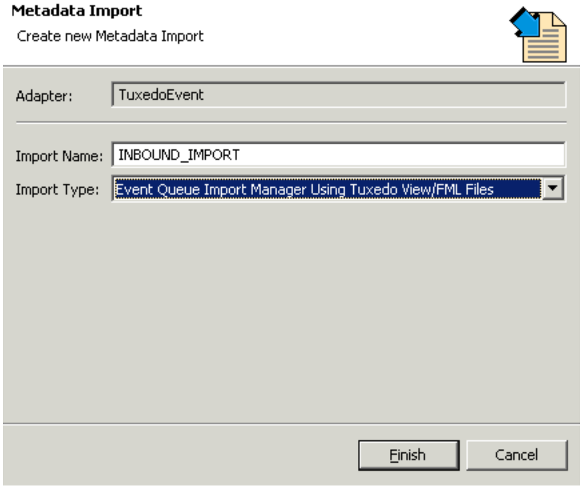
14. Click the **Save Editor Contents** button to save the changes.

15. Right-click the event adapter in the **Configuration Explorer** and choose **Edit metadata** from the popup menu, to open the Metadata perspective, with the event adapter displayed under the Events list.

16. Right-click the event adapter and choose **New Import** from the popup menu, to open the New Metadata Import wizard.

17. Enter a name for the import. The name can contain letters and numbers and the underscore character only.

18. Select **Event Queue Import Manager Using Tuxedo View/FML Files** for the `Import Type`.



Metadata Import

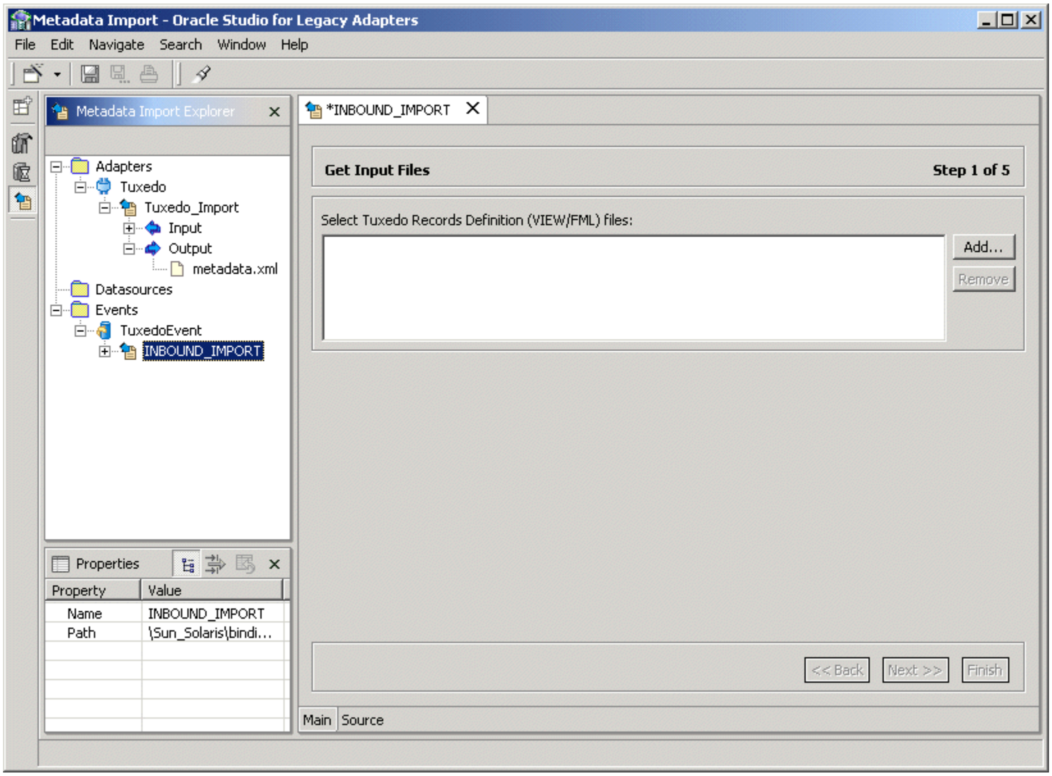
Create new Metadata Import

Adapter: TuxedoEvent

Import Name: INBOUND_IMPORT

Import Type: Event Queue Import Manager Using Tuxedo View/FML Files

19. Click **Finish**. The metadata import wizard opens in Oracle Studio.



Property	Value
Name	INBOUND_IMPORT
Path	\Sun_Solaris\bindi...

Task 2: Generating Inbound Interactions

After defining an import as described in the preceding task, the metadata import wizard opens in Oracle Studio. Tuxedo View and FML files are used to create the metadata. The import wizard generates record structures, which are used for the record structures for inbound interactions.

View and FML files are binary files, compiled from text files. The import wizard uses the text files to generate the metadata. If the text files are no longer available, you can recreate them from the binary files as follows:

Run either of the following Tuxedo command:

```
viewdis view_file
```

or, for 32bit view files:

```
viewdis32 view_file
```

where *view_file* is the name of the configuration file.

> **Note:** The generation of inbound interactions involves similar steps to the steps described in "Task 2: Generating Outbound Interactions" on page 3-5.

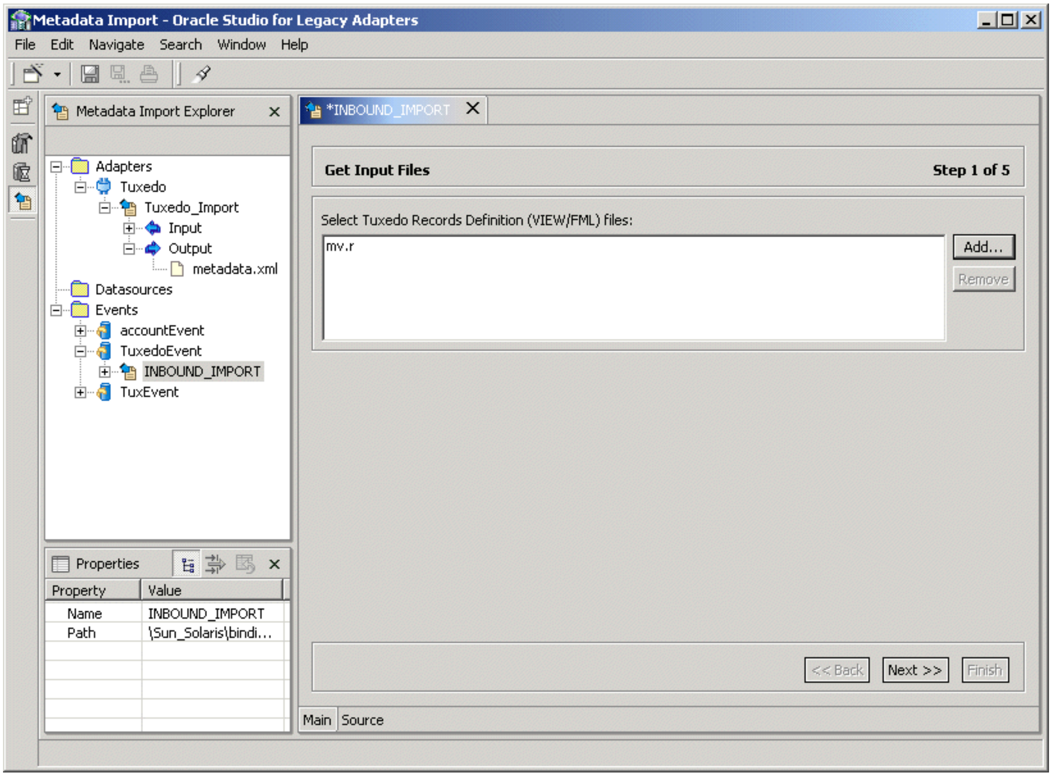
1. Click **Add** to add View or FML files.

   A Select window opens. This window lists any files selected.

2. Click **Add**.

   The Add Resource window opens, which provides the option to select files from the local machine or FTP the files from another machine.

3. If the files are on another machine, select the **Import resource from remote machine** option to start a file transfer process from the machine.
4. Click **Next** to open the File Transfer wizard.
5. Enter the server name where the files reside and, if not using anonymous access, enter a username and password to access the machine.
6. Click **Next**. The machine is accessed and the directory structure displayed, enabling you to browse to the files to be transferred.
7. Navigate to the location where the files reside and select the files to be transferred and the transfer method (**Text** or **Binary**)
8. Click **Next**. A summary screen is displayed with details of what is going to be transferred.
9. Click **Finish** to start the transfer. The **Select** window is redisplayed with the list of transferred files.
10. Select the files you want to use in the import and click **OK**.



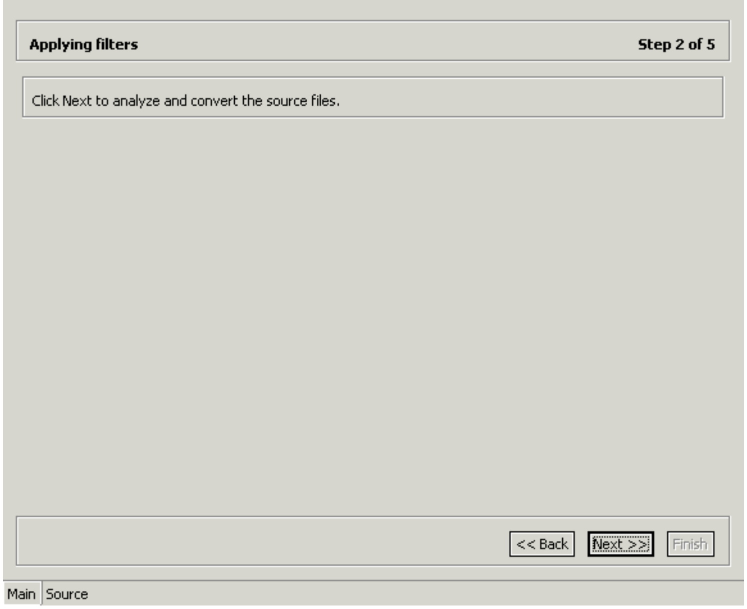
Get Input Files Step 1 of 5

Select Tuxedo Records Definition (VIEW/FML) files:

mv.r

Property	Value
Name	INBOUND_IMPORT
Path	\Sun_Solaris\bindi...

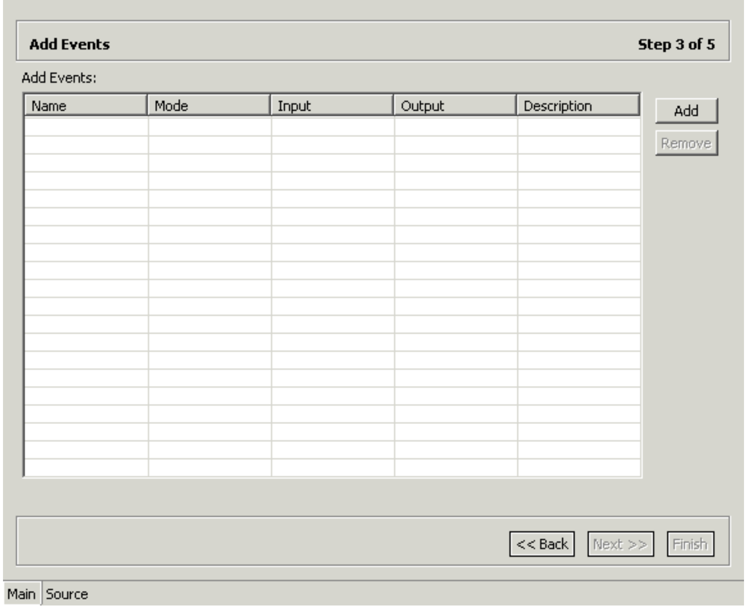
11. In the metadata import wizard, click **Next** after you have completed adding the files.



Applying filters Step 2 of 5

Click Next to analyze and convert the source files.

12. Click **Next** to apply the filters. The Add Events window opens.



Add Events Step 3 of 5

Add Events:

Name	Mode	Input	Output	Description

13. Click **Add** to add an interaction for the Tuxedo adapter.

    Name - The name of the interaction.

    Mode - The interaction mode:

    > **async-send** - The inbound interaction sends a request and does not expect to receive a response.

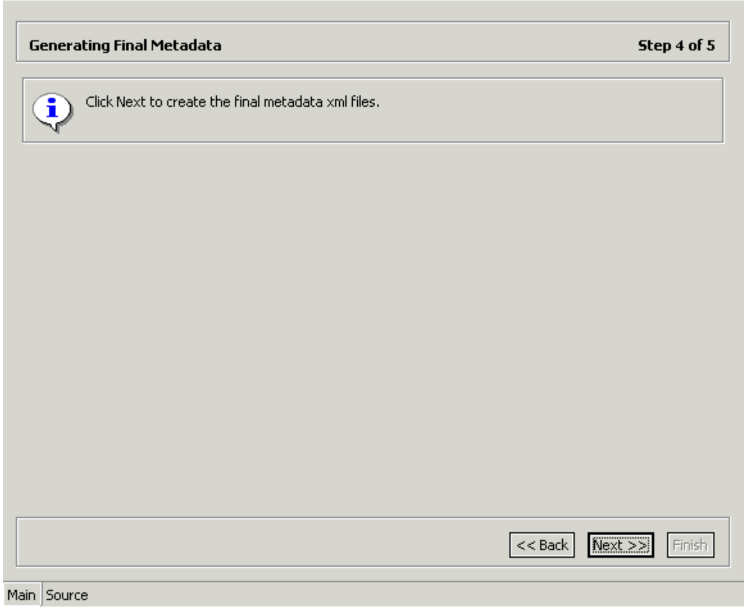
    Input - Identifies an input record. The input record is the data structure for the inbound interaction. The records generated from the program specified at the beginning of the procedure are listed in a drop-down list. Select the relevant record for the interaction.

    > **Note:** You must specify an input record for each interaction before you can click **Next**.

    Output - For internal use. You cannot enter a value in this field.

    Description - Free text describing the interaction.

14. Add as many interactions as necessary and then click **Next**.

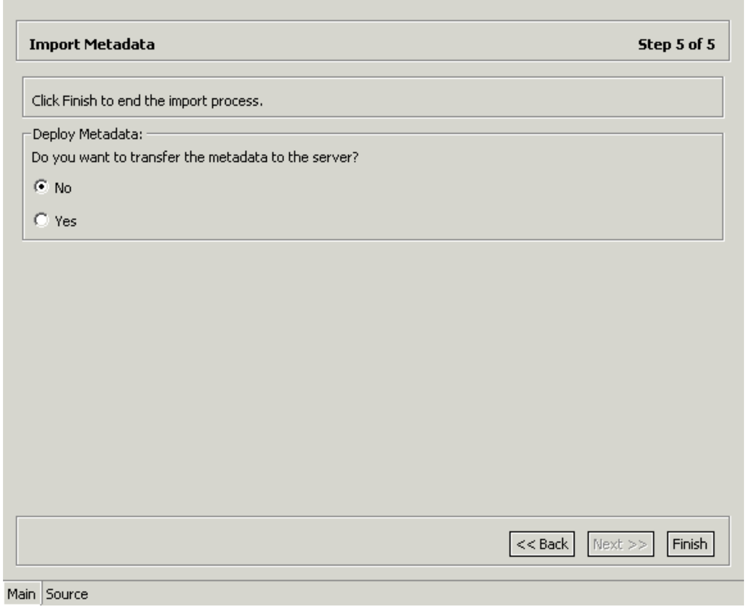


Generating Final Metadata

Step 4 of 5

Click Next to create the final metadata xml files.

