# Oracle® Applications InterConnect

User Guide

Release 4.1

May 2001

Part No. A90225-02

ORACLE

Oracle Applications Interconnect User Guide, Release 4.1

Part No. A90225-02

# Contents

**Send Us Your Comments** ..... vii

**Preface** ..... ix

## 1 Introduction to Oracle Applications InterConnect

**Oracle Applications InterConnect Overview** ..... 1-2
**Oracle Applications InterConnect Features** ..... 1-2
Separation of Integration Logic and Platform Services ..... 1-3
Unique Integration Methodology ..... 1-4
Business Process Collaboration ..... 1-7
Use iStudio for Integration Specification ..... 1-8
Integration Lifecycle Management ..... 1-8
A Robust Oracle Infrastructure ..... 1-9
Prepackaged Adapters ..... 1-9
Extensibility Using SDKs ..... 1-11
Standard Messaging Middleware Services ..... 1-11
Value Added Features ..... 1-12
Flexible Deployment ..... 1-13
**Oracle Applications InterConnect Core Components** ..... 1-13
Dependent Products ..... 1-15
**Oracle Applications InterConnect Architecture** ..... 1-16

## 2 Design Time Concepts and iStudio

**Integration Process Overview** ..... 2-2

Design Time ... 2-2
Runtime ... 2-2
**Unique Integration Methodology** ... 2-4
**iStudio Concepts** ... 2-8
Workspaces ... 2-8
Projects ... 2-9
Applications ... 2-9
Common View ... 2-9
Integration Points ... 2-9
Business Objects ... 2-10
Events ... 2-10
Procedures ... 2-11
Common Data Types ... 2-11
Application View ... 2-12
Transformations or Mappings ... 2-12
Application Data Types ... 2-12
Metadata Versioning ... 2-13
Event Maps ... 2-14
Content-Based Routing ... 2-15
Cross Reference Tables ... 2-15
Domain Value Mapping ... 2-15
Routing and the Message Capability Matrix ... 2-15
Tracking Fields ... 2-16
**Using iStudio** ... 2-16
Using the iStudio Toolbar ... 2-16
Workspaces and Projects ... 2-17
Common View ... 2-22
Application View ... 2-31
Enabling Infrastructure ... 2-61

## 3 Oracle Applications InterConnect and Oracle Workflow

**Oracle Workflow Overview** ... 3-2
Oracle Workflow Solves Common Business Problems ... 3-2
**Oracle Applications InterConnect Integration with Oracle Workflow** ... 3-4
Design Time Tools ... 3-4

Runtime ..... 3-6
**Using Oracle Workflow with Oracle Applications InterConnect** ..... 3-8
Install Oracle Workflow Components ..... 3-8
Design Business Process ..... 3-8
Deploy Business Processes for Runtime ..... 3-8
**Design Business Process** ..... 3-9
Process Bundle ..... 3-9
Business Process ..... 3-9
Activity ..... 3-9
Creating a Process Bundle ..... 3-10
Creating a Business Process ..... 3-11
Populating a Business Process with Activities ..... 3-13
Deploying to Oracle Workflow ..... 3-15
Launching Oracle Workflow Tools ..... 3-19
Modifying Existing Oracle Workflow Processes ..... 3-21

## 4 Runtime System Concepts and Compents

**Integration Architecture** ..... 4-2
**Features** ..... 4-3
Messaging Paradigms ..... 4-3
Message Delivery ..... 4-3
Message Retention ..... 4-4
Routing Support ..... 4-4
Error Management ..... 4-4
Scalability and Load Balancing ..... 4-5
**Components** ..... 4-7
Adapters ..... 4-7
Repository ..... 4-9
Advanced Queues ..... 4-10
Workflow ..... 4-10

## 5 Runtime Management

**Introduction to Runtime Management** ..... 5-2
Starting the Enterprise Manager Console ..... 5-2
**Features** ..... 5-4

Common Features for Adapters and the Repository ..... 5-4
Repository Specific Features ..... 5-5
Adapter Specific Features ..... 5-5

## A Transformations

Copy Fields ..... A-2
Copy Object ..... A-2
Concat Fields ..... A-2
Expand Fields ..... A-2
Set Constant ..... A-3
True Conditional Lookup XRef ..... A-3
True Conditional Lookup DVM ..... A-3
Conditional Copy ..... A-4
True Conditional Copy ..... A-4
True Conditional Concat ..... A-4
True Conditional To Number ..... A-5
False Conditional Copy ..... A-5
False Conditional Concat ..... A-5
False Conditional To Number ..... A-6
To Number ..... A-6
` ..... A-6
Char Replace ..... A-7
String Replace ..... A-7
LTrim ..... A-7
RTrim ..... A-8
LPad ..... A-8
RPad ..... A-8
Lookup XRef ..... A-9
Delete XRef ..... A-9
Lookup DVM ..... A-9
Truncate ..... A-10
Increment ..... A-10

## Index

# Send Us Your Comments

**Oracle Applications InterConnect User Guide, Release 4.1**

**Part No. A90225-02**

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. 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 about this manual?

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

- Electronic mail - iasdocs_us@oracle.com
- Fax - (650) 506-7409 Attn: Oracle Applications InterConnect Documentation Manager
- Postal service:

  Oracle Corporation
  Oracle Applications InterConnect Documentation Manager
  500 Oracle Parkway, M/S 2op4
  Redwood Shores, CA 94065 USA

If you would like a reply, please give your name, address, and telephone number below.

---

---

---

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

# Preface

Oracle Applications InterConnect is the integration component of Oracle 9iAS and provides a comprehensive application integration framework to enable seamless integration of enterprise software. Oracle Applications InterConnect is built on top of Oracle's robust integration platform and leverages its underlying services. Oracle Applications InterConnect is designed to integrate heterogeneous systems, such as Oracle Applications, non-Oracle applications, or 3rd party messaging oriented middleware. This integration can be deployed either within an enterprise or across enterprise boundaries through the Internet.

This preface contains these topics:


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

## Audience

This guide is targeted at the following types of users:

- Business analysts and integration engineers, for iStudio.
- System Administrators, for the runtime component.

The audience should have the following pre-requisites, which are discussed but not explained:

- Domain knowledge of the applications being integrated.
- Database concepts and working knowledge of SQL, PL/SQL, or SQL* Plus.

## Organization

This document contains:

**Chapter 1, "Introduction to Oracle Applications InterConnect"**
Introduces Oracle Applications InterConnect and presents an overview of the product and the tools.

**Chapter 2, "Design Time Concepts and iStudio"**
Describes the design-time components and concepts of Oracle Applications InterConnect. It also explains iStudio, the point and click wizard based tool that allows you to specify your integration logic.

**Chapter 3, "Oracle Applications InterConnect and Oracle Workflow"**
Describes how Oracle Applications InterConnect works with Oracle Workflow.

**Chapter 4, "Runtime System Concepts and Compents"**
Describes the runtime components and concepts of Oracle Applications InterConnect.

**Chapter 5, "Runtime Management"**
Introduces the Runtime Management Console and describes how you use it to manage your integration environment.

**Appendix A, "Transformations"**
Provides a list of Oracle Applications InterConnect transformations.

## Related Documentation

For more information, see these Oracle resources:

- Oracle9*i* Application Server Documentation Library CD-ROM
- Oracle9*i* Application Server Platform Specific Documentation on Oracle9*i* Application Server Disk 1
- *Oracle Applications InterConnect Installation Guide*
- *Oracle Applications InterConnect Release Notes*

In North America, printed documentation is available for sale in the Oracle Store at

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

Customers in Europe, the Middle East, and Africa (EMEA) can purchase documentation from

`http://www.oraclebookshop.com/`

Other customers can contact their Oracle representative to purchase printed documentation.

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://technet.oracle.com/membership/index.htm`

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://technet.oracle.com/docs/index.htm`

# 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 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 Concepts*<br>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.<br>You can back up the database by using the `BACKUP` command.<br>Query the `TABLE_NAME` column in the `USER_TABLES` data dictionary view.<br>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.<br><br>**Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown. | Enter `sqlplus` to open SQL*Plus.<br><br>The password is specified in the `orapwd` file.<br><br>Back up the datafiles and control files in the `/disk1/oracle/dbs` directory.<br><br>The `department_id`, `department_name`, and `location_id` columns are in the `hr.departments` table.<br><br>Set the `QUERY_REWRITE_ENABLED` initialization parameter to `true`.<br><br>Connect as `oe` user.<br><br>The `JRepUtil` class implements these methods. |
| *`lowercase monospace (fixed-width font) italic`* | Lowercase monospace italic font represents placeholders or variables. | You can specify the *`parallel_clause`*.<br><br>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}`<br>`[COMPRESS \| NOCOMPRESS]` |

| Convention | Meaning | Example |
|---|---|---|
| ... | Horizontal ellipsis points indicate either:<br>■ That we have omitted parts of the code that are not directly related to the example<br>■ That you can repeat a portion of the code | `CREATE TABLE ... AS subquery;`<br>`SELECT col1, col2, ... , coln FROM employees;` |
| .<br>.<br>. | Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example. | |
| Other notation | You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown. | `acctbal NUMBER(11,2);`<br>`acct CONSTANT NUMBER(4) := 3;` |
| *Italics* | Italicized text indicates placeholders or variables for which you must supply particular values. | `CONNECT SYSTEM/system_password`<br>`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;`<br>`SELECT * FROM USER_TABLES;`<br>`DROP TABLE hr.employees;` |
| lowercase | Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files.<br>**Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown. | `SELECT last_name, employee_id FROM employees;`<br>`sqlplus hr/hr`<br>`CREATE USER mjones IDENTIFIED BY ty3MU9;` |

## Documentation Accessibility

Oracle's goal is to make our products, services, and supporting documentation accessible to the disabled community with good usability. 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 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/`

# 1

# Introduction to Oracle Applications InterConnect

This chapter provides an overview of Oracle Applications InterConnect, its features, and components.

Topics include:

- Oracle Applications InterConnect Overview
- Oracle Applications InterConnect Features
- Oracle Applications InterConnect Core Components
- Oracle Applications InterConnect Architecture

# Oracle Applications InterConnect Overview

Oracle Applications InterConnect is the integration component of Oracle 9iAS and provides a comprehensive application integration framework to enable seamless integration of enterprise software. It is built on top of Oracle's robust integration platform and leverages its underlying services. It is designed to integrate heterogeneous systems, such as Oracle Applications, non-Oracle applications, or 3rd party messaging oriented middleware. This integration can be deployed either within an enterprise or across enterprise boundaries through the Internet.

The following are technical design goals for Oracle Applications InterConnect:

- Elevate the integration problem from a technical coding exercise to a functional modeling exercise thereby reducing, or even eliminating, the programming effort normally associated with integration.
- Develop and expose an integration methodology that promotes reuse and reduces the complexity and management issues that arise over the software lifecycle.

# Oracle Applications InterConnect Features

Oracle Applications InterConnect features include:

- Separation of Integration Logic and Platform Services
- Unique Integration Methodology
- Business Process Collaboration
- Use iStudio for Integration Specification
- Integration Lifecycle Management
- A Robust Oracle Infrastructure
- Prepackaged Adapters
- Extensibility Using SDKs
- Standard Messaging Middleware Services
- Value Added Features
- Flexible Deployment

## Separation of Integration Logic and Platform Services

Oracle Applications InterConnect cleanly breaks up the integration problem into two discrete components:

- High-level integration logic.
- Low-level platform services.

### Integration Logic

Integration logic consists of the business rules and transformation logic necessary to integrate heterogeneous systems. Using iStudio, this integration logic can be modeled. The results are then stored in the Oracle Applications InterConnect repository as metadata.

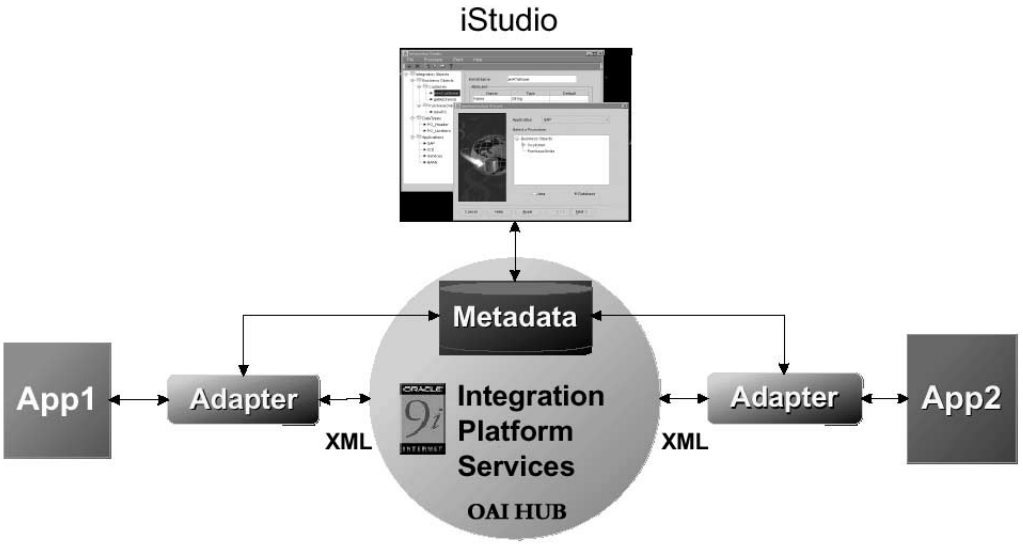
### Platform Services

Platform Services consist of the integration infrastructure provided with Oracle 9iAS and the Oracle database. In addition, Oracle Applications InterConnect provides application and protocol adapters. The platform services provide the requisite infrastructure necessary for integration.

### Integration Logic Drives Platform Services

Integration using Oracle Applications InterConnect is a two-step process. During design time, integration logic is modeled in iStudio and captured in the repository as metadata. At runtime, the underlying services treat this metadata as runtime instructions to enable the conversation among participating applications.

***Figure 1–1 Oracle Applications InterConnect Metadata and Data Flow***




## Unique Integration Methodology

iStudio exposes an integration methodology that eliminates the complexities of point-to-point custom integration solutions. The integration methodology is based on a hub-and-spoke model.

### How the Hub-and-Spoke Methodology Works

An integration point is defined as an "event" that triggers communication between two or more participating applications in the integration scenario. The following are examples of such "events:"

- Create Customer—The integration scenario above might require that customer information across the two applications be synchronized in realtime. Whenever a new customer is created in App1 above, the customer should be created in App2 also. Therefore, "Create Customer" is an "event" that triggers the communication between the two applications—App1 produces the information, App2 consumes it.

- Get Item Info—A user of App1 might request information on some item stored in App1. The information on that item might be segmented across the two applications. To give a meaningful response to the user of App1, it is necessary to query App2 for information on the item. Therefore, "Get Item Info" is an integration point between the two applications because it triggers communication between the two applications—App1 produces a query, App2 consumes it, App2 produces the response, and App1 consumes it.

The common view consists of a list of such integration points, each with its own associated data. Applications participate in the integration by binding to one or more of these common view integration points.

In the context of each binding, applications have their own application view of data that needs to be exchanged. Each binding involves mapping, or transformation, between the application view and the common view in the context of the integration point. In this model, the application views are at the spokes and the common view is the hub.

The Create Customer example helps to explain Figure 1–2, "Oracle Applications InterConnect Hub-and-Spoke Model". Create Customer is an integration point. If the information to exchange is the new customer's name only, then the common view has all the information potentially captured in a name defined in an application-independent way. In the very least, this information must be a superset of all the information that needs to be exchanged across App1 and App2.

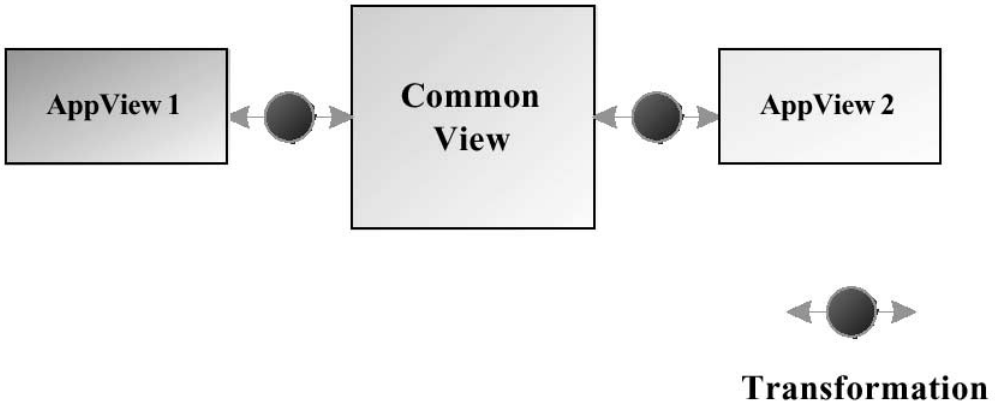
`Prefix, First Name, Last Name, Middle Initial, Maiden Name, Suffix` is an example of a common view customer name definition.

Now, App1's internal "definition" of name (App1's application view) could be `First Name, Last Name, Middle Initial, Prefix`.

App2's application view could be `Name` (one field that contains Last Name, First Name).

For App1, in the context of sending this information out or publishing an event, transformations are defined from its application view to the common view. For App2, in the context of receiving this information or subscribing to an event, transformations are defined from the common view to its application view.

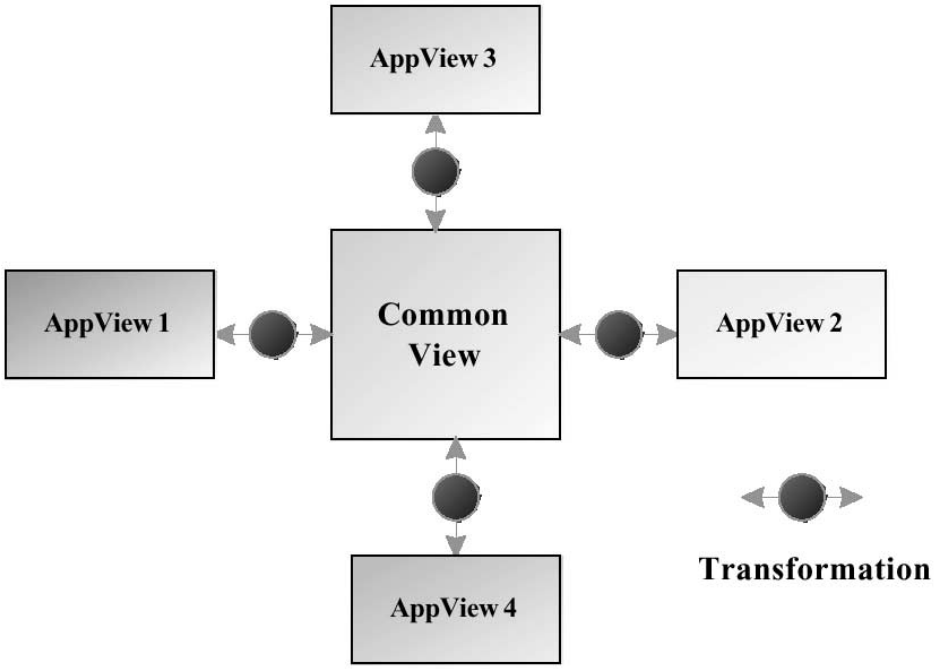
***Figure 1–2 Oracle Applications InterConnect Hub-and-Spoke Model***




This hub-and-spoke model has the following advantages:

- Loosely coupled integration—Applications integrate to the common view, not with each other directly. This reduces the number of integration interfaces.
- Easy Customization—Changes in application views due to application upgrades are localized. The changes in the upgraded application should only be reflected through changes in its application view and mappings to the common view. In other words, only the spoke of the changed application needs to be re-mapped to the hub. The other spokes and their relationships with the hub remain unchanged.
- Easy Extensibility—Applications can be added or removed from the integration scenario without affecting other applications. For example, if a new application is added to the integration scenario, it must define its spoke component (the application view) and map that to the hub (common view) on a per integration point basis. This exercise does not affect other applications in the integration.

  In the diagram below, App3 and App4 have been added to the integration scenario by plugging them into the common view. This does not affect the integration of App1 or App2.