15. Click **Next** to generate the metadata definitions for the adapter.



Import Metadata **Step 5 of 5**

Click Finish to end the import process.

Deploy Metadata:

Do you want to transfer the metadata to the server?

● No

○ Yes

16. Specify that you want to transfer the metadata from the Windows machine to the Tuxedo machine and click **Finish**.

The metadata is imported based on the options specified and it is stored on the Tuxedo machine. An XML representation of the metadata is generated. This XML file can be viewed by expanding the Output node.

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

> **See Also:** "Interaction Metadata for the Adapter" on page A-2 for details about the adapter metadata

4

Using Interactions in a Business Process

After the Tuxedo data source and Oracle Connect for Tuxedo have been modeled as described in Chapter 3, "Modeling Interactions for Oracle Application Server Integration Adapter for Tuxedo", the interactions can be used in a business process in Oracle Application Server ProcessConnect. You define a delivery channel for the Tuxedo adapter part of the Oracle Application Server Integration Adapter for Tuxedo. This delivery channel is used during the modeling in Oracle Application Server ProcessConnect with the interactions defined for the adapter in Oracle Studio in a business process.

This chapter contains the following sections:

- Adding the Tuxedo Adapter in Oracle Application Server ProcessConnect
- Setting Up the Delivery Channel
- Modifying an Adapter In Oracle Studio
- Comparing a ProcessConnect XSD to an Oracle Connect Record Schema

Adding the Tuxedo Adapter in Oracle Application Server ProcessConnect

When adding the adapter type during the application definition, specify `Tuxedo Adapter`.

> **See Also:** *Oracle Application Server ProcessConnect User's Guide* for details of setting up applications in Oracle Application Server ProcessConnect

Setting Up the Delivery Channel

Part of the application definition includes adding a delivery channel for the Oracle Application Server Integration Adapter for Tuxedo.

The delivery channel identifies where the backend adapter is located and the necessary security and encryption requirements to access the adapter.

Task: Setting Up the Delivery Channel

1. In the Create Delivery Channel page for the adapter, add the following information to create a delivery channel.

Field	Description
Name	A name for this delivery channel You must provide a value for this field
The Adapter Name	The name of the Oracle Connect for Tuxedo's backend adapter to use, as defined in Oracle Studio. You can check the name specified for the adapter in Oracle Studio You must provide a value for this field
The Workspace to use	The name of a server workspace to use. Leave blank to use the default workspace. If you used another workspace when modeling the adapter in Oracle Studio, you can check the workspace name in the Configuration perspective in Oracle Studio
The Remote machine Name	The server machine TCP/IP host name. The host name is the name of the machine where the Oracle Connect for Tuxedo's backend adapter was defined using Oracle Studio You must provide a value for this field

Field	Description
The Port to connect	The TCP/IP port where the daemon is running on the server. The default port is 2552. If you specified another port when modeling the adapter in Oracle Studio, you can check the port specified by right-clicking on the machine in Oracle Studio and checking the port number displayed
Protocol Name	The firewall protocol used: either none or fixedNat (the Nat protocol using a fixed address for the daemon). The default is none
Timeout via Connect	Connection timeout in seconds. The default is 0, meaning that there is no connection timeout
Encryption protocol name	Name of encryption protocol to use. The default is null. The RC4 protocol is supported
Encryption key name	Name of the symmetric encryption key to use
Encryption key value	Value of the symmetric encryption key to use
The User Name	The back-end user name for authentication. Leave blank if anonymous access is allowed.
The User Password	The back-end user password for authentication. Leave blank if anonymous access is allowed
Directory to store rejected inbound interactions	The directory where files with information about rejected inbound interactions are stored. Each rejected inbound interaction is stored in a file whose name consists of a timestamp and the interaction name.

2. Click **Apply**.

   The delivery channel is defined for the application and the Delivery Channels Details page for the added delivery channel is displayed.

You can now select any of the interactions defined for the adapter.

> **Note:** You cannot add, modify or delete interactions for the adapter from Oracle Application Server ProcessConnect. Any changes to the interactions must be done in Oracle Studio.

Modifying an Adapter In Oracle Studio

The procedure for making changes in Oracle Studio to an interaction or to the interaction input and output records and having the change reflected in Oracle Application Server ProcessConnect depends on whether the configuration was deployed or not.

Before the configuration has been deployed, delete the interaction from the application (and all relevant items that reference the interaction) and after making the change in Oracle Studio, add the changed interaction to the application.

After the configuration has been deployed, you cannot change the interaction until you have performed the following procedure:

1. Stop both the adapter framework and integration manager processes.
2. Delete the interaction from the application (and all relevant items that reference the interaction).
3. Make the changes to the interaction in Oracle Studio.
4. In the same application, add the changed interaction.
5. Redeploy the configuration.
6. Start both the adapter framework and integration manager processes.

Comparing a ProcessConnect XSD to an Oracle Connect Record Schema

The interaction that is modeled in Oracle Studio is automatically translated to an xsd presentation in Oracle Application Server ProcessConnect. The xsd can be viewed by displaying the record in the Native Dataypes window (under Modeling > Datatypes).

The following example shows the Oracle Application Server ProcessConnect xsd for an input record.

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="noNamespace://OS390/Navigator/CustAccount#findCustAccountInput"
targetNamespace="noNamespace://OS390/Navigator/CustAccount#findCustAccountInput"
elementFormDefault="qualified">
  <xsd:element name="findCustAccountInput" type="findCustAccountInput"/>
  <xsd:complexType name="findCustAccountInput">
    <xsd:attribute default="ATCLIENT" name="EYE-CATCH" type="xsd:string"/>
    <xsd:attribute name="ACCT-NUM" type="xsd:string"/>
  </xsd:complexType>
</xsd:schema>
```

The Oracle Connect metadata for the input record, which was the source for the above xsd, is part of the metadata definition:

```
<record name="findCustAccountInput">
  <field name="EYE-CATCH" type="string" default="ATCLIENT"
         size="8" nativeType="string"/>
  <field name="ACCT-NUM" type="string" size="9" nativeType="numstr_u"/>
</record>
```

The Oracle Connect metadata definition can be viewed in Oracle Studio as follows:

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Click the + next to the **Bindings** node.
4. Click the + next to the **NAV** binding node.
5. Click the + next to the **Adapters** node to display the adapters.
6. Right-click the CICS adapter in the **Configuration Explorer** and choose **Edit Metadata** to open the Metadata perspective.
7. Right-click the adapter in the Metadata Explorer and choose **Edit definition**.
8. Choose the Source tab to review the metadata definition.

The metadata definition includes the following parts:

- General information about the adapter
- Information about each interaction, including the names of the input and output record
- The record schemas used to support the interactions, including input and output (response) record schemas

The following example shows the Oracle Application Server ProcessConnect xsd for an output record.

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="noNamespace://OS390/Navigator/CustAccount#findCustAccountResponse"
targetNamespace="noNamespace://OS390/Navigator/CustAccount#findCustAccountRespon
se" elementFormDefault="qualified">
```

```
  <xsd:element name="findCustAccountResponse" type="findCustAccountResponse"/>
  <xsd:complexType name="findCustAccountResponse">
    <xsd:attribute name="RET-CODE" type="xsd:string"/>
    <xsd:attribute name="ACCT-ID" type="xsd:string"/>
    <xsd:attribute name="ACCT-FIRST-NAME" type="xsd:string"/>
    <xsd:attribute name="ACCT-LAST-NAME" type="xsd:string"/>
    <xsd:attribute name="ACCT-BIRTH-DATE" type="xsd:string"/>
    <xsd:attribute name="ACCT-MARRIED" type="xsd:int"/>
    <xsd:attribute name="ACCT-CHILDREN" type="xsd:int"/>
    <xsd:attribute name="ACCT-TOTAL-ASSETS" type="xsd:int"/>
    <xsd:attribute name="ACCT-EMPLOYED" type="xsd:int"/>
    <xsd:attribute name="ACCT-OVERTIME-HOURS" type="xsd:int"/>
    <xsd:attribute name="ACCT-COMMISSION" type="xsd:float"/>
    <xsd:attribute name="ACCT-SALARY" type="xsd:double"/>
    <xsd:attribute name="ACCT-BALANCE" type="xsd:decimal"/>
    <xsd:attribute name="ACCT-VACATION-DAYS" type="xsd:string"/>
  </xsd:complexType>
</xsd:schema>
```

The Oracle Connect metadata for the output record has the following structure:

```
<record name="findCustAccountResponse">
  <field name="RET-CODE" type="string" nativeType="int4"/>
  <field name="ACCT-ID" type="string" size="9" nativeType="numstr_u"/>
  <field name="ACCT-FIRST-NAME" type="string" size="20" nativeType="string"/>
  <field name="ACCT-LAST-NAME" type="string" size="20" nativeType="string"/>
  <field name="ACCT-BIRTH-DATE" type="string" size="8" nativeType="numstr_u"/>
  <field name="ACCT-MARRIED" type="int" nativeType="int2"/>
  <field name="ACCT-CHILDREN" type="int" nativeType="int4"/>
  <field name="ACCT-TOTAL-ASSETS" type="int" nativeType="int8"/>
  <field name="ACCT-EMPLOYED" type="int" nativeType="int2"/>
  <field name="ACCT-OVERTIME-HOURS" type="int" nativeType="int4"/>
  <field name="ACCT-COMMISSION" type="float" nativeType="single"/>
  <field name="ACCT-SALARY" type="double" nativeType="dfloat"/>
  <field name="ACCT-BALANCE" type="numeric" size="9"
         scale="-2" nativeType="decimal"/>
  <field name="ACCT-VACATION-DAYS" type="string" size="3"
         nativeType="numstr_s"/>
</record>
```

5 Oracle Application Server Integration Adapter for Tuxedo Features and Restrictions

After the Oracle Application Server Integration Adapter for Tuxedo is installed and the modeling described in Chapter 3, "Modeling Interactions for Oracle Application Server Integration Adapter for Tuxedo" and Chapter 4, "Using Interactions in a Business Process" completed, you can use the Oracle Application Server Integration Adapter for Tuxedo to invoke interactions that access programs under Tuxedo.

This chapter contains the following sections:

- Tuxedo Message Buffer Support
- Sending Inbound Interactions Using a Tuxedo Service
- High Availability with Oracle Connect
- Known Restrictions

Tuxedo Message Buffer Support

Oracle Connect for Tuxedo provides support for all standard types of Tuxedo’s message buffers as Input/Output:

STRING - A null terminated character array

CARRAY - An array of un-interpreted arbitrary binary data.

XML - An XML formatted data

VIEW - A "C" structure layout.

VIEW32 - A "C" structure layout where 32-bit FML identifiers are used.

FML - A special Tuxedo system type that provides transparent data portability.

FML32 - An FML type where 32-bit FML identifiers are used.

> **Note:** Synonyms such as X_C_TYPE, X_OCTET are also recognized.

Unstructured message buffers are wrapped within a record as follows:

- A message buffer of type STRING is wrapped within a record containing a single field of type string.
- A message buffer of type CARRAY is wrapped within a record containing a single field of type binary with a fixed size.
- A message buffer of type XML is wrapped within a record containing a single field of type xml.

Sending Inbound Interactions Using a Tuxedo Service

The Oracle Application Server Integration Adapter for Tuxedo includes a set of APIs that enable sending a native event as an inbound interaction from a C program.

To use the APIs, you must include the following in the C program:

- The `gap.h` header file - This header file includes all the API declarations
- The `GAP_LOAD` function - This function loads the API functions. The function does not have any parameters

This appendix describes the APIs and functions:

- Connection APIs
- Transaction APIs
- The Execute Function
- The Get Adapter Schema Function
- The Get Event Function
- The Ping Function
- The Get Error Function

Connection APIs

The following functions handle the connection and connection context for an inbound interaction:

- The Connect Function
- The Disconnect Function
- The Reauthenticate Function
- The Clean Connection Function

There are two kinds of connections:

Transient Connection - Transient connections are created for use within a single inbound interaction. A transient connection is disconnected when an inbound interaction ends, or when the connection context changes (that is, with the `connect`, `setConnection` or `disconnect` functions).

Persistent Connection - Persistent connections can persist across multiple inbound interactions or connection context changes. Persistent connections are disconnected upon an explicit `disconnect` function or when a connection idle timeout expires.

The Connect Function

The `Connect` function establishes a new connection context. All the interactions defined take place within a connection context.

Upon a successful connect, a connection context is established and an implicit setConnection is performed with the newly created connection ID. A failed `connect` function leaves the inbound interaction with no connection context (that is, if a connection context was established prior to invoking the `connect` function, that connection context will no longer be in effect).

The function returns an integer of 1 (TRUE) or 0 (FALSE), to determine the success of the function.

Syntax

ACXAPI_CONNECT(char * ServersUrls, char* Username, char* Password, char* Workspace, char* AdapterName, int Persistent, long IdleTimeout, ACXAPI_CONNECT_MODE ConnectMode, char* DefinitionFileName, char* KeyName, char* KeyValue, void **ConnectHandle)

Table 5–1 ACXAPI_CONNECT Parameters

Parameter	Usage	Description
ServersUrls	Input	The address of the server being connected to in the format: *server_name*:*port*. Or: *TCP/IP_address*:*port*. A series of servers, separated by commas, can be specified. The connection is made to the first server, in the list, that is up
Username	Input	The username required by the adapter
Password	Input	The user password required by the adapter
Workspace	Input	Optional. The name of the workspace where the adapter associated to the connection runs. The default workspace is Navigator
AdapterName	Input	The name of the adapter to connect to
Persistent	Input	Persistent connection indication. If *true*, a persistent connection is created. Otherwise a transient connection is created. The default is *false*.
IdleTimeout	Input	A per-connection client idle timeout setting (in seconds). If the client does not use the connection for the specified amount of time, the connection is disconnected by the server and its associated resources released. This setting is limited by the server side maximum idle connection timeout setting
ConnectMode	Input	The mode of the connection: **0** - Immediate connection **1** - Deferred connection - the connection is made when the execute function is performed

Table 5–1 (Cont.) ACXAPI_CONNECT Parameters

Parameter	Usage	Description
DefinitionFileName	Input	The name and path of the local definition used for the adapter. If omitted, the API queries the underlying adapter for the metadata This parameter should be specified, when the adapter is remote
KeyName	Input	The name of the encryption key
KeyValue	Input	The value associated with the encryption key
ConnectHandle	Output	A pointer to the connection. A pointer is always returned when the connection fails. This enables calling the `getError` function to determine what caused the error The `Disconnect` function must always be called to clear the connection handle

The Disconnect Function

The `Disconnect` function destroys the current connection context. All the resources associated with the current connection (persistent or transient) are released.

Syntax

ACXAPI_DISCONNECT(void* ConnectHandle)

Table 5–2 ACXAPI_DISCONNECT Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection

The Reauthenticate Function

The `Reauthenticate` function establishes a new client identity for the active connection. Adapters are not required to support reauthentication.

An adapter that does not support reauthentication is required to produce an exception if required to reauthenticate with an identity different from the current one. Failure of reauthentication prevents further activity on the connection (other than retrying authentication). Once the reauthentication succeeds, the connection's client is authorized based upon the authenticated identity established.

The function returns an integer of 1 (TRUE) or 0 (FALSE), to determine the success of the function.

Syntax

ACXAPI_REAUTHENTICATE(void* ConnectHandle, char* Username, char* Password)

Table 5–3 ACXAPI_REAUTHENTICATE Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection.
Username	Input	The username of the new client using the connection
Password	Input	The user password of the new client using the connection

The Clean Connection Function

The `Clean Connection` function indicates that the client is working with connection pooling and that the connection is being 'soft-closed'. That is, the connection is being placed in a connections pool. The connection is still valid but various resources on it are freed (for example, objects related to local interactions).

The function returns an integer of 1 (TRUE) or 0 (FALSE), to determine the success of the function.

Syntax

ACXAPI_CLEAN_CONNECTION(void* ConnectHandle, int forgetAuthentification)

Table 5–4 ACXAPI_CLEAN_CONNECTION Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection.
forgetAuthentification	Input	Indicates the adapter should forget the authentication information. This behavior is reflected in the adapter metadata

Transaction APIs

Transaction APIs are used in the following scenarios:

Non-transacted operation - The adapter works in auto-commit mode. Work is committed immediately and automatically upon execution. This operation mode is the default operation mode when no transaction APIs are used, or when the `setAutoCommit` function sets auto-commit to true.

Local transaction operation - When auto-commit is set to false, the first interaction starts a transaction that lasts until an explicit commit (using the `transactionCommit` function) or an explicit rollback (using the `transactionRollback` function) occurs. All interactions performed in between are part of that transaction.

> **Note:** 'local' is used here to indicate the scope of the transaction, rather than its location: the local transaction may be running on a remote machine.

The following functions handle transaction operations:

- The Set Autocommit Function
- The Transaction Commit Function
- The Transaction Rollback Function

The Set Autocommit Function

The `Set Autocommit` function sets the auto-commit mode of the connection.

Syntax

ACXAPI_SET_AUTO_COMMIT(void* ConnectHandle, int AutoCommit)

Table 5–5 ACXAPI_SET_AUTO_COMMIT Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection.

Table 5–5 (Cont.) ACXAPI_SET_AUTO_COMMIT Parameters

Parameter	Usage	Description
AutoCommit	Input	New auto-commit mode of the connection. If set to `true`, each interaction immediately commits once executed. The auto-commit mode must be turned off if multiple interactions need to be grouped into a single transaction and committed or rolled back as a unit When auto-commit is reset and no global transaction is in progress, any interaction starts a local transaction. The client is required to use transactionCommit or transactionRollback at the appropriate time to commit or rollback the transaction The auto-commit mode is `true` by default and is reset if a distributed (global) transaction is started

The Transaction Commit Function

The `Transaction Commit` function commits the work done under the global or local transaction.

Syntax

ACXAPI_TRANSACTION_COMMIT(void* ConnectHandle, ACX_XID Xid)

Table 5–6 ACXAPI_TRANSACTION_COMMIT Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection.
Xid	Input	For internal use only. Supply an empty value

The Transaction Rollback Function

The `Transaction Rollback` function rolls back the work done under the (global) transaction.

Syntax

ACXAPI_TRANSACTION_ROLLBACK(void* ConnectHandle, ACX_XID Xid)

Table 5–7 ACXAPI_TRANSACTION_ROLLBACK Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection

Table 5–7 (Cont.) ACXAPI_TRANSACTION_ROLLBACK Parameters

Parameter	Usage	Description
Xid	Input	For internal use only. Supply an empty value

The Execute Function

The `Execute` function executes a given interaction against the application. The function supplies the input record that is required by the application.

Syntax

ACXAPI_EXECUTE(void* ConnectHandle, char* InterationName, void* BufferIn, void* BufferOut, long BufferOutLen)

Table 5–8 ACXAPI_EXECUTE Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection
InteractionName	Input	Name of interaction to execute
BufferIn	Input	A pointer to the input record
BufferOut	Output	For internal use only. Supply an empty value
BufferOutLen	Output	For internal use only. Supply an empty value

The Get Adapter Schema Function

The `Get Adapter Schema` function returns the schema of Oracle Connect for Tuxedo's backend adapter.

Syntax

ACXAPI_GET_ADAPTER_SCHEMA(void* ConnectHandle, void** *Definition)

Table 5–9 ACXAPI_GET_ADAPTER_SCHEMA Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection
Definition	Output	The listing of the adapter definition

The Get Event Function

The `Get Event` function determines the event to wait for and how long to wait. When an event is received, the function returns the results of performing the event.

Syntax

ACXAPI_GET_EVENT(void* ConnectHandle, char* EventName, long Wait, int Keep, void* BufferOut, long BufferOutLen)

Table 5–10 ACXAPI_GET_EVENT Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection
EventName	Input	Name of event to wait for
Wait	Input	Length of time to wait to receive the event, in seconds
Keep	Input	Whether the event should be stored in the repository or deleted once finished. The default is false (to delete the event)
BufferOut	Output	A pointer to the output record
BufferOutLen	Output	The length of the output record

The Ping Function

The `Ping` function returns, in a pingResponse response, information about an active adapter.

Syntax

ACXAPI_PING(void* ConnectHandle, void** OutputStruct)

Table 5–11 ACXAPI_PING Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection
OutputStruct	Output	The return information describing the structure of the adapter

The Get Error Function

The `Get Error` function returns error information.

Syntax

ACXAPI_GET_ERROR(void* ConnectHandle, char* *Error, long *Status)

Table 5–12 ACXAPI_GET_ERROR Parameters

Parameter	Usage	Description
ConnectHandle	Input	A pointer to the connection
Error	Output	The error message returned by the function
Status	Output	The status of the returned error

High Availability with Oracle Connect

You can configure a pool of server processes so that a server process is always available for a client request.

> **See Also:** "Configuring the Daemon for High Availability" on page 7-2 for details about high availability

Known Restrictions

If you encounter incompatibility problems, please contact Oracle Support Services. The following section describes the known restrictions and includes suggestions for dealing with them when possible:

- Data Handling
- Datatype Handling
- User-Defined Datatypes

> **Note:** If you have any questions or concerns about the restrictions, contact Oracle Support Services.

Data Handling

Oracle Connect for Tuxedo handles Tuxedo's data at a record level only. Individual fields cannot be addressed independently.

Datatype Handling

Specifying a value greater than 1 in the COUNT field of Tuxedo's field definition within a VIEW file, translates to array='xx' attribute

Specifying a value in the SIZE field of Tuxedo's field definition within a VIEW file, translates to size='yy' attribute

Specifying C in the FLAG field of Tuxedo's field definition within a VIEW file, imposes generation of an extra leading COUNTER field to hold the actual count value.

Specifying L in the FLAG field of Tuxedo's field definition within a VIEW file, imposes generation of an extra leading LENGTH field to hold the actual length value.

Note: The L flag is applicable for STRING and CARRAY datatypes only.

In the case where a COUNT value greater than 1 is specified for the field, the extra leading LENGTH field is generated as an ARRAY. The array entries should hold the actual length values of the corresponding field array entries.

Specifying a value in the NULL field of Tuxedo's field definition within a VIEW file is not reflected in Oracle Connect for Tuxedo’s backend adapter record.

Tuxedo's restrictions with regard to VIEW fields are preserved.

The following is an example of a VIEW definition:

```
VIEW emp
#TYPE     CNAME             FBNAME          COUNT   FLAG  SIZE   NULL
long      lSalary           SALARY            1      -      -     0
short     nDeptnum          DEPTNUM           1      -      -     0
short     nEmpnum           EMPLOYEE_NUMBER   1      -      -     0
short     nJobcode          JOBCODE           1      -      -     0
string    szMessageText     MESSAGE_TEXT      1      -     80     ""
string    szFirstName       FIRST_NAME        1      -     16     ""
string    szLastName        LAST_NAME         1      -     21     ""
END
```

The following is an example of a field table that defines the FML field IDs:

```
*base 100
```

```
# name           number   type    flags comments
MESSAGE_TEXT          1   string   -     -
#
DEPTNUM             100   short    -     -
EMPLOYEE_NUMBER     101   short    -     -
JOBCODE             102   short    -     -
FIRST_NAME          103   string   -     -
LAST_NAME           104   string   -     -
SALARY              105   long     -     -
```

User-Defined Datatypes

Oracle Connect for Tuxedo does not handle user-defined datatypes.

6

Troubleshooting the Oracle Application Server Integration Adapter for Tuxedo

Troubleshooting Oracle Application Server Integration Adapter for Tuxedo can involve checking both Oracle Application Server ProcessConnect and Oracle Connect for Tuxedo.

This chapter contains the following sections:

- Design-Time Troubleshooting
- Troubleshooting the Daemon

Design-Time Troubleshooting

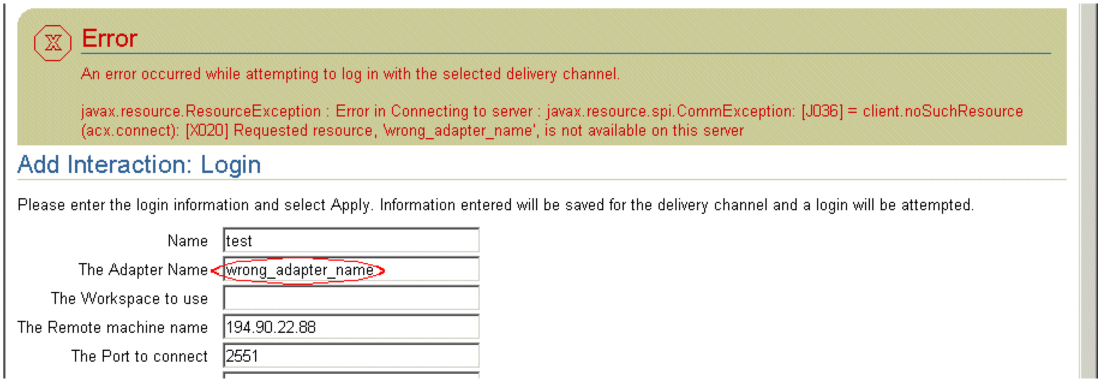
The most common cause of errors when defining the Oracle Application Server Integration Adapter for Tuxedo is incorrect definitions in the delivery channel. If you encounter problems during the modeling phase, check the delivery channel definitions.

> **See Also:** Task: Setting Up the Delivery Channel on page 4-2 for details of setting up the delivery channel

The following errors occur during the modeling of interactions, when using a delivery channel that was defined incorrectly.

Specifying the Wrong Adapter Name

The following error occurs when the wrong adapter name is specified in the delivery channel:



Error

An error occurred while attempting to log in with the selected delivery channel.

javax.resource.ResourceException : Error in Connecting to server : javax.resource.spi.CommException: [J036] = client.noSuchResource (acx.connect): [X020] Requested resource, 'wrong_adapter_name', is not available on this server

Add Interaction: Login

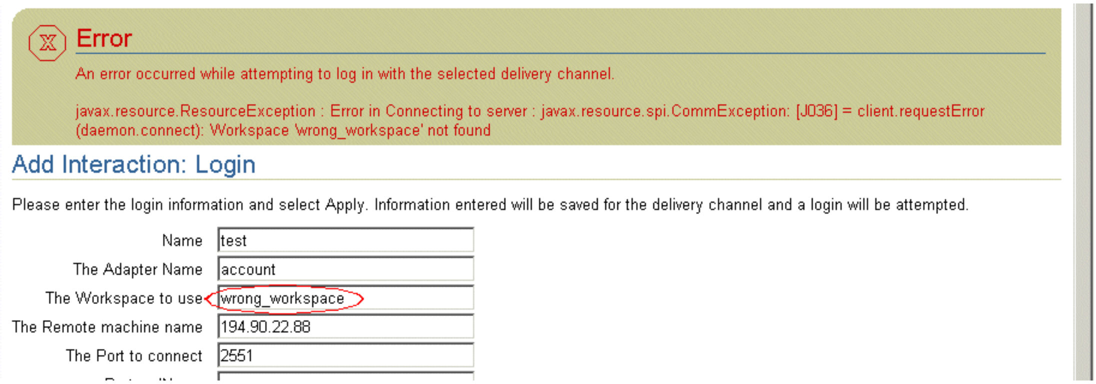
Please enter the login information and select Apply. Information entered will be saved for the delivery channel and a login will be attempted.

Name: test
The Adapter Name: wrong_adapter_name
The Workspace to use:
The Remote machine name: 194.90.22.88
The Port to connect: 2551

Check that the name specified for the adapter is the same as the name specified in the delivery channel.

Specifying the Wrong Workspace Name

The following error occurs when the wrong workspace name is specified in the delivery channel:



Error

An error occurred while attempting to log in with the selected delivery channel.

javax.resource.ResourceException : Error in Connecting to server : javax.resource.spi.CommException: [J036] = client.requestError (daemon.connect): Workspace 'wrong_workspace' not found

Add Interaction: Login

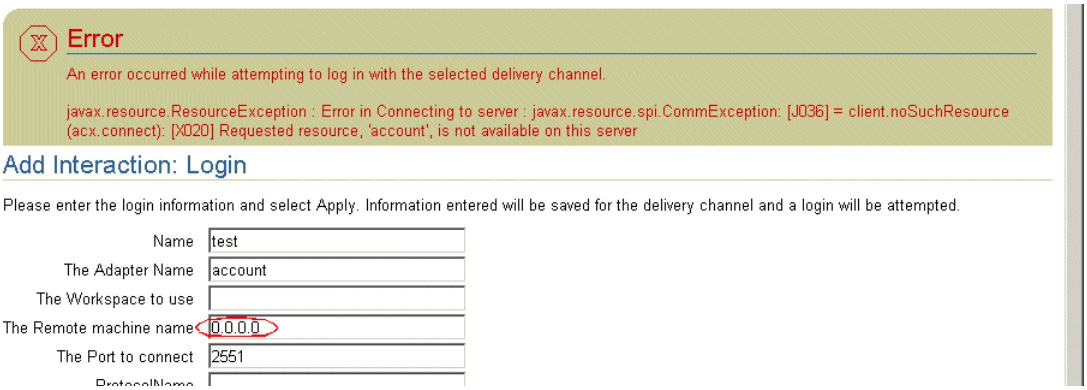
Please enter the login information and select Apply. Information entered will be saved for the delivery channel and a login will be attempted.

Name	test
The Adapter Name	account
The Workspace to use	wrong_workspace
The Remote machine name	194.90.22.88
The Port to connect	2551

Check that the name specified for the adapter is the same as the name specified in the delivery channel.