- Enhanced Reusability—This is best explained through an example. To integrate the Marketing CRM module to SAP, the integration would be from iMarketing to common view to SAP. If there is a requirement to integrate iMarketing to Peoplesoft, then the iMarketing to common view integration can be reused. Only the common view to Peoplesoft integration needs to be built.

## Business Process Collaboration

Oracle Workflow provides a comprehensive business process management system that enables traditional workflow applications, as well as process collaboration in a single solution. Oracle Workflow's new Business Event System enables Oracle Applications InterConnect and Oracle Workflow to work together to provide a complete business process driven integration solution. With Oracle Applications InterConnect and Oracle Workflow, you can define business collaborations across two or more applications to implement the organization's business processes.

## Use iStudio for Integration Specification

iStudio is a design time integration specification tool targeted at business analysts. This tool helps business analysts specify the integration logic at a functional level, not at a technical coding level. iStudio exposes the integration methodology using simple wizards and drastically reduces, even eliminates, the need for writing code to specify the integration logic. This reduces the total time required to complete an integration.

Considering iStudio is a multi-user tool with fine-grained locking for all Oracle Applications InterConnect first class objects, multiple users can work simultaneously on the same integration scenario without compromising the consistency of the metadata.

## Integration Lifecycle Management

Managing, customizing, and evolving an integration over time is as important as creating the integration in the first place. The hub-and-spoke integration model has advantages to help achieve this goal. In addition, the Oracle Applications InterConnect repository, which contains all the integration logic, provides extensive services for managing changes over time. The repository provides fine-grained versioning of all Oracle Applications InterConnect first class objects such as events, messages, and data types. Some of the important aspects of versioning to help with the lifecycle support include the following:

- Basic Versioning—New versions of first class objects such as messages, can be created to address changing integration needs. Different versions of the same object can co-exist in the repository. This approach has two advantages:
  - Eliminates the need for an expanded namespace to address modifications.
  - Allows related entities to be grouped together for easier management.
- Multiple Active Versions—Multiple versions of the same message can be active in the same integration scenario simultaneously. This can help transition an integration incrementally without requiring changes to existing messages. For example, if a purchase order definition for an application, or the application view of the purchase order, needs to change, a new version of the message can be created and activated for that application. Once this metadata is created, the application can smoothly transition from sending / receiving messages based on the old definitions to the new one.

- Migration Support—Different versions of metadata can be migrated across repositories on a first class object basis. This feature allows fine-grained control of content in different repositories, such as a development repository and a production repository.
- Consistency Control—Oracle Applications InterConnect detects and flags metadata conflicts. This helps to prevent accidental overwriting of metadata and maintains consistency of metadata in the repository.

## A Robust Oracle Infrastructure

All Oracle Applications InterConnect components are written in pure Java and utilize proven Oracle infrastructure to deliver a robust, reliable, and scalable integration solution. Oracle 9iAS and its corresponding database constitutes the messaging hub for Oracle Applications InterConnect. In particular, Oracle Workflow in Oracle 9iAS and Advanced Queues in the database allows an Oracle Applications InterConnect user to build integrations on top of a solid platform. Using Oracle Workflow, complex business processes can be built to handle different integration needs. In addition, Advanced Queues provide a robust messaging infrastructure, message retention, auditing, and tracking support.

## Prepackaged Adapters

Prepackaged adapters help re-purpose applications at runtime to participate in the integration without any programming effort.

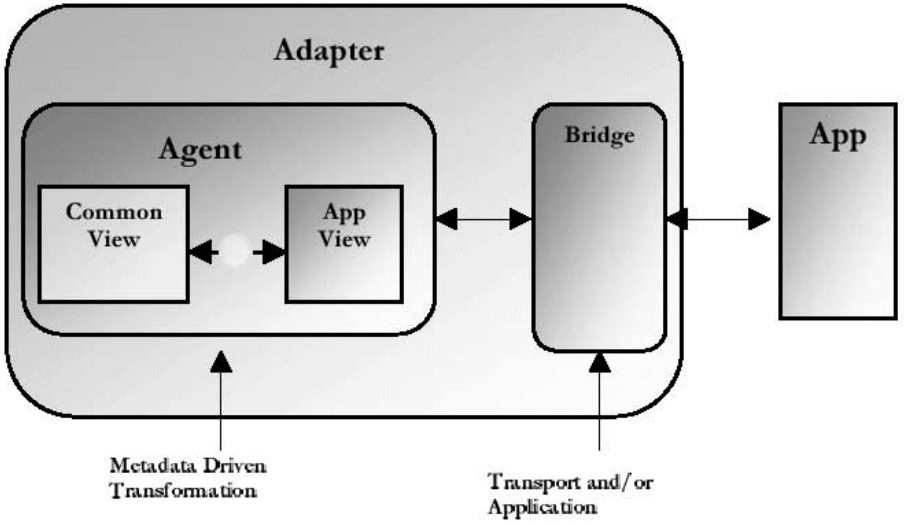
### Agent and Bridge Combination

Adapters are the run-time component for Oracle Applications InterConnect. Adpaters have the following responsibilities:

- Application Connectivity—Connect to applications to transfer data between the application and Oracle Applications InterConnect. The logical sub-component within an adapter handles this responsibility is called a bridge. This is the protocol / application-specific piece of the adapter that knows how to communicate with the application. For example, the database adapter is capable of connecting to an Oracle database using JDBC and calling SQL APIs. This sub-component does not know which APIs to call, only how to call them.
- Transformations—Transform data from the application view to common view and vice versa as dictated by the repository metadata. In general, adapters are responsible for carrying out all the runtime instructions captured through iStudio as metadata in the repository. Transformations are an important subset

of these instructions. The logical sub-component within an adapter that handles the runtime instructions is called an agent. This is the generic runtime engine in the adapter that is independent of the application to which the adapter connects. It focuses on the integration scenario based on the integration metadata in the repository. There is no integration logic coded into the adapter itself; all integration logic is stored in the repository. The repository contains the metadata that drives this sub-component. In the database adapter example, this is the sub-component that knows which SQL API's to call, but not how to call them. All adapters have the same agent code. It is the difference in metadata that each adapter receives from the repository that controls and differentiates the behavior of each adapter.

***Figure 1–3 Oracle Applications InterConnect Adapter Architecture***




These adapters can be technology or application adapters. Oracle Applications InterConnect currently packages technology adapters for the Oracle database, Advanced Queues, and HTTP/S technology. Access to non-Oracle databases is possible through Transparent Gateways provided by Oracle. Application adapters include Oracle and SAP R/3 adapters.

## Extensibility Using SDKs

Oracle Applications InterConnect is easily extended through SDKs which addresses custom integration needs. There are two SDKs to provide this flexibility:

- iStudio SDK—Extend iStudio using the iStudio SDK by writing Java code to do the following:
  - Add custom transformation functions.
  - Add custom browsers to import APIs or data structures from application-native repositories into iStudio. For example, the database browser provided with a standard edition of iStudio, allows a SQL API, table, view, or Abstract Data Type definitions to be imported from the database to define the application view of data for a database-based application.
- Adapter SDK—Write new adapters in Java using the Adapter SDK for applications or protocols not currently supported by Oracle Applications InterConnect. Specifically, only the bridge sub-component must be written. The agent is a generic engine already written and is part of each adapter.

## Standard Messaging Middleware Services

Oracle Applications InterConnect provides all the basic services expected of a messaging middleware platform including:

- Support for all major messaging paradigms—Publish/Subscribe, Request/Reply (synchronous and asynchronous), and Point-to-Point. For more information, see Chapter 2, "Design Time Concepts and iStudio".
- Guaranteed delivery of messages—All messages have guaranteed delivery end-to-end. Messages are delivered exactly once and in the order sent.
- Scalability—Multiple adapters are instantiated to serve one application. The hub runs in an Oracle Parallel Server environment.
- Load Balancing—Messages can be partitioned based on load between multiple adapters servicing one application. At one extreme, one or more adapters can serve all messages for one application. At the other extreme, one or more adapters are dedicated per integration point in which the application participates.

- Runtime Management—An Oracle Enterprise Manager-based runtime management console helps manage the integration scenario and components at runtime. This console allows users to start and stop components, monitor message flow, detect problems, and manage errors.
- Deployment Support—The messaging hub consists of Advanced Queues that are configured for runtime. Number of queues to create, naming these queues, and matching adapters with messages in a specific named queue can be configured using iStudio.

## Value Added Features

The following value-added services are exposed through iStudio and do not require any coding:

- Content Based Routing—Route messages by building business rules based on message content. For example, a procurement system routes fulfillment requests to different fulfillment centers based on an originating location.
- Cross Referencing—Correlate keys that uniquely identify the entities in one application with corresponding entities created in other applications. For example, a purchase order created in a procurement system has a native id X. It is then routed to a fulfillment system and the purchase order is created in the fulfillment system with native id Y. Therefore, X and Y must be cross referenced for Oracle Applications InterConnect to correlate communication about this same logical entity in two different systems without each system knowing the native id of the other system.
- Domain Value Mapping—Map code tables across systems. For example, a purchase order in a procurement system has a PO Status field with possible domain values `Booked` and `Shipped`. The corresponding field in a fulfillment system has the possible domain values `1` and `2`. Oracle Applications InterConnect allows the user to create the mappings `booked=1`, `shipped=2` so it can correlate these values at runtime without each system knowing the domain value set of the other system.

## Flexible Deployment

Oracle Applications InterConnect provides a complete framework for e-Business application integration across the Application-to-Application, Application Service Provider, and Business-to-Business domains. Oracle Applications InterConnect components can be deployed for all of these domains as described:

- Application-to-Application—Applications are distributed within a Local Area Network or across a Wide Area Network. Oracle Applications InterConnect is deployed within the organization to integrate these applications.
- Application Service Provider—Applications are distributed across firewall boundaries with some applications residing inside the Application Service Provider firewall and others inside the customer firewall. Oracle Applications InterConnect can be deployed inside either one of the two firewalls, or both, to integrate these applications across the firewalls.
- Business to Business—This is similar to the Application Service Provider model if you replace the Application Service Provider with another business.