Specifying the Wrong Machine

The following error occurs when the remote machine is specified incorrectly:



Error

An error occurred while attempting to log in with the selected delivery channel.

javax.resource.ResourceException : Error in Connecting to server : javax.resource.spi.CommException: [J036] = client.noSuchResource (acx.connect): [X020] Requested resource, 'account', is not available on this server

Add Interaction: Login

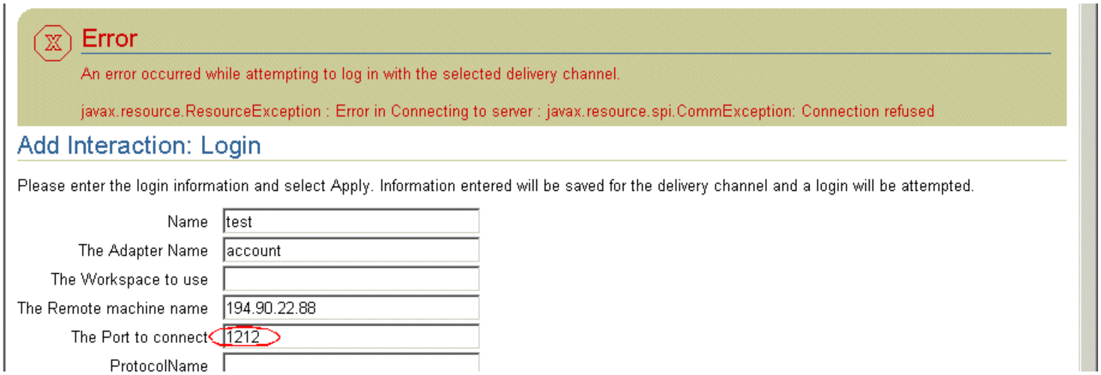
Please enter the login information and select Apply. Information entered will be saved for the delivery channel and a login will be attempted.

Name	test
The Adapter Name	account
The Workspace to use	
The Remote machine name	0.0.0.0
The Port to connect	2551
ProtocolName	

Check the definition of the remote machine in the delivery channel definition.

Specifying the Wrong Port Number

The following error occurs when the port number to connect to is specified incorrectly:



Error

An error occurred while attempting to log in with the selected delivery channel.

javax.resource.ResourceException : Error in Connecting to server : javax.resource.spi.CommException: Connection refused

Add Interaction: Login

Please enter the login information and select Apply. Information entered will be saved for the delivery channel and a login will be attempted.

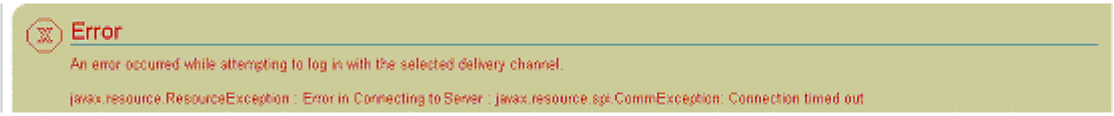
Name	test
The Adapter Name	account
The Workspace to use	
The Remote machine name	194.90.22.88
The Port to connect	1212
ProtocolName	

Check the port number defined in the delivery channel. The default port number is 2552.

When the Connection with the Tuxedo Workstation Machine Is Down

If the delivery channel definitions are correct, the problem is probably with the connection with the Tuxedo workstation machine itself. In this case, check that you have a connection to the Tuxedo workstation machine in Oracle Studio.

For example you receive a connection timeout error when the daemon is not running on the Tuxedo workstation machine:



Error

An error occurred while attempting to log in with the selected delivery channel.

javax.resource.ResourceException : Error in Connecting to Server : javax.resource.spi.CommException: Connection timed out

Troubleshooting the Daemon

A daemon runs on the Tuxedo workstation machine. The daemon is responsible for allocating a server process for a client. The daemon is built as a standard single-threaded RPC server.

The daemon authenticates clients, authorizes requests for a server process within a certain server workspace and provides the clients with the required servers. When a client requests a connection, the daemon allocates a server process to handle this connection, and refers the client to the allocated process. This may be a new process (dedicated to this client) or an already-started process. Further communication between the client session and the server process is direct and does not involve the daemon. The daemon is notified when the connection ends and the process is either killed or remains for use by another client

The daemon supports multiple server configurations called workspaces. Each workspace defines accessible data sources, applications, environment settings, security requirements and server allocation rules. The allocation of servers by the daemon is based on the workspace that the client uses to access the data source. Thus, a client can access a data source using one workspace, where a server process is allocated from an existing pool of servers, or the client can access a data source using a different workspace, where a new server process is allocated for each client request.

The daemon loads configuration settings such as various operational parameters as well as the list of workspaces accessible through the daemon.

> **See Also:** Appendix C, "Daemon Configuration Settings" for details about the configuration settings

Troubleshooting the daemon and the communication between Oracle Application Server ProcessConnect and Oracle Connect for Tuxedo is performed using Oracle Studio to monitor the daemon and server activity and control what happens to the daemon and server processes.

This section contains the following sections:

- Starting the Daemon
- Shutting Down the Daemon
- Monitoring the Daemon During Runtime
- Daemon Logs
- Resolving Communication Errors

Starting the Daemon

The daemon is started when the Oracle Application Server Integration Adapter for Tuxedo is installed. If for any reason you shut down the daemon, as described in "Shutting Down the Daemon" on page 6-6, you can restart the daemon as described in the following task.

Note: The daemon is started on the platform where the Tuxedo workstation runs only. It cannot be started remotely using Oracle Studio.

Task: Starting the Daemon

- Enter the following command:

```
irpcd [-u username [-p password]] -l ip:2552 start
```

where *username* is the name of a user with permission to start the daemon, *password* is the password for this user and *ip* is the ip address of the machine.

Shutting Down the Daemon

If you need to shut down the daemon for any reason (for example, you want to re-IPL the system), use Oracle Studio.

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Right-click the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11 in the **Configuration Explorer** and choose **Open Runtime Perspective** from the popup menu.
4. In the Runtime Explorer, right-click the machine and choose **Shutdown Daemon** from the popup menu.

Monitoring the Daemon During Runtime

Use Oracle Studio to monitor the daemon during runtime.

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Right-click the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11 in the **Configuration Explorer** and choose **Open Runtime Perspective** from the popup menu.

You can manage the daemon by right-clicking the relevant node: daemon, workspace or server process.

Daemon (Machine) 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 machine level:

Status - Checks the status of the daemon. The information about the daemon includes the name of the daemon configuration used, the active client sessions and logging information.

Reload Configuration - Reloads the configuration after any changes. Any servers currently started are not effected by the changed configuration.

> **See Also:** Appendix C, "Daemon Configuration Settings" for details about the configuration settings

View Log - View the daemon log. For details see Daemon Logs on page 6-9.

View Events - For future use.

Daemon Properties - Displays information about the machine where the daemon is running, such as the physical address and any username and password needed to access the machine.

Shutdown Daemon - Shuts down the daemon on the machine.

Rename - Enables changing the name of the daemon displayed in the tree.

Remove - Removes the machine from the perspective tree.

Edit Daemon Configuration - Open the daemon editor to enable you to reconfigure the daemon.

> **See Also:** Appendix C, "Daemon Configuration Settings" for details about the configuration settings

Refresh - Refreshes the display.

Workspace Options

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

The following options are available at the workspace level:

Status - Checks the status of the workspace, whether it is available or not.

Reload Configuration - Reloads the configuration of the workspace after any changes. Any servers currently started are not effected by the changed configuration.

View Log - View the log for all servers for the workspace. For details see Daemon Logs on page 6-9.

View Events - For future use.

End Unused Servers - Ends all server processes for the workspace that are not active. All available and unconnected servers are terminated and any connected servers are marked and terminated on release.

Use this option when changes to the daemon configuration were made after servers were started up. On the next operation, servers are restarted, based on the new settings.

End All Servers - Ends all the active server processes for the specified workspace, regardless of whether the server has an active client.

Edit Workspace Configuration - Open the daemon editor to enable you to reconfigure the workspace.

> **See Also:** Appendix C, "Daemon Configuration Settings" for details about the configuration settings

Refresh - Refreshes the display.

Server Options

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

The following options are available at the server level:

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 - View the server log. For details see Daemon Logs on page 6-9.

View Events - For future use.

End - Ends the server process.

Refresh - Refreshes the display.

Daemon Logs

Oracle Connect for Tuxedo produces a number of logs that you can use to troubleshoot problems. The daemon manages the following logs:

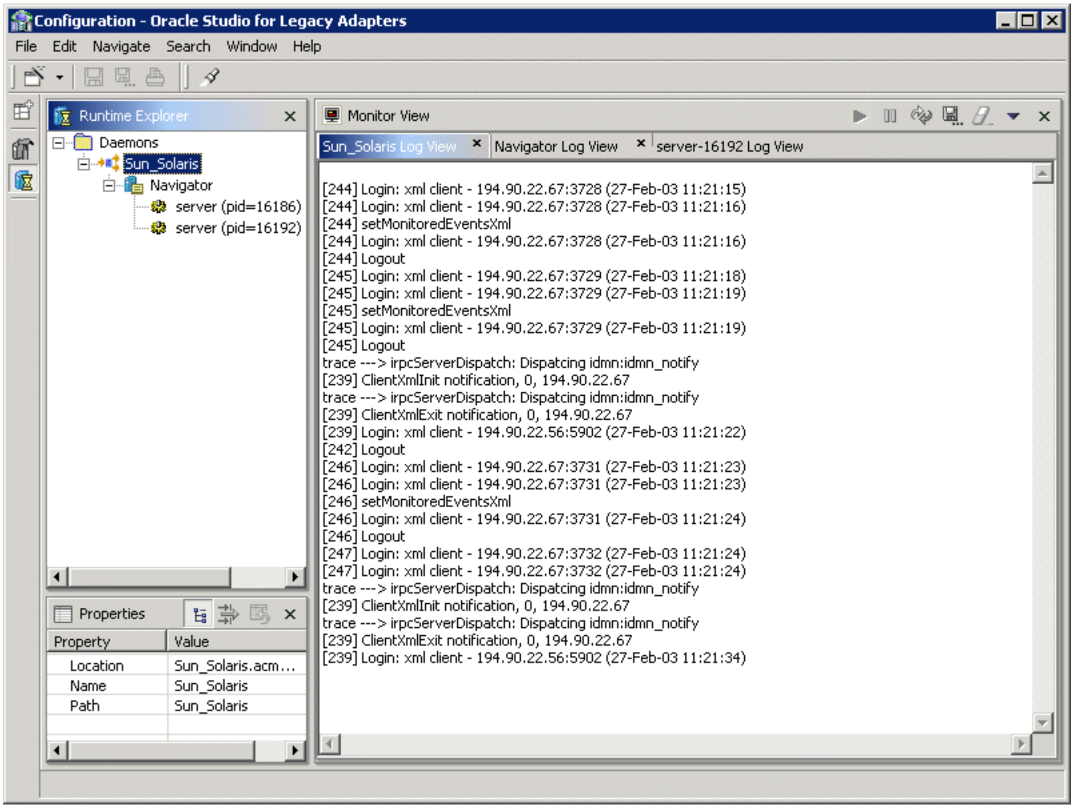
- The daemon log
- The workspace log
- The server process log

The Runtime Manager perspective of Oracle Studio provides a monitor for these logs.

Display the required log by right-clicking the level you want (daemon, workspace or server) and choosing the View Log option.

The Daemon Log Monitor

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



```
[244] Login: xml client - 194.90.22.67:3728 (27-Feb-03 11:21:15)
[244] Login: xml client - 194.90.22.67:3728 (27-Feb-03 11:21:16)
[244] setMonitoredEventsXml
[244] Login: xml client - 194.90.22.67:3728 (27-Feb-03 11:21:16)
[244] Logout
[245] Login: xml client - 194.90.22.67:3729 (27-Feb-03 11:21:18)
[245] Login: xml client - 194.90.22.67:3729 (27-Feb-03 11:21:19)
[245] setMonitoredEventsXml
[245] Login: xml client - 194.90.22.67:3729 (27-Feb-03 11:21:19)
[245] Logout
trace ---> irpcServerDispatch: Dispatcing idmn:idmn_notify
[239] ClientXmlInit notification, 0, 194.90.22.67
trace ---> irpcServerDispatch: Dispatcing idmn:idmn_notify
[239] ClientXmlExit notification, 0, 194.90.22.67
[239] Login: xml client - 194.90.22.56:5902 (27-Feb-03 11:21:22)
[242] Logout
[246] Login: xml client - 194.90.22.67:3731 (27-Feb-03 11:21:23)
[246] Login: xml client - 194.90.22.67:3731 (27-Feb-03 11:21:23)
[246] setMonitoredEventsXml
[246] Login: xml client - 194.90.22.67:3731 (27-Feb-03 11:21:24)
[246] Logout
[247] Login: xml client - 194.90.22.67:3732 (27-Feb-03 11:21:24)
[247] Login: xml client - 194.90.22.67:3732 (27-Feb-03 11:21:24)
trace ---> irpcServerDispatch: Dispatcing idmn:idmn_notify
[239] ClientXmlInit notification, 0, 194.90.22.67
trace ---> irpcServerDispatch: Dispatcing idmn:idmn_notify
[239] ClientXmlExit notification, 0, 194.90.22.67
[239] Login: xml client - 194.90.22.56:5902 (27-Feb-03 11:21:34)
```

Property	Value
Location	Sun_Solaris.acm...
Name	Sun_Solaris
Path	Sun_Solaris

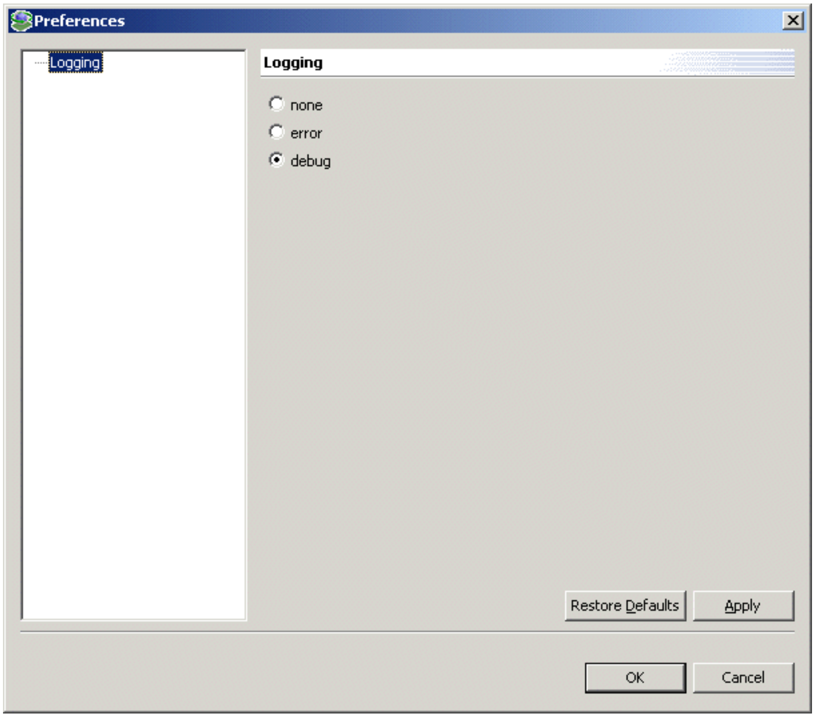
You can change the level of logging by clicking the Update Monitor button on the toolbar. The following levels of logging are available:

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 C-4 for details



Logging

○ none
○ error
● debug

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 the Update Monitor button on the toolbar. The following levels of logging are available:

none - The log displays who has connected and disconnected from the server process.