# Oracle Applications InterConnect Core Components

Oracle Applications InterConnect has the following core components.

### iStudio

iStudio is a graphical wizard-based design time tool. iStudio allows business analysts to complete the following tasks:

- Define the data that needs to be exchanged across applications.
- Semantically map the data across applications.
- Define the business process collaboration across applications utilizing Oracle Workflow and associate the semantic maps with business processes if required.
- Configure and deploy the integration.

iStudio is deployed as a stand-alone Java application running outside the database. iStudio runs only on Windows NT and can be deployed anywhere with access to the hub machine.

### Repository

The repository consists of a middle tier repository server program talking to a backend database. The following functionality is provided:

- At design time, all integration logic defined in iStudio is stored in tables in the repository as metadata.
- At runtime, the repository provides access to this metadata for adapters to integrate applications.

The repository server is deployed as a stand-alone Java application running outside the database. The repository schema is a set of tables in the backend database. Both the server and the database are on the hub machine.

### Adapters

Adapters are runtime components which process integration logic captured in the repository as runtime instructions to enable the integration. Adapters have two major tasks:

- Provide connectivity between an application and the hub.
- Transform and route messages between the application and the hub.

Adapters are deployed as stand-alone Java applications running outside the database. Adapters are physically co-located with the applications they connect to either on the same machine as the application itself, or on a separate machine. Adapters are usually not deployed on the hub machine.

### Runtime Management Console

The Runtime Management Console is an Oracle Enterprise Manager-based client / server combination. The Runtime Management Console provides the following functionality:

- Manages runtime components and resources.
- Troubleshoots errors and track messages.

The Oracle Enterprise Manager console includes a server component usually installed on the hub machine. The client component can be installed on any machine where the integration environment is managed. Both client and server components are stand-alone Java applications running outside the database.

### iStudio SDK

The iStudio SDK is a collection of Java jar and Javadoc files usually deployed on the same machine as iStudio. Using the iStudio SDK and Java, users can build the following:

- New transformation functions.
- New browsers to import application-native data structures and APIs into iStudio.

Documentation and samples are provided with the iStudio SDK.

### Adapter SDK

Using the Adapter SDK and Java, users can build new adapters. The Adapter SDK is a collection of Java jar and Javadoc files that can be deployed on any machine. Documentation and samples are provided with the Adapter SDK.

## Dependent Products

Oracle Applications InterConnect uses the following products.

### Oracle Database

The Oracle database is utilized for storing the repository metadata in tables and storing messages in Advanced Queues. Oracle Applications InterConnect supports the following releases of the Oracle database:

- Oracle 8.1.6
- Oracle 8.1.7
- Oracle9*i*

### Oracle Workflow

Oracle Workflow 2.6 comes with 9iAS 1.0.2.2. This is required only if business process collaborations need to be defined. For basic messaging based integration, this component is not utilized. The Oracle Workflow Business Event System enables Oracle Applications InterConnect and Oracle Workflow to work together to provide a complete business process driven integration solution. With Oracle Applications InterConnect and Oracle Workflow, business collaborations can be defined across two or more applications to implement the business process for an organization.

# Oracle Applications InterConnect Architecture

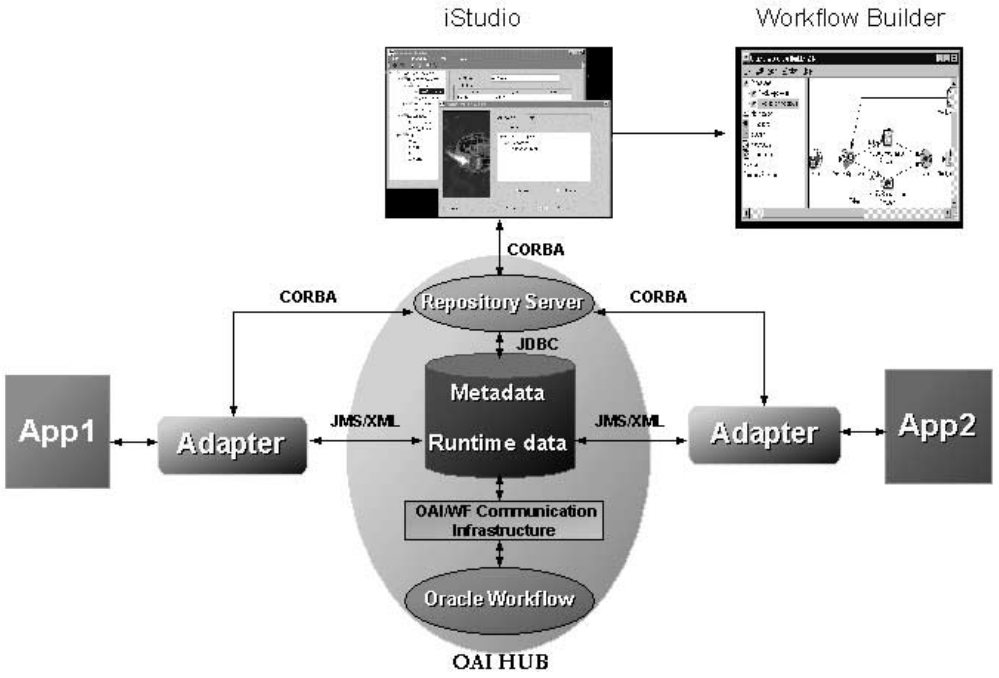



***Figure 1–4 An example of Oracle Applications InterConnect architecture.***

# 2

# Design Time Concepts and iStudio

This chapter describes the design time concepts and iStudio. In addition, this chapter describes how to perform important tasks using iStudio.

This chapter discusses the following topics:

- Integration Process Overview
- Unique Integration Methodology
- iStudio Concepts
- Using iStudio

# Integration Process Overview

Application integration using Oracle Applications InterConnect involves the following two phases:

- Design Time
- Runtime

## Design Time

During the design phase, a business analyst uses iStudio to define the integration objects, applications which participate in the integration, and the specifications of the data exchanged between applications. All the specifications are stored as metadata in the Oracle Applications InterConnect Repository.

## Runtime

For each application participating in a specific integration, Oracle Applications InterConnect attaches one or more adapters to it. At runtime, the adapters retrieve the metadata from the Repository to determine the format of messages, perform transformations between the various data formats, and route the messages to the appropriate queues in the Oracle Applications InterConnect hub.

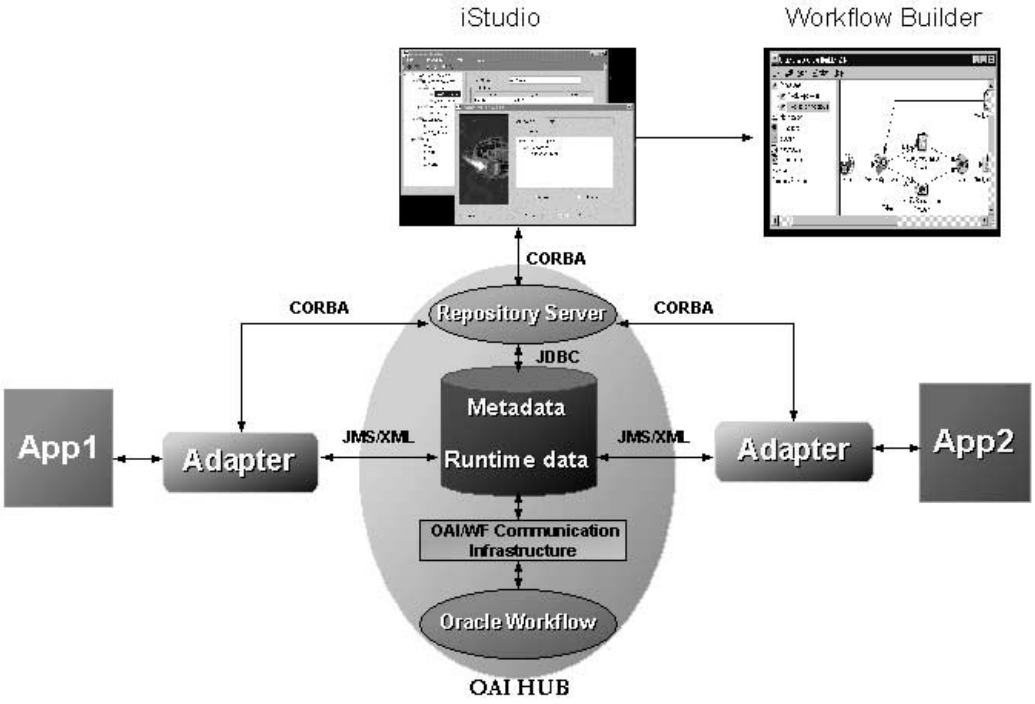



***Figure 2–1 A graphical overview of design time and runtime phases in integration***

# Unique Integration Methodology

iStudio exposes an integration methodology that eliminates the complexities of point-to-point custom integration solutions. The integration methodology is based on a hub-and-spoke model.

## How the Hub-and-Spoke Methodology Works

An integration point is defined as an "event" that triggers communication between two or more participating applications in the integration scenario. The following are examples of such "events:"

- Create Customer—The integration scenario above might require that customer information across the two applications be synchronized in real-time. Whenever a new customer is created in App1 above, the customer should be created in App2 also. Therefore, "Create Customer" is an "event" that triggers the communication between the two applications—App1 produces the information, App2 consumes it.
- Get Item Info—A user of App1 might request information on some item stored in App1. The information on that item might be segmented across the two applications. To give a meaningful response to the user of App1, it is necessary to query App2 for information on the item. Therefore, "Get Item Info" is an integration point between the two applications because it triggers communication between the two applications—App1 produces a query, App2 consumes it, App2 produces the response, and App1 consumes it.

The common view consists of a list of such integration points, each with its own associated data. Applications participate in the integration by binding to one or more of these common view integration points.

In the context of each binding, applications have their own application view of data that needs to be exchanged. Each binding involves mapping, or transformation, between the application view and the common view in the context of the integration point. In this model, the application views are at the spokes and the common view is the hub.

The Create Customer example helps to explain Figure 2–2, "Oracle Applications InterConnect Hub-and-Spoke Model". Create Customer is an integration point. If the information to exchange is the new customer's name only, then the common view has all the information potentially captured in a name defined in an application-independent way. In the very least, this information must be a superset of all the information that needs to be exchanged across App1 and App2.