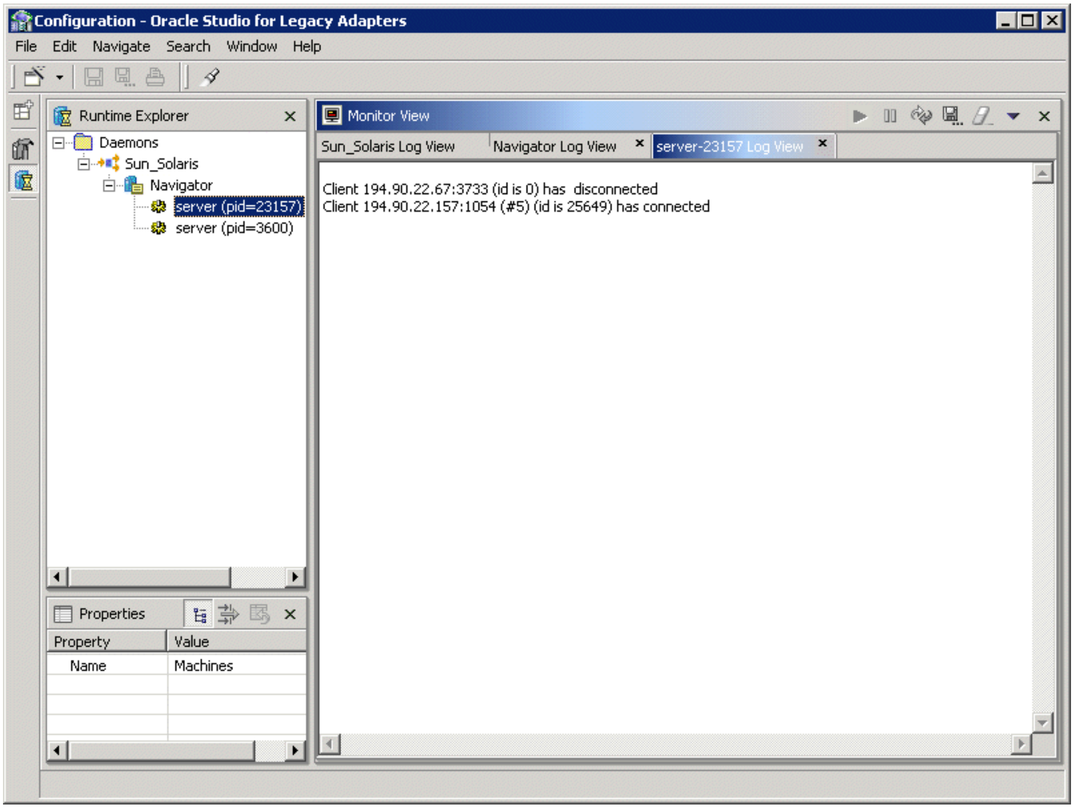
error - The log displays who has connected and disconnected from the server process and any errors that have been generated.

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:** "WS Logging" on page C-16 for details

The Server Log Monitor

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



```
Client 194.90.22.67:3733 (id is 0) has  disconnected
Client 194.90.22.157:1054 (#5) (id is 25649) has connected
```

You can change the level of logging by clicking the Update Monitor button on the toolbar. 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: "WS Logging" on page C-16 for details

The following table describes the log monitor toolbar buttons and their functionality.

Table 6–1 Log Monitor Toolbar Buttons

Button	Name	Function
▶	Resume	Resumes monitoring the machine after the monitor has been suspended. This button toggles with the suspend button.
⏸	Suspend	Suspends the monitor. This button toggles with the resume button.
⟳	Update Monitor	Updates the information being monitored.
💾	Save as	Enables specifying a name and location for saving the log.
⌫	Clear view	Clears the monitor view.
▼	Menu	Displays a menu with the options available using the buttons.

Resolving Communication Errors

When Oracle Studio disconnects from the Tuxedo machine, the machine is displayed in Oracle Studio with an X in a red circle. If this situation occurs, try to access the machine later.

The following table describes the various scenarios that may exist when Oracle Application Server ProcessConnect disconnects from the platform where the Tuxedo workstation runs.

Table 6–2 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 is dead 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 need to be familiar with the following:

- The daemon configuration settings
- Oracle Connect for Tuxedo security.
- The TCP/IP subsystem. Oracle Application Server Integration Adapter for Tuxedo 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 machines involved and whether a portmapper is being used.

Resolving Specific Errors

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

C005: Could not open the IRPCD log file for write.

Cause: The daemon was not able to create or write to its log file. The log file is viewed using the Oracle Studio Runtime perspective, as described in "Daemon

Logs" on page 6-9. The log file location is set in the daemon configuration as described in "Daemon Logging" on page C-4.

Action: Check that the account where the daemon runs has permission to generate/write to the log file.

Action: Check the path specified for the log file in the daemon configuration.

Action: Check that there is no existing log file owned by another user at the specified location.

Action: Ensure that the disk device is not full.

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 delivery channel for the adapter.

Action: Check the TCP/IP subsystem on the current machine 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 machine with the port specified in the delivery channel 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 delivery channel is also defined in the daemon configuration on the platform where the Tuxedo workstation runs. Use the Status option in the Runtime Manager perspective.

C00B: Invalid IRPCD client context.

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

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

C00C: Daemon request requires a server login.

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

Action: Check the processes and kill 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 username/password.

Action: Edit the user profile in Oracle Studio to specify a valid username and password for the platform where the Tuxedo workstation runs.

See Also: "Setting Up Runtime User Access to the Tuxedo Workstation Machine" on page 2-18 for details

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 username and password.

See Also: "Daemon Security" on page C-7 for details

C00F: Anonymous client logins are not allowed.

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

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

See Also: "Daemon Security" on page C-7 for detail

Action: Edit the user profile in Oracle Studio to specify a valid username and password for the platform where the Tuxedo workstation runs.

See Also: "Setting Up Runtime User Access to the Tuxedo Workstation Machine" on page 2-18 for details

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 WS Info. section of the daemon configuration.

See Also: "WS Info." on page C-9 for details

C012: Invalid username/password.

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

Action: See the daemon log file for the reason that the username/password were not accepted.

Action: Edit the user profile in Oracle Studio to specify a valid username and password for the platform where the Tuxedo workstation runs.

See Also: "Setting Up Runtime User Access to the Tuxedo Workstation Machine" on page 2-18 for details

Action: Make sure the daemon is started from an APF-authorized account that is allowed to check for system usernames 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 `Set maximum number of servers` and/or `Maximum` parameter for the `Clients per server limit` in the WS Server section of the daemon configuration.

See Also: "WS Server" on page C-12 for details

Action: Try running the command later.

C015: Failed to start server process.

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

Action: See the daemon and server log files 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.

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.

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 machine.

Action: Check that the name specified for the machine in the delivery channel 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 machine 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 for Tuxedo 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 for Tuxedo.

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

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

Cause: An Oracle Connect for Tuxedo 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 WS Info. section of the daemon configuration.

See Also: "WS Info." on page C-9 for details

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

Action: Increase the value of the `Set maximum number of servers` and/or `Maximum` parameter for the `Clients per server limit` in the WS Server section of the daemon configuration.

See Also: "WS Server" on page C-12 for details

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 machine 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 machine 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 for Tuxedo 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 port specified in the delivery channel 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 machine 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 for Tuxedo 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 for Tuxedo 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.

C02E: Call made to non-existing procedure %d

Cause: This error typically is caused by a client of a newer version calling an old server.

Action: Verify that the client and server are using the same version of Oracle Connect for Tuxedo.

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 and/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 re-register 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: "Daemon Logging" on page C-4 for details

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

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

Action: Check the remote machine definition in the delivery channel for the adapter.

Action: Check the daemon is up on the platform where the Tuxedo workstation runs. Use the Status option in the Runtime Manager perspective.

Action: Check the network connection by trying to ping the host machine 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 in the WS Info. section of the daemon configuration.

See Also: "WS Info." on page C-9 for details

C047: %s out of memory

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

Action: Kill 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 for Tuxedo daemon or server failed, or an established session with a server has failed.

Action: Check the remote machine definition in the delivery channel.

Action: Check the daemon is up on the platform where the Tuxedo workstation runs. 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 machine 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 an Invalid Instruction).

Action: Contact Oracle Support Services.

C068: Fail to login daemon

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

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 re-install Oracle Connect for Tuxedo on the server.

C069: Fail to get server

Cause: The Oracle Connect for Tuxedo daemon on the server machine 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 for Tuxedo 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 username given is not allowed to use the requested server. Use an authorized username.

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, above.

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 for the reason that the server was not available to accept its assigned client.

Action: If a multi-threaded server is used and many clients are trying to connect to it at the same time, some may get a Connection Refused error if the TCP/IP request queue fills up.

C06B: Disconnecting from server

Cause: A network failure, or a server machine failure or a server program failure caused the connection to abort. The currently active transaction is aborted as well.

Action: Oracle Connect for Tuxedo automatically tries to re-establish a connection with a server upon the next SQL command issued against the server. Once the network or machine failure is corrected, the connection to the daemon is re-established 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, a server machine failure or a server program failure caused the connection to abort. The currently active transaction is aborted as well.

Action: Normally, Oracle Connect for Tuxedo automatically tries to create a new 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 and/or server machine should be fixed before connection can be resumed.

Action: In case of a server crash not related to callable user code, contact Oracle support.

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 via the Administrator parameter of the Daemon Security or WS Security sections in the daemon configuration.

> **See Also:** "Daemon Security" on page C-7 or "WS Security" on page C-18 for details

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.

7

Configuring the Working Environment

Oracle Connect for Tuxedo includes a number of tuning parameters that can improve performance. Specifically, the daemon can be configured to optimize communication between the Tuxedo workstation machine and the client machine and the binding environment can be tuned to optimize the way actual requests are handled.

This chapter contains the following sections:

- Configuring the Daemon for High Availability
- Configuring a Binding Environment

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 a number of daemon workspace configurations. Thus, you can tailor individual workspaces for use with different adapters.

The workspace that will be used is specified in the delivery channel, as described in Setting Up the Delivery Channel on page 4-2.

Task: 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.

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. In the Configuration Explorer, click the + next to the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Click the + next to the **Daemons** node. The daemon configurations available on this machine are listed.
4. Right-click **IRPCD** and select New Workspace from the popup menu.
5. In the New daemon workspace window, specify a name the new workspace and optionally a description.
6. Specify whether you want it to have default settings or to copy the properties of an existing workspace.

   To copy the properties of an existing workspace, click the ellipsis button and select the workspace from which you want to copy the properties.

Once the workspace is defined the daemon editor is displayed, showing the **WS Info** tab.

Task: Editing the Workspace

You edit a workspace using the following tabs:

- **WS Info** - Specifies general information including the server type, the command procedure used to start the workspace, the binding configuration associated with this workspace and the timeout parameters.
- **WS Server Mode** - Specifies workspace server information including features that control the operation of the servers started up by the workspace and allocated to clients.
- **WS Logging** - Specifies parameters for logging and the format to use for a log file.
- **WS Security** - Specifies administration privileges, user access, ports available to access the workspace and workspace account specifications.
- **WS Governing** - For internal use.

Task: Configuring the Server Mode

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

Single Client - 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 (since 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 (since it requires as many server processes as concurrent clients).

Multi-Client - Clients share a server process and are processed serially.

This mode has low overhead since the server processes are already initialized. However, because clients share the same process, they may impact one another, especially if they issue lengthy queries.

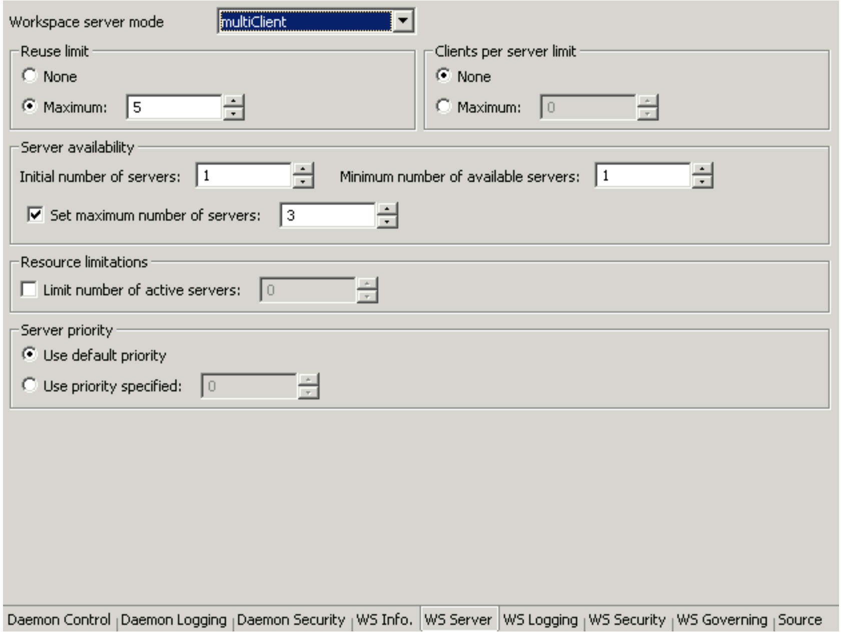
The number of clients that share a process is determined by the `Clients per server limit` (the maximum number of concurrent clients a server process for the current workspace accepts).

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 since the servers are already initialized. However, this server mode may use a lot of server resources (since 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 since it eliminates the need to repeat initializations. However, reuse runs a risk of higher memory leakage over time. The default for the `Reuse limit` field value is **None**, indicating that no reuse limit is enforced.

Set the server mode in the WS Server tab of the daemon editor:



Workspace server mode: multiClient

Reuse limit
○ None
● Maximum: 5

Clients per server limit
● None
○ Maximum: 0

Server availability
Initial number of servers: 1
Minimum number of available servers: 1
☑ Set maximum number of servers: 3

Resource limitations
□ Limit number of active servers: 0

Server priority
● Use default priority
○ Use priority specified: 0

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 WS Server tab.

Initial number of servers - The number of server processes that are prestarted for this workspace when the daemon starts up. 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 below 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.

Minimum number of available servers - 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 above). If this parameter is set to a value greater than that of 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, meaning that new servers are created only when there are no other available servers.

Set maximum number of servers - The maximum number of available server processes pooled for this workspace. If the server is reusable, once a client disconnects from the server, the daemon returns the server to the pool of available servers. If the limit is reached, excess server processes are discarded.

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 machine
- Inbound queues for adapters on the Tuxedo workstation machine. (Inbound queues are referred to as events in Oracle Studio)

Configuring data sources, adapters and inbound queues are described in Chapter 3, "Modeling Interactions for Oracle Application Server Integration Adapter for Tuxedo". The following section describes the environment settings.

Task: Setting a Binding Environment

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. In the Configuration Explorer, click the + next to the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Click the + next to the **Bindings** node. The binding configurations available on this machine are listed.
4. Right-click **NAV** and select Edit Binding.
5. In the Properties tab edit the environment settings as needed.

To edit an environment setting, expand the property category and click the value to edit. The binding environment is divided into the following categories:

- comm Category
- debug Category
- miscellaneous Category
- odbc Category
- oledb Category
- optimizer Category
- queryProcessor Category
- transactions Category
- tuning Category

comm Category

The following parameters define communication buffers:

comCacheBufferSize - The size of a memory buffer on a client, which is used by the Oracle Connect client/server to store read-ahead data. The default is 200000 bytes.

comMaxSocketSize - The maximum bytes that can be written in one chunk on a socket. The default is -1 (no limitation).

comMaxXmlInMemory - The maximum size of an XML document held in memory. The default is 65535 bytes.

comMaxXmlSize - The maximum size of an XML document passed to another machine. The default is 65535 bytes.

debug Category

The following parameters define debugging and logging operations.

acxTrace - When set to "true", the input xml sent to the backend adapter and the output xml returned by the backend adapter, are written to the log.

analyzerQueryPlan - For internal use.

gdbTrace - For internal use.

generalTrace - When set to "true", logs general trace information. The default writes only error messages to the log.

logFile - The high-level qualifier of the log file for messages. The following types of message are written to the log:

- Error messages
- Trace information and information about the query optimization strategy if generalTrace is set to "true"

oledbTrace - For internal use.

optimizerTrace - For internal use.

queryWarnings - For internal use.

traceDir - For internal use.

miscellaneous Category

The following parameters define miscellaneous operations, including national language support (NLS) and the directory where temporary files are written.

codepage - For use with National Language Support (NLS) to identify the codepage for the workspace.