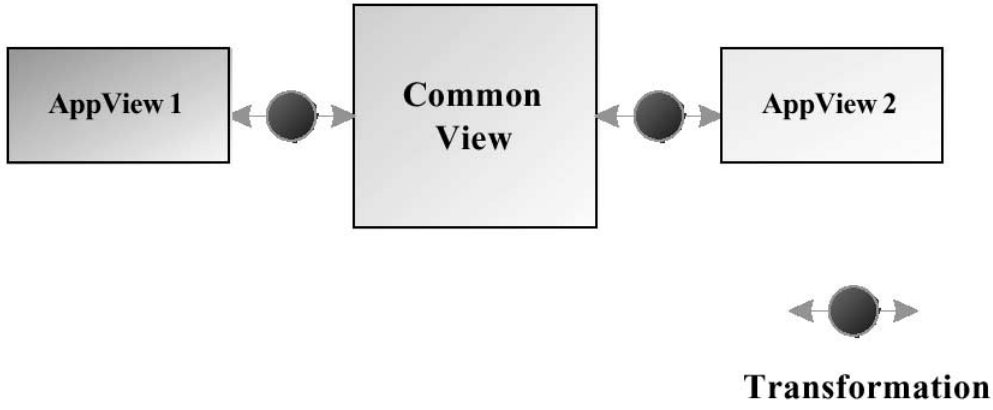
`Prefix, First Name, Last Name, Middle Initial, Maiden Name, Suffix` is an example of a common view customer name definition.

Now, `App1`'s internal "definition" of name (`App1`'s application view) could be `First Name, Last Name, Middle Initial, Prefix`.

`App2`'s application view could be `Name` (one field that contains Last Name, First Name).

For `App1`, in the context of sending this information out or publishing an event, transformations are defined from its application view to the common view. For `App2`, in the context of receiving this information or subscribing to an event, transformations are defined from the common view to its application view.

***Figure 2–2 Oracle Applications InterConnect Hub-and-Spoke Model***

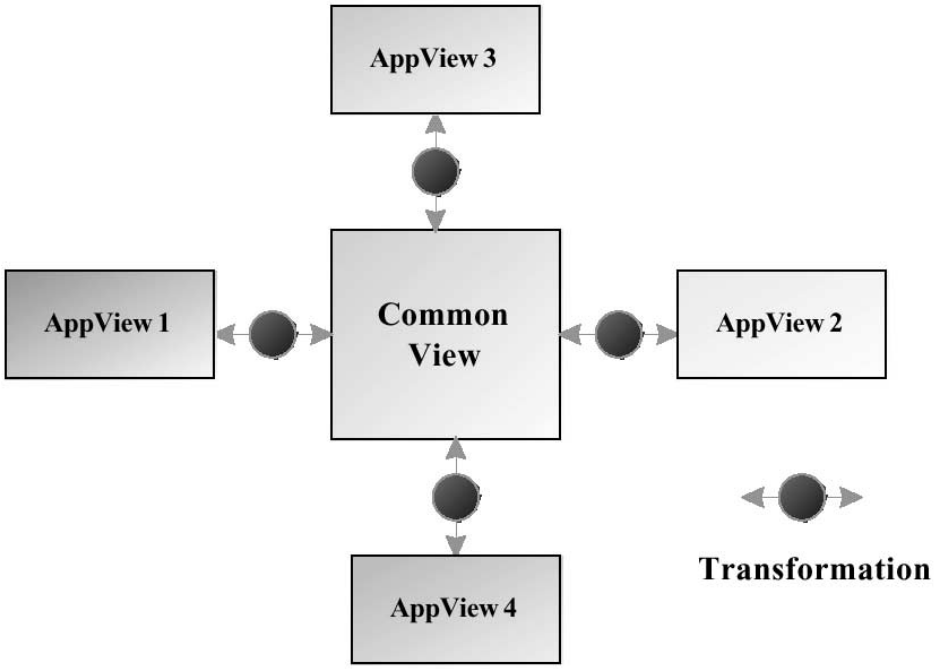



This hub-and-spoke model has the following advantages:

- Loosely coupled integration—Applications integrate to the common view, not with each other directly. This reduces the number of integration interfaces.
- Easy Customization—Changes in application views due to application upgrades are localized. The changes in the upgraded application should only be reflected through changes in its application view and mappings to the common view. In other words, only the spoke of the changed application needs to be re-mapped to the hub. The other spokes and their relationships with the hub remain unchanged.

- Easy Extensibility—Applications can be added or removed from the integration scenario without affecting other applications. For example, if a new application is added to the integration scenario, it must define its spoke component (the application view) and map that to the hub (common view) on a per integration point basis. This exercise does not affect other applications in the integration.

  In the diagram below, App3 and App4 have been added to the integration scenario by plugging them into the common view. This does not affect the integration of App1 or App2.




- Enhanced Reusability—This is best explained through an example. To integrate the Marketing CRM module to SAP, the integration would be from iMarketing to common view to SAP. If there is a requirement to integrate iMarketing to Peoplesoft, then the iMarketing to common view integration can be reused. Only the common view to Peoplesoft integration needs to be built.

Oracle Applications InterConnect supports the following messaging paradigms:

- Publish / Subscribe Messaging
- Request / Reply Messaging
- Point-to-Point Messaging

These paradigms are defined in iStudio at design time. The definitions are used at runtime to route the messages appropriately.

### Publish/Subscribe Messaging

An application publishes a message if it sends data out to the Oracle Applications InterConnect hub without knowing the destination applications. Furthermore, it does not expect any data in return. An application subscribes to a message if it receives the data from the Oracle Applications InterConnect hub regardless of who sent out the data. Furthermore, it does not send any data out in return. Events in iStudio are used to model this paradigm.

### Request/Reply Messaging

If an application publishes a message and expects a message in return as a reply, it is participating in the Request / Reply paradigm. The application subscribing to the request sends a reply back to the sender after consuming the request. Procedures in iStudio are used to model this paradigm. Request / Reply has the following two flavors:

- Synchronous—The application making the request blocks until it receives a reply.
- Asynchronous—The application makes the request and proceeds with normal processing. It does not wait for a response. A reply comes later and is consumed by the application.

### Point-to-Point Messaging

Both Publish / Subscribe and Request / Reply can take a point-to-point characteristic if the sending application explicitly calls out which application should receive the message. This can be modeled through content-based routing in iStudio.

# iStudio Concepts

iStudio is the design time modeling tool for Oracle Applications InterConnect. iStudio has several concepts explained in this section. The following concepts are discussed:

- Workspaces
- Projects
- Applications
- Common View
- Integration Points
- Business Objects
- Events
- Procedures
- Common Data Types
- Application View
- Transformations or Mappings
- Application Data Types
- Metadata Versioning
- Event Maps
- Content-Based Routing
- Cross Reference Tables
- Domain Value Mapping
- Routing and the Message Capability Matrix
- Tracking Fields

## Workspaces

A workspace stores user settings and preferences such as application login credentials and last opened project. Inside a workspace, users can work on multiple projects.

## Projects

A project in iStudio encapsulates all the integration logic for one integration scenario. An integration scenario is defined as a set of two ore more applications integrated with each other using Oracle Applications InterConnect. One project corresponds to one repository. For example, a user may have a development integration environment and a production integration environment. These are two separate projects and must, therefore be self-contained in their own separate repositories.

Since iStudio is a multi-user tool, multiple users can work on the same project simultaneously without jeopardizing the integrity of the metadata. To create a project in iStudio, a repository connection must be defined.

## Applications

Each component integrated with Oracle Applications InterConnect is referred to as an application. Each application expresses interest in specific messages, what its internal data type is, and how the message should be mapped to or from that internal type to the external world.

## Common View

As described earlier in this chapter, Oracle Applications InterConnect follows a hub-and-spoke integration methodology. The common view is the "hub view" of the integration where each spoke is the application that wishes to participate in the integration. The common view consists of the following elements:

- Business Objects
- Events
- Procedures
- Common Data Types

## Integration Points

An integration point is defined as any "event" triggering communication across two or more applications. Consider the following examples with two applications, `App1` and `App2`.

- Create Customer—A certain integration scenario may require customer information across the two applications be synchronized in real-time. Whenever a new customer is created in `App1`, the customer should also be created in

App2. Therefore, "Create Customer" is an "event" triggering the communication between the two applications—App1 produces the information, App2 consumes it.

- Get Item Info—A user of App1 requests information on some item stored in App1. The information on that item may be segmented across the two applications. To give a meaningful response to the user of App1 it is necessary to query App2 for information on the item. Therefore, "Get Item Info" is an integration point between the two applications because it triggers communication between the two applications—App1 produces a query, App2 consumes it, App2 produces the response, App1 consumes it.

## Business Objects

Business Objects are a collection of logically related integration points. For example, Create Customer, Update Customer, Delete Customer, Get Customer Info are all integration points that logically belong under a Customer business object. In other words, a business object is a way to organize integration points in iStudio.

## Events

An event is an integration point used to model the Publish/Subscribe paradigm. An event has associated data which is the common view of all the data to be exchanged through this event. In other words, the data associated with an event in the common view must be a superset of the data of participating applications.

For example App1 and App2 publishes customer names and App3 subscribes to it. If App1 publishes First Name, Last Name, and Middle Initial, and App2 publishes First Name, Last Name, Middle Initial, Prefix, and Suffix, the event could be defined as follows:

```
New Customer Event
Prefix
First Name
Last Name
Middle Initial
Suffix
```

**Note:** Standard application-independent definitions can be used for event-associated data in the common view such as Open Applications Group XML business object definitions.

## Procedures

A procedure is an integration point used to model the Request/Reply paradigm. This is a modeling paradigm only, no actual procedures are called. An application can either invoke a procedure to model sending a request and receiving a reply, or implement a procedure to model receiving a request and sending a reply. Similar to events, a procedure has associated data. While an event is only associated with one data set, a procedure has two data sets—one for the request or IN data and one for the reply or OUT data.

For example, if a Get Address procedure is defined so that the request contains the social security number for a person and the reply contains the address in four fields—Street, City, Zip, State, then the procedure is defined as follows:

```
get Address Procedure
SS# IN
Street OUT
City OUT
Zip OUT
State OUT
```

Procedures can be used to implement both synchronous and asynchronous request/reply.

---

**Note:** Standard application-independent definitions can be used for procedure-associated data in the common view such as Open Applications Group XML business object definitions.

---

## Common Data Types

When defining the data associated with an event or a procedure, it is possible to define the data once and reuse it for different integration points. Common data types are used to define such data for reuse and is especially useful for defining complex hierarchical data.

For example, a purchase order contains a header object and an array of line item objects. In addition, the Header object contains two address objects: Bill To and Ship To. Therefore, the purchase order can be defined once and used for other purchase order-related integration points such as `Create Purchase Order`, `Update Purchase Order`, and `Get Purchase Order`. Moreover, `Address` can be defined once and used in the `Bill To` and `Ship To` addresses.

## Application View

For participating in the integration, each application has its own application view of data. This application view of data plugs into the common view through transformations.

## Transformations or Mappings

Transformations are in the context of one integration point. For example, an event is created for transferring customer names across applications:

```
Common View Event New Customer
    Prefix
    First Name
    Last Name
    Middle Initial
    Suffix

Application View for App1 that publishes the event
    First Name
    Last Name
    Middle Initial

Application View for App2 that subscribes to the event
    Name – One field which contains First Name and Last Name separated by a
    comma.
```

When publishing or subscribing to the event, iStudio users must map the application view for App1 and App2 to the common view using transformations. There are twenty-seven built-in transformation routines provided with Oracle Applications InterConnect which are used to build complex mappings. In addition, the iStudio SDK allows users to create new transformation routines using Java, import them into iStudio to add to the built-in list, and use them just like a built-in routine.

## Application Data Types

Application data types have the same function as Common Data Types but are in the context of a particular application.

## Metadata Versioning

iStudio supports versioning for application and common data types, events, procedures, and messages.

An owner is the creator of the object and only the creator of an object can modify the object. However, other users can create new versions or copy the original object under a new name. The owner is specified at the time of Repository installation.

In the following example, the metadata is created at Oracle Corporation and at the time of Repository installation, Oracle Applications InterConnect is specified as the owner of the metadata. The following functionality is available for versioning:

- Automatic Versioning—First, an event called "NewCustomerEvent" is created. When this object is created for the first time, the assigned owner is `OAI` and the version is `V1`. This event name is `NewCustomerEvent/OAI/V1`.
- Modify Object—The owner is the only user who can modify the contents of an event and the data associated with it. However, the owner cannot change the version number or the name of the event.
- Create New Version—If instead, the owner wants to keep the original `NewCustomerEvent` but wants create a new version of the information with modified data, the owner can create a new version. When this version is saved, there are now two objects—`NewCustomerEvent/OAI/V1` and `NewCustomerEvent/OAI/V2`.
- Load Version—Not all versions of objects are loaded into iStudio. To work with a specific version of an object, use the **Load Version** capability. When a new version is created, it becomes the current version.
- Copy Object—To create a `NewBigCustomerEvent` which has many common elements with `NewCustomerEvent/OAI/V1`, first load `NewCustomerEvent/OAI/V1`and copy the object in iStudio. Copying the object allows not only modifications to the data, but also modifications to the name of the event. When the name of the `NewBigCustomerEvent/OAI/V1`event has been modified, `NewCustomerEvent/OAI/V1`will coexist in the Repository.

**Note:** Names of events must be unique.

In the example, all the metadata is built at Oracle Corporation and this metadata can be transmitted to the customer, NewCorp. When NewCorp installs the repository and specifies the owner as `NewCorp`, the metadata is in a read-only state. If NewCorp wants to customize `NewBigCustomerEvent/OAI/V1`, they cannot modify the existing version since the owners are different. However, they can use the other features described.

To customize the metadata, NewCorp must create a new version so that `NewBigCustomerEvent/OAI/V1` and `NewBigCustomerEvent/NewCorp/V2` coexist in the repository. NewCorp can use both events in defining messages if required and NewCorp can now modify the event it owns.

## Event Maps

Event maps allow you to map application data to an Oracle Applications InterConnect event without the application having to know about the Oracle Applications InterConnect event itself. For example, if an application is publishing a Create Customer event, it doesn't have to explicitly say that the message it is publishing corresponds to an Oracle Applications InterConnect Create Customer event. Instead, in iStudio, you can associate certain fields in the application view to help Oracle Applications InterConnect figure out which event the message maps to.

Event maps allow application data to be mapped to an Oracle Applications InterConnect event without the application needing to know about the event itself. For example, if an application publishes a Create Customer event, it does not need to explicitly specify that the message it is publishing corresponds to the Oracle Applications InterConnect Create Customer event. Instead, using iStudio, certain fields can be associated with the application view to help determine which event the message maps to.

In addition, if an application publishes exactly the same structure of data for two or more events, event maps help Oracle Applications InterConnect distinguish which message corresponds to which event. For example, an application publishes the same Customer Application Data Type regardless of whether it is a Create Customer or an Update Customer event. Through event map, Oracle Applications InterConnect can determine which messages correspond to Create Customer and Update Customer.

## Content-Based Routing

Messages can be routed to specific applications based on business rules or message content. For example, a procurement system can route fulfillment requests to different fulfillment centers based on originating location or item requested. iStudio provides support for defining the business rules through wizard-based point-and-click steps.

## Cross Reference Tables

Keys for corresponding entities created in different applications can be correlated through cross referencing. For example, a purchase order created in a procurement system has a native id X. It is then routed to a fulfillment system. The purchase order is created in the fulfillment system with native id Y. X and Y must be cross referenced so Oracle Applications InterConnect can correlate communication about this same logical entity in two different systems without each system knowing the native ids of the other.

## Domain Value Mapping

Code tables can be mapped across systems using domain value mapping. For example, a purchase order in a procurement system has a purchase order status field with possible domain values of `Booked` and `Shipped`. The corresponding field in a fulfillment system has the domain value set of `1` and `2`. Oracle Applications InterConnect creates mappings such as `booked=1` and `shipped=2` so these values can be correlated at runtime without each system knowing the domain value set of the other.

## Routing and the Message Capability Matrix

In the Oracle Applications InterConnect hub, Advanced Queues in the database are used to store, route, and forward messages from the sending application adapters to the receiving application adapters. The following paradigm is used for routing messages. The sending adapters evaluate who the recepients are based on metadata.

1. Every adapter has one or more queues where it receives messages.
2. The Message Capability Matrix allow queues to be specified for receiving messages on a per message per receiving application basis.

**Note:** By default, there is only one queue call the `oai_hub_queue`. This queue is used for all messages for all applications. This queue does not need to be changed unless the single queue implementation turns out to be a performance bottleneck.

### Tracking Fields

Tracking fields are one or more application view fields in the context of a particular message. If specified in iStudio, tracking fields can be used to track messages at runtime using the Oracle Applications InterConnect Runtime Management Console. Tracking is executed only from the perspective of the sending application.

For example, if App1 publishes a new purchase order and specifies the purchase order number field as the tracking field, then the user can log into the runtime console and specify the message to track, or New Purchase Order in this case. The user is then prompted to enter the purchase order number to display the corresponding tracking information.

## Using iStudio

iStudio is the graphical development tool that implements the Oracle Applications InterConnect concepts.

### Using the iStudio Toolbar

The following graphic displays the iStudio toolbar. You can also select the tasks by clicking the icons in the toolbar.

## Workspaces and Projects

The following topics discuss workspaces and projects.

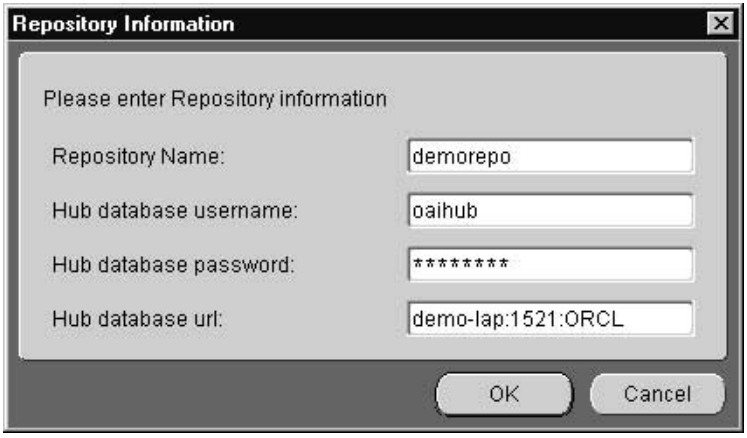
### Creating a New Project

For more information on projects, see "Projects". To define a new project in iStudio:

1. From the File menu, select New Project in the iStudio main menu panel.

   The New Project Dialog displays:

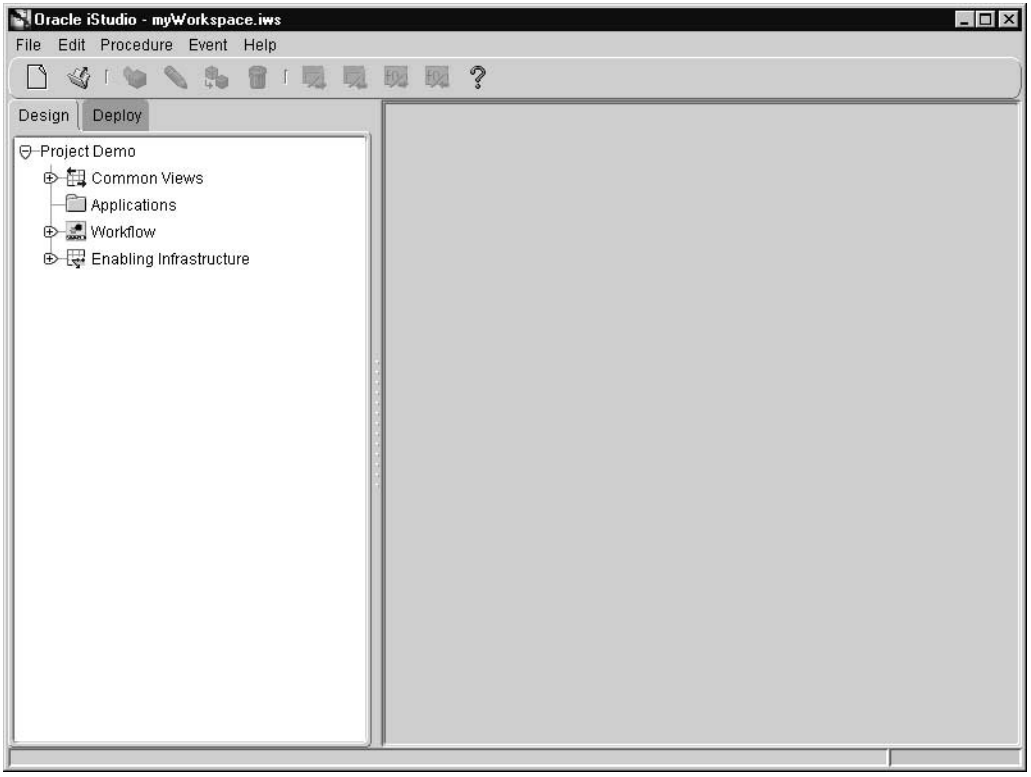

2. Enter the project name and click OK. The Repository Information dialog displays:




3. Enter information in the following fields:
   - Repository Name—The name of the repository server.
   - Hub database username—The name of the hub database user.