> **See Also:** Appendix D, "National Language Settings"

cvtSeverityLevel - The data type conversion policy when a conversion error occurs:

- 0 (Default) - The data in the output column will be a null or empty value.
- 1 - The data in the output column will be a null or empty value and the error is reported to the log.
- 2 - An error is reported and processing stops.

edit - For internal use.

language - Identifies the application language. A default codepage is selected based om the value specified for this parameter.

> **See Also:** Appendix D, "National Language Settings"

nlsString - Specifies the codepage used by a field whose data type is defined as "nlsString". You use this for a field whose codepage is other than that of the machine's codepage. This parameter includes the following values:

- The name of the codepage.
- Whether the character set reads from right to left (as in middle eastern character sets). The default is "false".

tempDir - 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.

year2000Policy - Determines the way 2-digit years are converted into 4-digit years. When the parameter year2000Policy is not set, or when it is set to a value outside the range of values defined for the policy, as described below, 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 4-digit year after 1900 that can be represented by a 2-digit year. For example, if year2000Policy is set to 1905, the years 2000->2004 will be represented by 00->04. All other 2 digits will map to 19xx.
>
> This solution is most appropriate if there is live data at the low end (close to the year 1900), which the user wants to keep with the current 2-digit format.
>
> The user will probably change 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 2-digit number. With each passing year the earliest year that can be represented by a 2-digit number changes to a year later.

odbc Category

The odbc parameters are for internal use.

oledb Category

The oledb parameters are for internal use.

optimizer Category

The optimizer parameters are for internal use.

queryProcessor Category

The queryProcessor parameters are for internal use.

transactions Category

The transactions parameters are for internal use.

tuning Category

The tuning parameters are for internal use.

A

The Oracle Connect Tuxedo Interaction Metadata Definitions

Oracle Studio includes a wizard to help define the adapter inbound and outbound interactions and input and output structures used for these interactions. The interactions and input and output structures are maintained as metadata in the Metadata perspective of Oracle Studio.

This appendix contains the following section:

- Interaction Metadata for the Adapter

Interaction Metadata for the Adapter

Use Oracle Studio to maintain the metadata for the Tuxedo adapter.

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Click the + next to the **Bindings** node.
4. Click the + next to the **NAV** binding node.
5. Click the + next to the **Adapters** node to display the adapters.
6. Right-click the Tuxedo adapter in the **Configuration Explorer** and choose **Edit Metadata** to open the Metadata perspective.
7. Right-click the interaction in the Metadata Explorer you want to view and choose **Edit**.

The metadata editor opens, displaying the **General** tab, with general table details. The following tabs are used to edit the metadata:

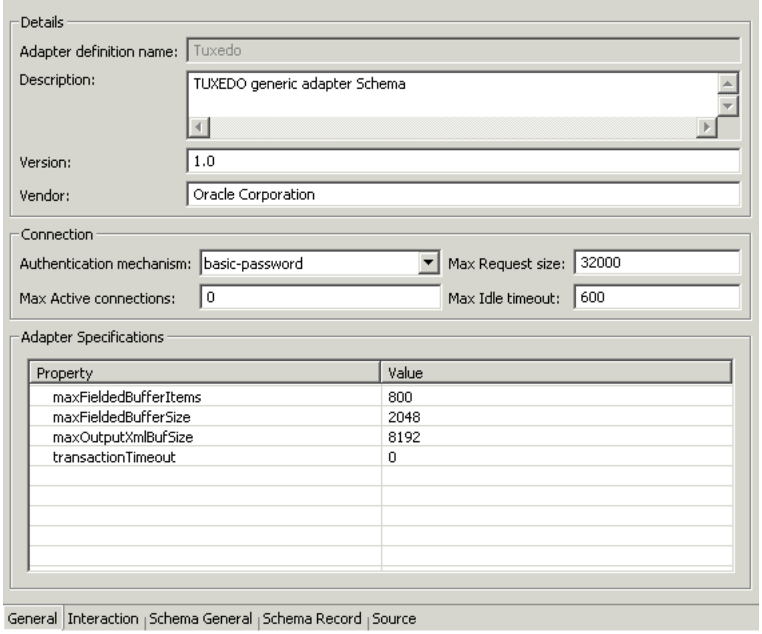
- General Tab
- Interaction Tab
- Schema General Tab
- Schema Record Tab

The following tab provides an XML representation of the metadata:

- Source Tab

General Tab

Use the **General** tab to maintain information generally describing the adapter and the connection to it.



Details

Adapter definition name: Tuxedo

Description: TUXEDO generic adapter Schema

Version: 1.0

Vendor: Oracle Corporation

Connection

Authentication mechanism: basic-password

Max Request size: 32000

Max Active connections: 0

Max Idle timeout: 600

Adapter Specifications

Property	Value
maxFieldedBufferItems	800
maxFieldedBufferSize	2048
maxOutputXmlBufSize	8192
transactionTimeout	0

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

Adapter definition name - The name of the adapter definition.

Description - An identifying description of the adapter.

Version - The adapter version.

Vendor - The vendor who supplied the adapter.

Authentication mechanism - The authentication to access the adapter. The available mechanisms are:

- kerbv5
- none
- basic-password

Max request size - The maximum size in bytes for an XML request or reply. Larger messages are rejected with an error.

Max active connections - The maximum number of simultaneous connections for an adapter (per process).

Max idle timeout - The maximum time, in seconds, that an active connection can stay idle. After that time, the connection is closed.

Adapter Specifications - Adapter-specific properties for an outbound interaction.

maxFieldedBufferItems - Set a limit to the number of items that a fielded buffer may hold. The default is 800.

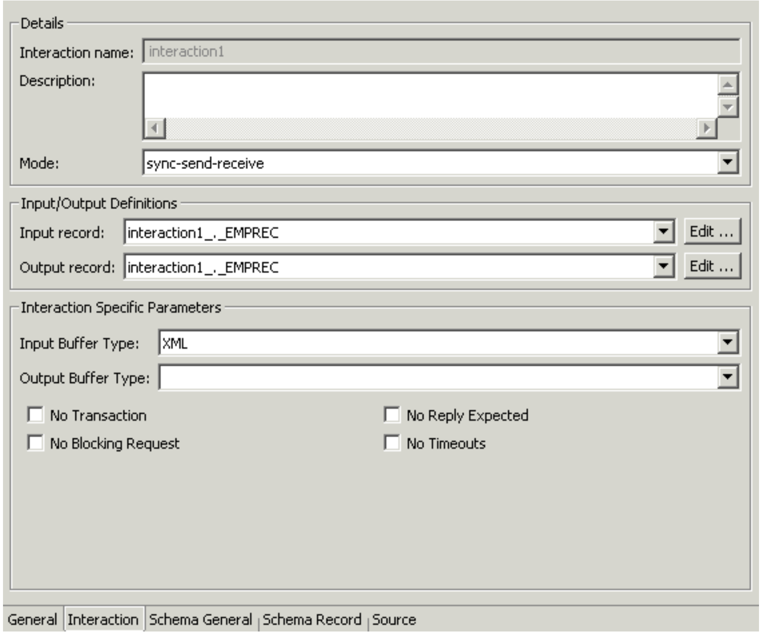
maxFieldedBufferSize - Limits the total size of a message buffer of type FML. The default is set to 2K bytes.

maxOutputXmlBufSize - Controls the size of memory allocated for output message buffers of type XML. The default taken as 8K bytes.

transactionTimeout - Controls the duration of a Tuxedo transaction (in seconds). If not set (or set to 0), the Tuxedo transaction will not be interrupted and will last until explicitly terminated. This is the default.

Interaction Tab

Use the **Interaction** tab to define the general details of the interaction in addition to its input and output definitions.



Details

Interaction name: interaction1

Description:

Mode: sync-send-receive

Input/Output Definitions

Input record: interaction1_._EMPREC

Output record: interaction1_._EMPREC

Interaction Specific Parameters

Input Buffer Type: XML

Output Buffer Type:

□ No Transaction

□ No Blocking Request

□ No Reply Expected

□ No Timeouts

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

Interaction name - The name of the interaction.

Description - A descriptive identifier for the interaction.

Mode - The interaction mode:

- **sync-send-receive** - The outbound interaction sends a request and expects to receive a response.
- **sync-send** - The outbound interaction sends a request and does not expect to receive a response.
- **sync-receive** - The outbound interaction expects to receive a response.
- **async-send** - The inbound interaction sends a request and does not expect to receive a response.

Input record - Identifies an input record.

Output record - Identifies an output record for the results of an outbound interaction.

Interaction Specific Parameters - Defines the properties and values of parameters specific to an outbound interaction.

Input Buffer Type - The type of data used for the input.

Output Buffer Type - The type of the buffer to use for the results of an outbound interaction.

No Transaction - Enables a service to be executed, regardless of transaction context.

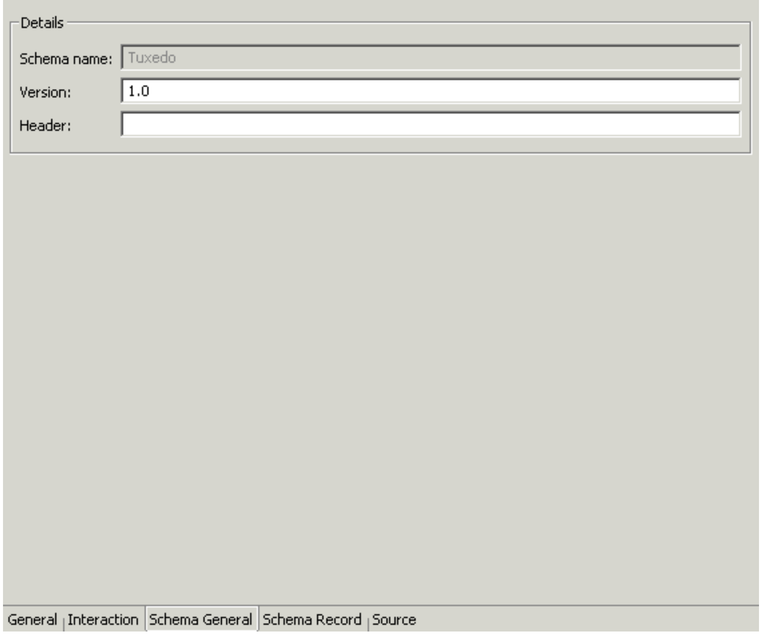
No Reply Expected - For future use.

No Blocking Request - Avoids a FROM request submission if a blocking condition exists.

No Timeouts - Ignores blocking timeouts.

Schema General Tab

Use the **Schema General** tab to define the general details of the input and output record structures for the interaction.



Details

Schema name: Tuxedo

Version: 1.0

Header:

The **Schema General** tab has the following fields:

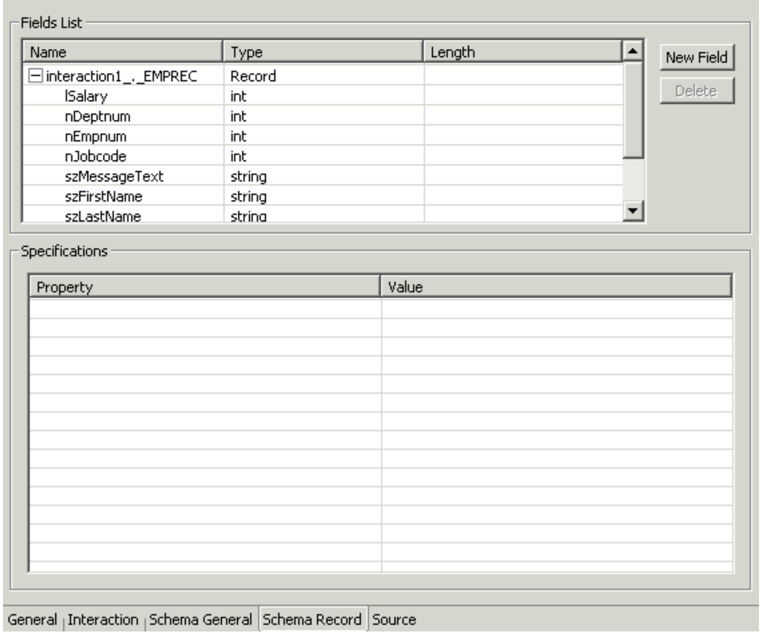
Schema name - The name of the adapter.

Version - The schema version.

Header - A C header, file to map between the data structure and the adapter.

Schema Record Tab

Use the **Schema Record** tab to define the input and output record structures for the interaction.



Name	Type	Length
interaction1_._EMPREC	Record	
lSalary	int	
nDeptnum	int	
nEmpnum	int	
nJobcode	int	
szMessageText	string	
szFirstName	string	
szLastName	string	

Property	Value

The **Schema Record** tab has the following fields:

The **Fields List** defines single data items within a record.

Name - The name of the field.

Type - The datatype of the field. The following are valid datatypes:

Binary

Boolean

Date

Double

Float

Int

Long

Numeric

Short

String

Time

Timestamp

Xml

See Also: Appendix B, "Datatype Support" for details about the mapping from COBOL datatypes to Oracle Connect for Tuxedo datatypes

Length - The size of the field including a null terminator, when the datatype supports null termination.

Use the **Specifications** box to specify field properties.

Source Tab

The **Source** tab displays the XML representation of the adapter metadata.

B

Datatype Support

The Oracle Application Server Integration Adapter for Tuxedo supports a number of datatypes that are used when defining metadata in Oracle Studio. The datatypes are mapped from the Tuxedo datatypes during the import procedure.

> **Note:** The mapping of datatypes between Oracle Application Server Integration Adapter for Tuxedo and Oracle Application Server ProcessConnect is performed internally by Oracle Connect for Tuxedo.

This appendix contains the following section:

- Datatype Mapping

Datatype Mapping

Table B–1 Datatype Mapping: Tuxedo and Oracle Connect for Tuxedo

Tuxedo Datatype	Oracle Connect Datatype
carray	string
char	string
double	double
float	double
int	int
long	int
short	int
string	string

C

Daemon Configuration Settings

The daemon runs on the platform where the Tuxedo workstation runs and is responsible for allocating a server process for a client. The daemon is built as a standard single-threaded RPC server. When started, the daemon loads configuration settings such as various operational parameters as well as the list of workspaces accessible through the daemon.

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

This appendix contains the following sections:

- Daemon Control
- Daemon Logging
- Daemon Security
- Workspaces

Daemon Control

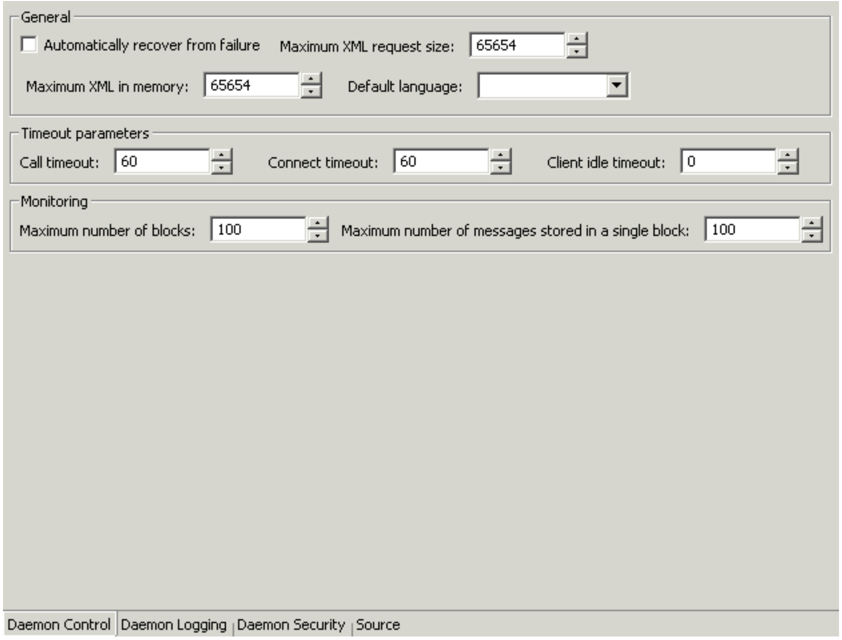
The **Daemon Control** section specifies various control options.

The **Daemon Control** section is accessed as follows:

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Right-click the machine and choose **Open Runtime Perspective**.
4. Right-click the daemon in the **Runtime Explorer** and choose **Edit Daemon Configuration**. The **Daemon Control** tab opens.
5. After making changes to the daemon, right-click the daemon and choose **Reload Configuration**.

Note: You can also change daemon settings in the using the **Configuration Explorer**, by selecting a machine and drilling-down the list until the daemon is displayed. Right-click the daemon and choose **Edit Daemon**.

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



General

☐ Automatically recover from failure

Maximum XML request size: 65654

Maximum XML in memory: 65654

Default language:

Timeout parameters

Call timeout: 60

Connect timeout: 60

Client idle timeout: 0

Monitoring

Maximum number of blocks: 100

Maximum number of messages stored in a single block: 100

The following fields are displayed:

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.

Maximum XML in memory - The maximum amount of space reserved for the XML in memory.

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.

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, the connection is aborted. 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, 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.

Specifying the timeout in a workspace overrides the value set in this field for that workspace.

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.

Daemon Logging

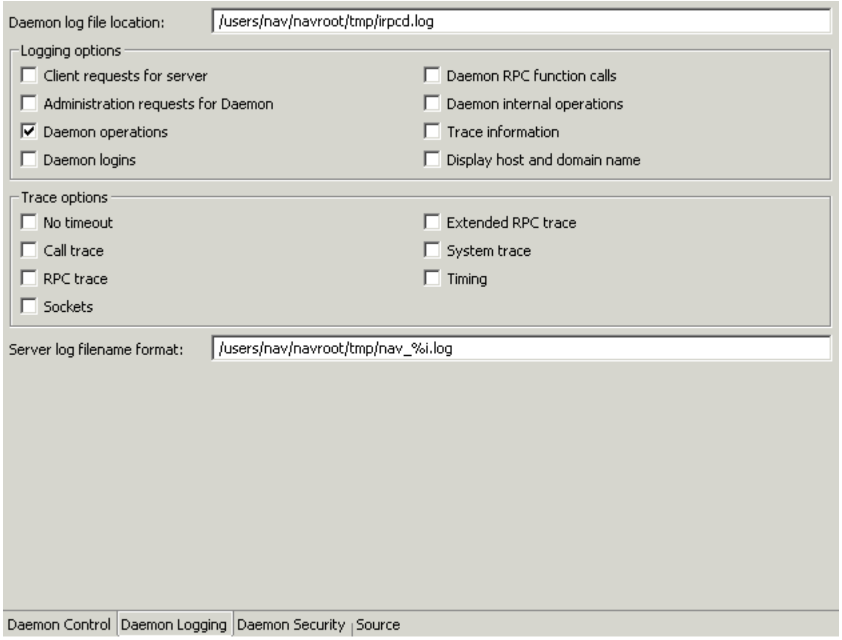
The **Daemon Logging** section defines the daemon log file settings, the log file structure and the location where the log is saved. In addition it defines the data that is logged and traced in the file.

The **Daemon Logging** section is accessed as follows:

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Right-click the machine and choose **Open Runtime Perspective**.
4. Right-click the daemon in the **Runtime Explorer** and choose **Edit Daemon Configuration**.
5. Choose the **Daemon Logging** tab.
6. After making changes to the daemon, right-click the daemon and choose **Reload Configuration**.

Note: You can also change daemon settings in the using the **Configuration Explorer**, by selecting a machine and drilling-down the list until the daemon is displayed. Right-click the daemon and choose **Edit Daemon**.

Changes made to the daemon configuration in the way are not implemented. They are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager.



Daemon log file location: /users/nav/navroot/tmp/irpcd.log

Logging options

□ Client requests for server	□ Daemon RPC function calls
□ Administration requests for Daemon	□ Daemon internal operations
☑ Daemon operations	□ Trace information
□ Daemon logins	□ Display host and domain name

Trace options

□ No timeout	□ Extended RPC trace
□ Call trace	□ System trace
□ RPC trace	□ Timing
□ Sockets	

Server log filename format: /users/nav/navroot/tmp/nav_%i.log

The following fields are displayed:

Daemon log file location - Where the daemon produces its log data. The full path must be specified.

Logging options - Specifies what tracing is performed.

> **Client requests for server** - Logs client requests for server activations; this provides logging of the process IDs of the started servers along with the location of the log files.

Administration requests for daemon - Logs all of the administration requests for the daemon.

Daemon operations - Logs all of the daemon operations.

Daemon logins - Logs daemon logins.

Daemon RPC function calls - Log all daemon RPC function calls.

Daemon internal operations - Log daemon internal operations.

Log trace information - Logs low-level RPC operations.

Display host and client domain name - Whether the client host and domain name are logged rather than the client IP address. The default is false.

Trace options - Specifies what tracing is performed.

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.

Sockets - Generates a message in the server log file for each socket operation.

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 timestamp for every entry to the server log file.

Server log filename format - Defines the name and location of the server log file. The field must specify the full pathname. If no directory information is provided for the log file, it will be located in the login directory of the account running Oracle Connect for Tuxedo workstation.

The following tokens can appear in the log file template and will be 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)

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`.

Daemon Security

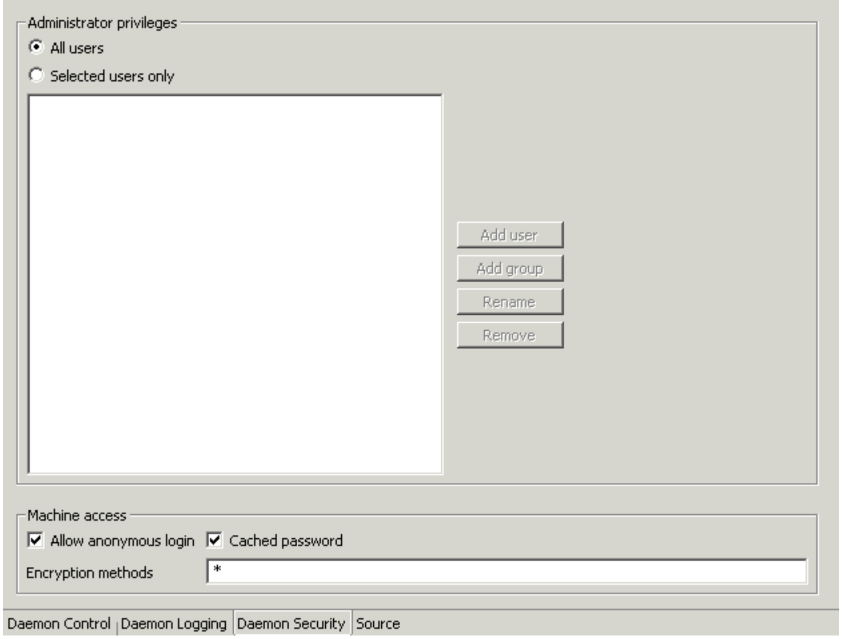
The **Daemon Security** section is used for the following:

- To grant administration rights for the daemon
- To determine access to the machine

The **Daemon Security** section is accessed as follows:

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Right-click the machine and choose **Open Runtime Perspective**.
4. Right-click the daemon in the **Runtime Explorer** and choose **Edit Daemon Configuration**.
5. Choose the **Daemon Security** tab.
6. After making changes to the daemon, right-click the daemon and choose **Reload Configuration**.

> **Note:** You can also change daemon settings in the using the **Configuration Explorer**, by selecting a machine and drilling-down the list until the daemon is displayed. Right-click the daemon and choose **Edit Daemon**.
>
> Changes made to the daemon configuration in the way are not implemented. They are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager.



Administrator privileges

● All users

○ Selected users only

Machine access

☑ Allow anonymous login ☑ Cached password

Encryption methods *

The following fields are displayed:

Administrators privileges - Identifies the users (accounts) allowed to perform administrative tasks (tasks that require administrative login).

All users - Anyone can access the daemon and change the settings.

Selected users only - The names of users (accounts) and groups that can be administrators.[1]

Note: 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 machine.

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

Anonymous login allowed - Whether workspaces allow anonymous logins (without user name/password entries). For the optimal level of security, keep this option unchecked and define a username for the Daemon Administrators parameter.

If unchecked, no workspace can have an anonymous client. If checked, a particular workspace allows anonymous clients.

Cached passwords - 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 - The encryption method being 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.

Workspaces

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

- WS Info.
- WS Server
- WS Logging
- WS Security
- WS Governing - for internal use

Warning: You must not delete the Navigator workspace.

WS Info.

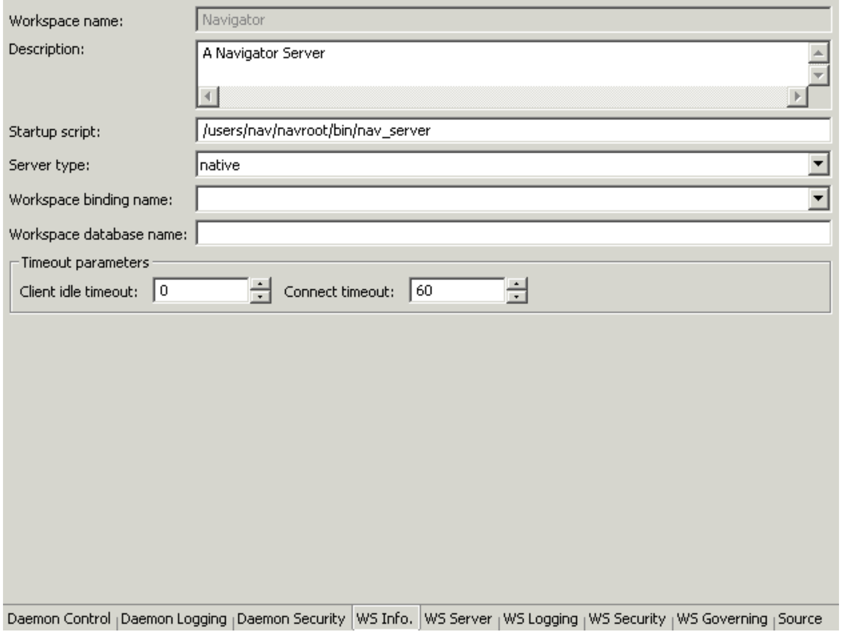
Using **WS Info.** you specify the features that control the operation of the workspace: the server type, the command procedure used to start the workspace and the binding configuration associated with this workspace.

The **WS Info.** section is accessed as follows:

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Right-click the machine and choose **Open Runtime Perspective**.
4. Drill-down the daemon to display the workspaces in the **Runtime Explorer**.
5. Right-click the workspace and choose **Edit Workspace Configuration**. The **WS Info.** tab opens.
6. After making changes to the daemon, right-click the daemon and choose **Reload Configuration**.

Note: You can also change workspace settings in the using the **Configuration Explorer**, by selecting a machine and drilling-down the list until the daemon is displayed. Drill-down the daemon to list the workspaces. Right-click the workspace and choose **Edit Workspace**.

Changes made to the daemon configuration in the way are not implemented. They are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager.



Workspace name: Navigator
Description: A Navigator Server
Startup script: /users/nav/navroot/bin/nav_server
Server type: native
Workspace binding name:
Workspace database name:
Timeout parameters
Client idle timeout: 0 Connect timeout: 60

The following fields are displayed:

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 pathname 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 for Tuxedo includes a default startup script, which it is recommended to use.

Server type - For internal use only.

Workspace binding name - For internal use only.

Timeout parameters - The time the client waits for the workspace server to start. If the workspace server does not start within this period, the client is notified that the server did not respond. Specifying the timeout here overrides the default setting, specified in the Control section.

> **See Also:** "Daemon Control" on page C-2 for details about the 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, 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.

WS Server

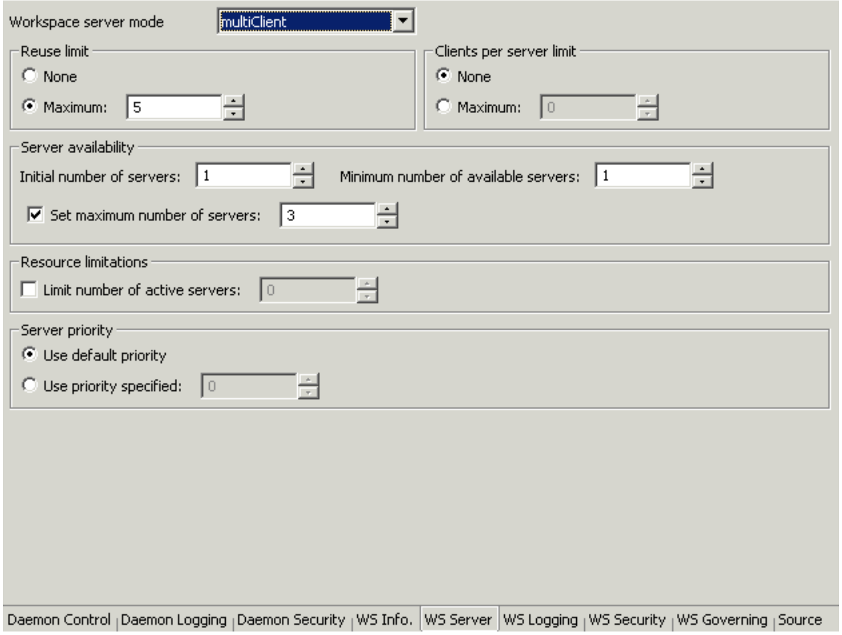
Using **WS Server**, you specify the features that control the operation of the servers started up by the workspace and allocated to clients. For example, you can configure the workspace to start up a number of servers for future use, prior to any client request, instead of starting each server when a request is received from a client.