- Hub database password—The password associated with the hub database user.
- Hub database url—Information of the form "machine name:port number:database sid".

4. Click OK. iStudio is now ready to work on the specified project.



## Using Workspaces

When you start iStudio, the default workspace `myWorkspace.iws` and the last opened project is automatically loaded. To save the SAP and DB login credentials, check the Save settings as default checkbox in the SAP and DB login dialogs. The user settings are automatically saved to the workspace. For more information on workspaces, see "Workspaces" on page 2-8.

**Creating a New Workspace** To create a new workspace:

1. From the File menu, select New Workspace. The New Workspace Dialog displays:


2. Enter a name for the Workspace in the Workspace Name field.
3. Click OK.

**Opening an Existing Workspace** To open a workspace that has already been created:

1. From the File and select Open Workspace. The file system dialog displays:

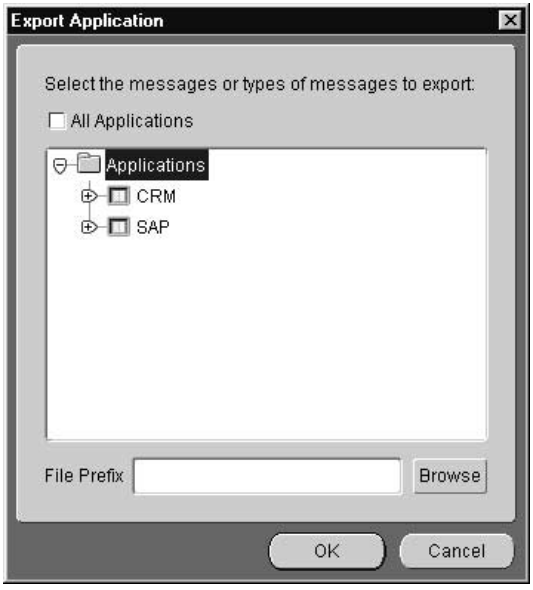

2. Enter the workspace name and path to open the workspace and click OK.

## Exporting Stored Procedures

iStudio generates stored-procedure stubs to enable an application to interface with the Oracle Applications InterConnect run-time easily. These stubs are exported to a file using the export functionality.

To export stored-procedures:

1. From the iStudio main menu panel, select File, then select Export. The Export Application dialog displays:




2. Select messages to export stored procedures. Messages can be filtered as follows:
   - Export all messages—Select Applications at the top of the directory.
   - Export all messages of a certain type for all applications—Check All Applications, then select one or more types of messages to export.
   - Export all messages for a specific application—Select the application name.
   - Export all messages of a certain type for a specific application—Select the type under the application name in the directory.
   - To export specific messages—Select the messages by name. To select more than one message or class of messages click the application.

3. In the File Prefix field, enter the name of the file to contain the exported stored procedures. The name generates multiple files. Click Browse to view the directory path.
4. Click OK. The stored-procedure is now exported.

## Common View

### Creating Business Objects

For more information on business objects, see "Business Objects" on page 2-10. To create a new business object:

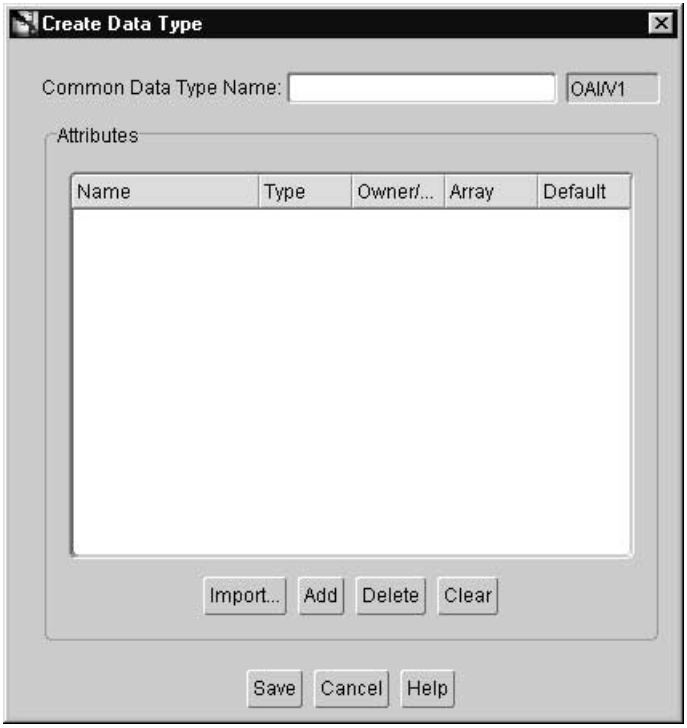
1. From the File menu select New, then select Business Object. The Create Business Object dialog displays:


2. Enter a name for the business object in the Business Object Name field.
3. Click OK.

### Creating Common Data Types

For more information on common data types, see "Common Data Types" on page 2-11. To create a common data type:

1. From the File menu select New, then select Common Data Type. The Create Data Type window displays:




2. Enter a name for the common data type in the Common Data Type field.

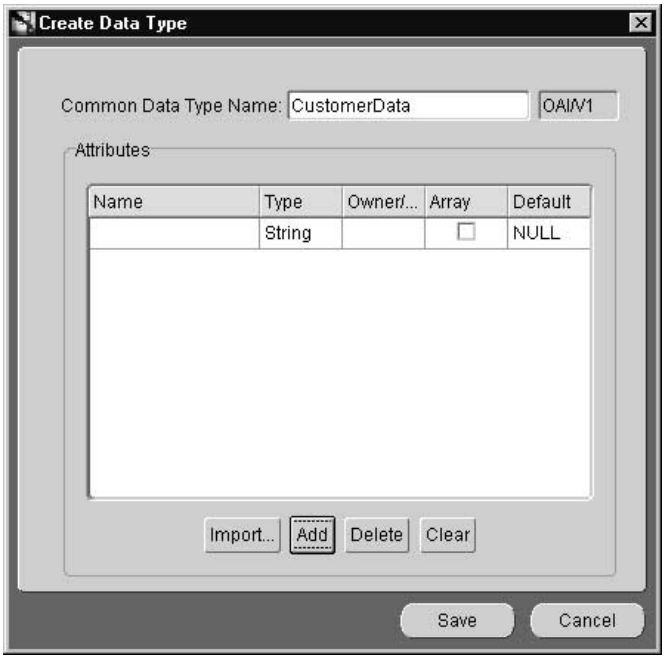
   The owner and version number of the common data type display next to the common data type name. This field cannot be edited.

3. Now you must specify the attributes for this common data type. There are two ways to specify attributes:

   - Add attributes one by one.
   - Import attributes from already existing application native data types or APIs.

**Adding Attributes One by One** To add attributes one by one:

1. On the Create Data Type dialog, click Add. A new entry appears in the attribute list as shown:

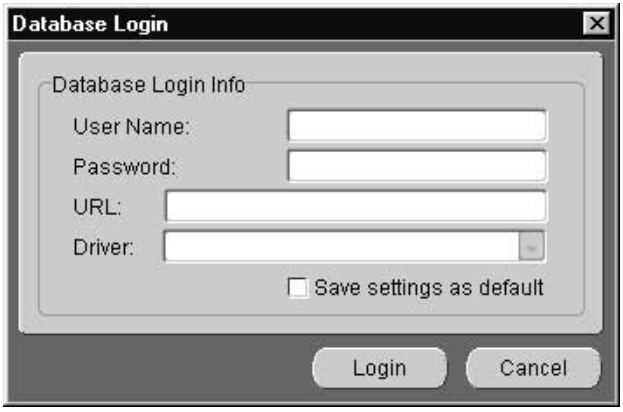



2. Specify the following by editing the information directly in the attribute list entry.
   - Name—The name of the attribute.
   - Type—The type of the attribute. Select the type by clicking on the Type column in the attribute entry. A dropdown list displays. The attribute can be of primitive type (string, integer, float, double, date) or another common data type. You can build hierarchical data types by using the latter.
   - Array—Check this box if it the attribute is a collection instead of a singleton.
   - Default— The default value of the field in case it is not populated at runtime.
3. Click Save. Repeat the above steps for adding other attributes.