The **WS Server** section is accessed as follows:

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Right-click the machine and choose **Open Runtime Perspective**.
4. Drill-down the daemon to display the workspaces in the **Runtime Explorer**.
5. Right-click the workspace and choose **Edit Workspace Configuration**.
6. Choose the **WS Server** tab.
7. After making changes to the daemon, right-click the daemon and choose **Reload Configuration**.

Note: You can also change workspace settings in the using the **Configuration Explorer**, by selecting a machine and drilling-down the list until the daemon is displayed. Drill-down the daemon to list the workspaces. Right-click the workspace and choose **Edit Workspace**.

Changes made to the daemon configuration in the way are not implemented. They are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager.



Workspace server mode: multiClient

Reuse limit
○ None
● Maximum: 5

Clients per server limit
● None
○ Maximum: 0

Server availability
Initial number of servers: 1
Minimum number of available servers: 1
☑ Set maximum number of servers: 3

Resource limitations
☐ Limit number of active servers: 0

Server priority
● Use default priority
○ Use priority specified: 0

The following fields are displayed:

Workspace server mode - The type of new server processes that the daemon starts up. The daemon supports the following server modes:

> **Single client** - 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 (since 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 (since it requires as many server processes as concurrent clients).

Multi-client - Clients share a server process and are processed serially. This mode has low overhead since the server processes are already 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.

Multi-threaded - For internal use only.

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 since the servers are already initialized. However, this server mode can use a lot of server resources (since it requires as many server processes as concurrent clients).

Note: The other modes can be set so that the server processes are reusable. The number of times a process can be reused is controlled by the `Reuse limit` field value.

Reuse limit - 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 enabled only if the server mode value is `Multi-client` or `reusable`.

Clients per server limit - 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 per server is unlimited. This field is enabled only if the server mode value is `Multi-client`.

Server availability

Initial number of servers - The number of server processes that are prestarted for this workspace when the daemon starts up. When the number of available server processes drops below the value specified in the `Minimum number` field (see below), 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.

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 a number of 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 above this value are terminated. The default for this field is 0, meaning that there is no maximum.

Resource limitations

Number of subtasks - Not applicable.

Note: The `Number of subtasks` field is displayed if an Oracle Connect legacy adapter, running on an IBM OS/390 machine is installed at the site, otherwise you do not see this field.

Limit number of active servers - The maximum number of active server processes (either available or in use). Once reached, no new server processes will be created for the particular workspace and client connections would be rejected if there is no available server to accept them. Once the number of active servers drops below the maximum (for example, a client disconnects from a server and the server terminates), new servers can again be started. If the value of this field is set to a nonzero value lower than the value for the **Initial number of servers** field, the daemon assumes it is set to the same value as the **Initial number of servers** field. The default for this field is 0, meaning that no maximum is enforced.

Server Priority - 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.

Use default priority - Sets the priority as 0. There is no specific priority for this workspace.

Use priority - Enables setting the priority.

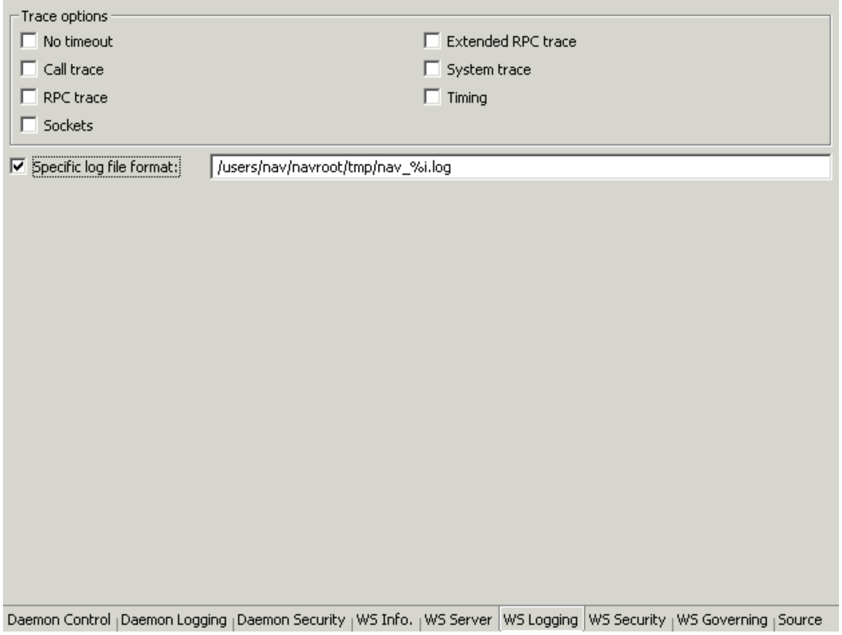
WS Logging

Using **WS Logging** you specify parameters to log that occur with the workspace server process.

The **WS Logging** section is accessed as follows:

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Right-click the machine and choose **Open Runtime Perspective**.
4. Drill-down the daemon to display the workspaces in the **Runtime Explorer**.
5. Right-click the workspace and choose **Edit Workspace Configuration**.
6. Choose the **WS Logging** tab.
7. After making changes to the daemon, right-click the daemon and choose **Reload Configuration**.

> **Note:** You can also change workspace settings in the using the **Configuration Explorer**, by selecting a machine and drilling-down the list until the daemon is displayed. Drill-down the daemon to list the workspaces. Right-click the workspace and choose **Edit Workspace**.
>
> Changes made to the daemon configuration in the way are not implemented. They are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager.



Trace options

□ No timeout

□ Call trace

□ RPC trace

□ Sockets

□ Extended RPC trace

□ System trace

□ Timing

☑ Specific log file format: /users/nav/navroot/tmp/nav_%i.log

The following fields are displayed:

Trace options - Specifies what tracing is performed.

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.

Sockets - Generates 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 - 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 operating system-specific tracing.

Timing - Generates a timestamp for every entry to the server log file.

Specific log file format - Defines the name and location of the server log file. The field must specify the full pathname. If no directory information is provided for the log file, it will be located in the login directory of the account running the Oracle Connect for Tuxedo workstation.

The following tokens can appear in the log file template and will be 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)

WS Security

Using **WS Security** you specify the level of security at the workspace level, as opposed to the daemon level, which is set in the Security section of the daemon.

> **See Also:** "Daemon Security" on page C-7 for details about security

The **WS Security** section is used for the following:

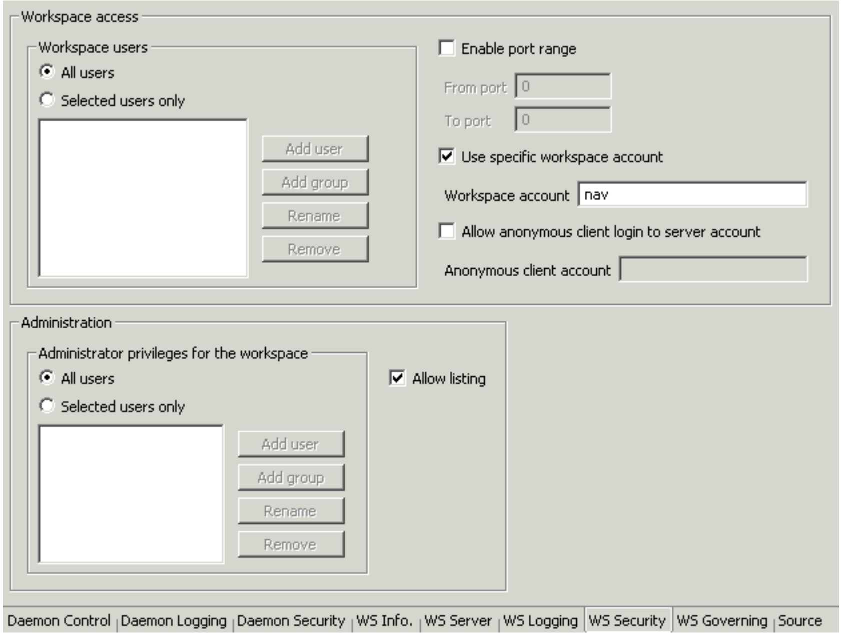
- To grant administration rights for the workspace
- To determine access to the workspace by a client

The **WS Security** section is accessed as follows:

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Right-click the machine and choose **Open Runtime Perspective**.
4. Drill-down the daemon to display the workspaces in the **Runtime Explorer**.
5. Right-click the workspace and choose **Edit Workspace Configuration**.
6. Choose the **WS Security** tab.

7. After making changes to the daemon, right-click the daemon and choose **Reload Configuration**.

> **Note:** You can also change workspace settings in the using the **Configuration Explorer**, by selecting a machine and drilling-down the list until the daemon is displayed. Drill-down the daemon to list the workspaces. Right-click the workspace and choose **Edit Workspace**.
>
> Changes made to the daemon configuration in the way are not implemented. They are only implemented after the configuration is reloaded using the **Reload Configuration** option in the Runtime Manager.



Workspace access

Workspace users

● All users

○ Selected users only

□ Enable port range

From port 0

To port 0

☑ Use specific workspace account

Workspace account nav

□ Allow anonymous client login to server account

Anonymous client account

Administration

Administrator privileges for the workspace

● All users

○ Selected users only

☑ Allow listing

The following fields are displayed:

Administration - Identifies the users (accounts) allowed to perform administrative tasks (tasks that require administrative login) on this workspace. For example, a user with administrative rights to a workspace can refresh the specific workspace servers using the `IRPCD` command with the `Refresh` *`workspace`* option.

Administrator privileges - Identifies the users (accounts) allowed to perform administrative tasks to the workspace (tasks that require administrative login).

All users - Anyone can access the workspace and change the settings.

Selected users only - The names of users (accounts) and groups that can be administrators.[2]

Note: If a user is not specified, the users specified in the `Workspace users` field (see below) have administrator rights for this workspace. In this case, if all users are allowed for the `Workspace users` field, all users have administrator rights for this workspace.

Allow Listing - Determines whether this workspace appears in the list of workspaces.

Workspace access - Defines the users (accounts) allowed to access this workspace.

Workspace users - Lists the users who are allowed to use the workspace (after logging on). If All users is specified, any user who has logged on to the daemon can use the workspace.

All users - Any user who has logged on to the daemon can use the workspace.

Selected users only - The names of users (accounts) and groups that can use the workspace.[3]

Note: If a user is not specified, any user who has logged on to the daemon can use the workspace.

Enable ports range - The firewall ports through which you access the workspace. Specifies 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 for Tuxedo can be accessed through a firewall.

[2] The name is prefixed with ’@’, to utilize the operating system GROUP feature.

[3] The name is prefixed with ’@’, to utilize the operating system GROUP feature.

Use specific workspace account - The operating system account used for the workspace. If not specified, the account name that was provided by the client is used.

Allow anonymous client login to server account - Whether this workspace can be invoked without authentication (user name/password). If anonymous login is allowed, specify the server account name to use. If this field is not specified, the value in the `Workspace account` field is used.

D

National Language Settings

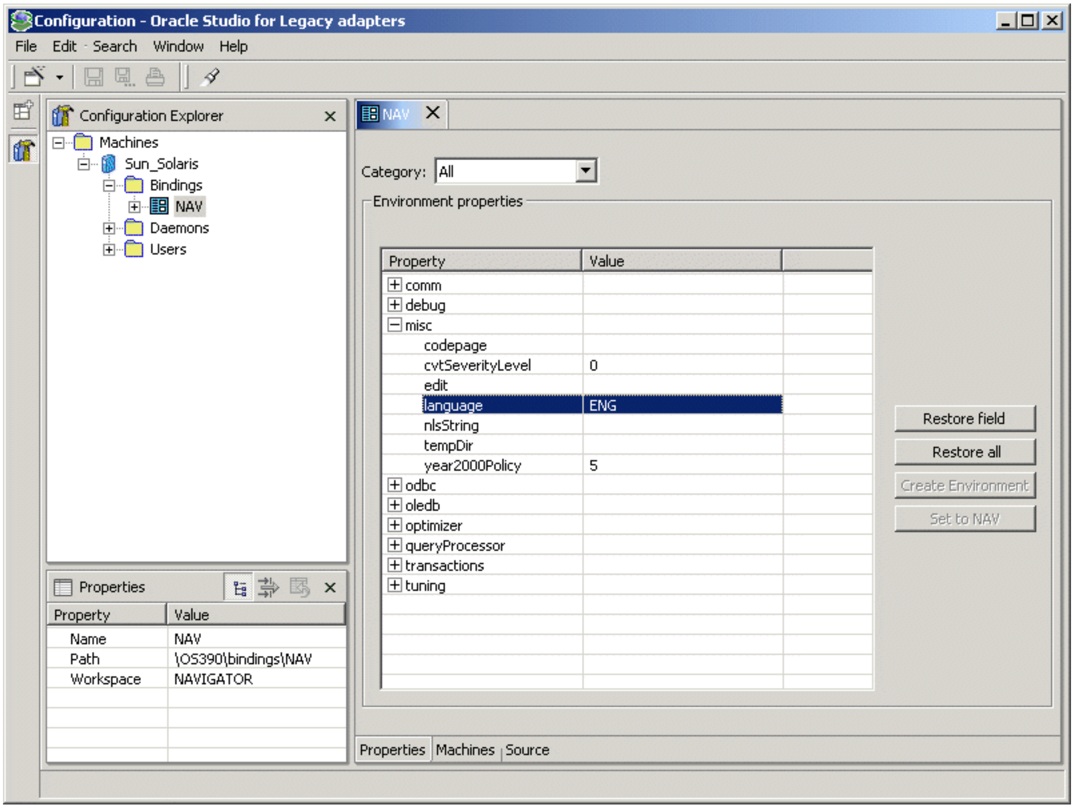
National Language Support (NLS) is provided by the Oracle Application Server Integration Adapter for Tuxedo for the following languages:

- Arabic
- English (the default)
- Hebrew
- Japanese
- Korean
- Simple Chinese
- Spanish
- Traditional Chinese

This appendix describes how to set up language support.

Defining the Language and Codepage

1. On the machine where Oracle Studio is installed, from the **Start** menu, choose **Start** > **Programs** > **Oracle > Studio > Studio**.
2. Select the machine defined in "Configuring Oracle Connect for Tuxedo" on page 2-11.
3. Right-click the **NAV** binding node and Choose **Edit Binding**.
4. In the **Binding Properties** tab, click the + next to the **misc** node.



Category: All

Property	Value
comm	
debug	
misc	
codepage	
cvtSeverityLevel	0
edit	
language	ENG
nlsString	
tempDir	
year2000Policy	5
odbc	
oledb	
optimizer	
queryProcessor	
transactions	
tuning	

Properties

Property	Value
Name	NAV
Path	\OS390\bindings\NAV
Workspace	NAVIGATOR

5. Enter a valid value for the language. The following language codes are valid:

   ARA - Arabic

   ENG - English

   HEB - Hebrew

JPN - Japanese

KOR - Korean

SCHI - Simple Chinese

SPA - Spanish

TCHI - Traditional Chinese

6. Optionally, in the codepage field, specify the codepage required.

   You can skip this step, and just specify a language (see the previous step). In this case, a default codepage is used. The following shows the default codepages:

   ARA - AR8ISO8859P6

   ENG - ASCII

   HEB - IW8ISO8859P8

   JPN - JA16SJIS on all platforms except for Solaris Operating System (SPARC), where the default is JA16EUC

   KOR - KO16KSC5601

   SCHI - ZHS16CGB231280

   SPA - ASCII

   TCHI - ZHT16BIG5

Index