**Importing Attributes** To import attributes:

1. On the Create Data Type dialog, click Import. Attributes can be imported from the following sources:
   - Other Common Data Types
   - Database
   - SAP ABAP Modules
   - SAP BAPI
   - SAP IDOC
   - XML

   The following example utilizes the Database import facility.
2. Click Database. The Database Login dialog displays:

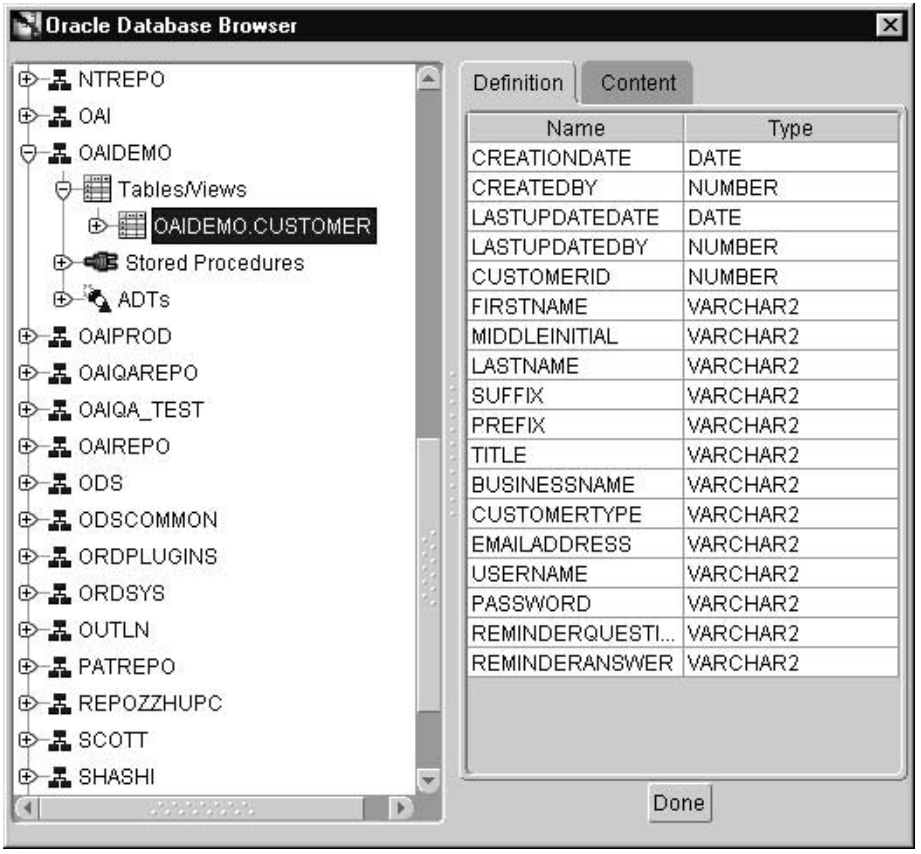



3. Enter information in the following fields:
   - User Name—The log in name.
   - Password—The log in password.
   - URL—The machine name: port number: database SID.
   - Driver—The JDBC driver used to connect to the database.
   - Save settings as default—Check this box to save the settings for the workspace.

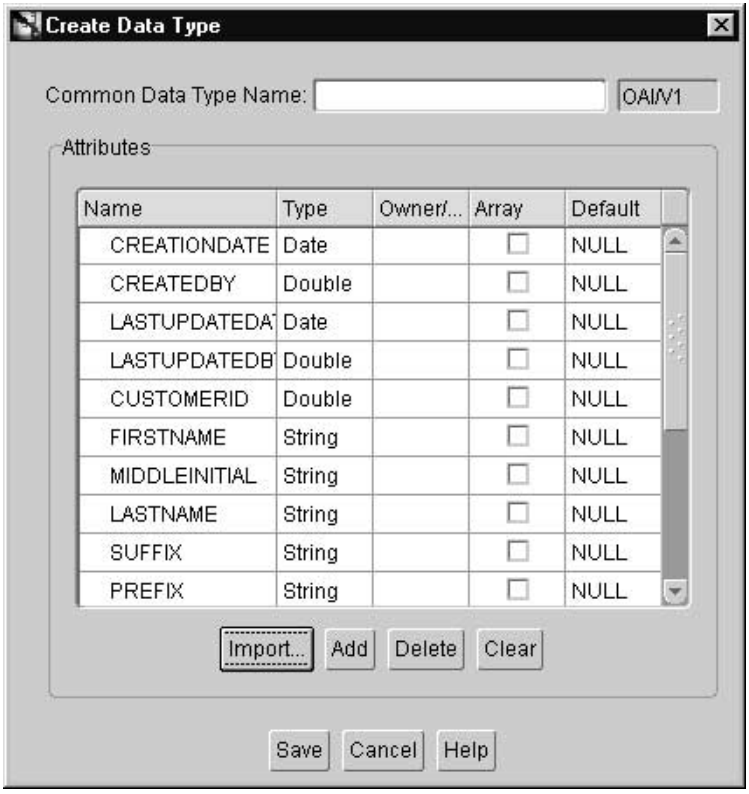
4. Click Login.

   After logging in, the database tables and arguments display in the Database Browser Window.

   Select the fields by clicking on them.




5. Click Done to import the attributes into the common data type.




**Deleting and Clearing Attributes** To delete a selected attribute:

- On the Create Data Type dialog, select the attribute to be deleted and click Delete.

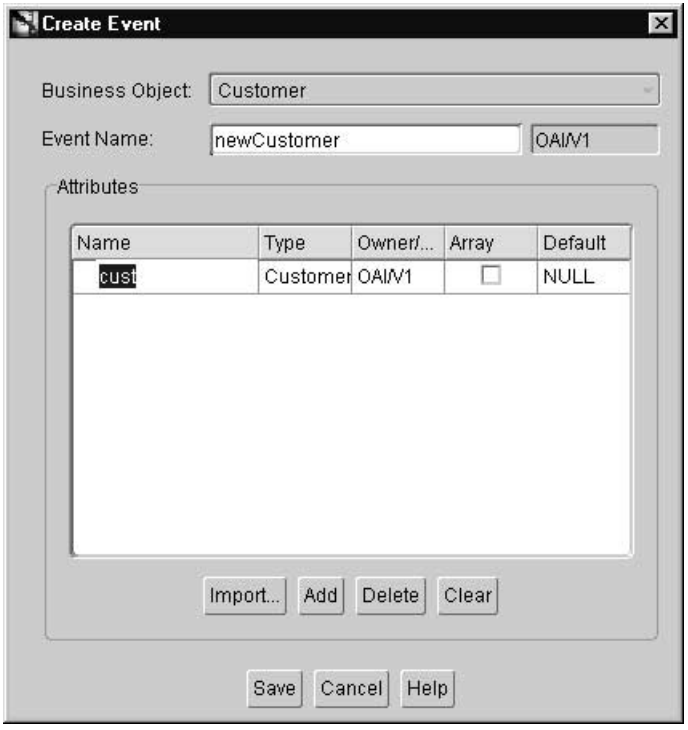
To clear all attributes:

- On the Create Data Type dialog, click Clear.

### Creating Events

For more information on events, see "Events" on page 2-10. To create an event:

1. From the iStudio main menu panel, select Project <project name>, then click New.
2. From the New drop down list, select Event. The Create Event dialog displays:




3. Enter the information in the following fields:
   - Business Object Name—The name of the category to which the event belongs.
   - Event Name—The name of the event. Only alphanumeric characters can be used.
   - OAI/V1—The owner and version number of the Business Object. This field cannot be edited.

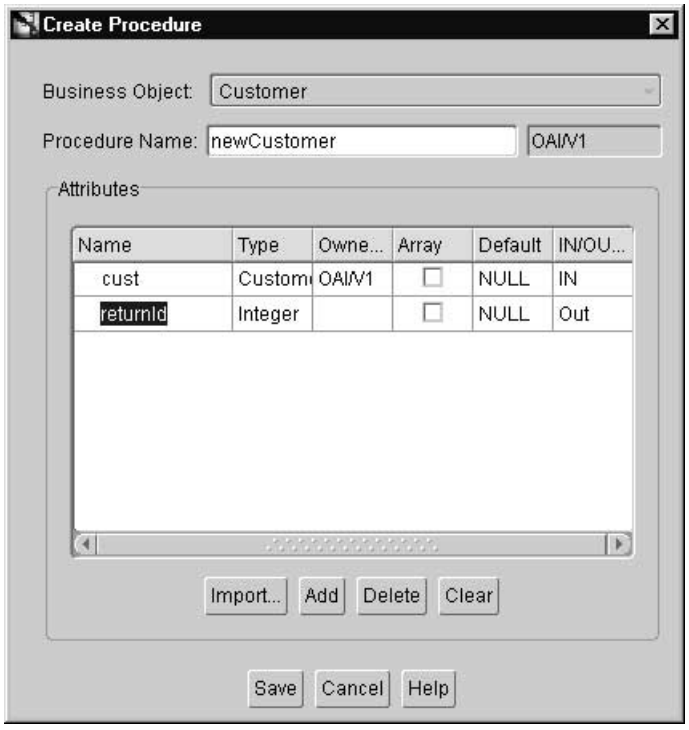
4. To add attributes to this event, enter information in the following fields:
   - Name—The name of the attribute.
   - Type —The type of event. Possible values include integer, string, date, float, double, data type. Previously defined data types can be used.
   - Array—Check this box if the attribute is an array field. Only user-defined data types can be of type array.
   - Default—The default value for the field is NULL.
5. Click Save.

### Creating Procedures

For more information on procedures, see "Procedures" on page 2-11. To create a procedure:

1. In the iStudio main menu panel, select File, then click New.

2. From the New drop down list, select Procedure. The Create Procedure dialog displays:




3. Enter information in the following fields:

   - Business Object Name—The name of the category to which the procedure belongs.
   - Procedure Name—The name of the procedure. Only alphanumeric characters can be used.
   - OAI/V1—The owner and version number of the procedure. This field cannot be edited.

4. To add arguments to this procedure:
   - Name—The name of the attribute.
   - Type —The type of the attribute. Types include integer, string, date, float, and double data type. A common data type that has been previously defined can be selected.
   - Array—Check this box the attribute is an array field. Only user-defined data types can be of type array.
   - Default—The default value for the field is NULL.
   - IN—The input parameter.
   - OUT—The output parameter.
   - INOUT—The input and return parameter.
5. Click Save to save and exit or Save As to save the procedure as a different version.

For more information on adding attributes, see "Adding Attributes One by One" on page 2-24.

For more information on importing attributes, see "Importing Attributes" on page 2-25.

For more information on deleting and clearing attributes, see "Deleting and Clearing Attributes" on page 2-27.

## Application View

The following topics discuss application view information in iStudio.

### Creating an Application

To create an application:

1. From the iStudio main menu panel, click File.

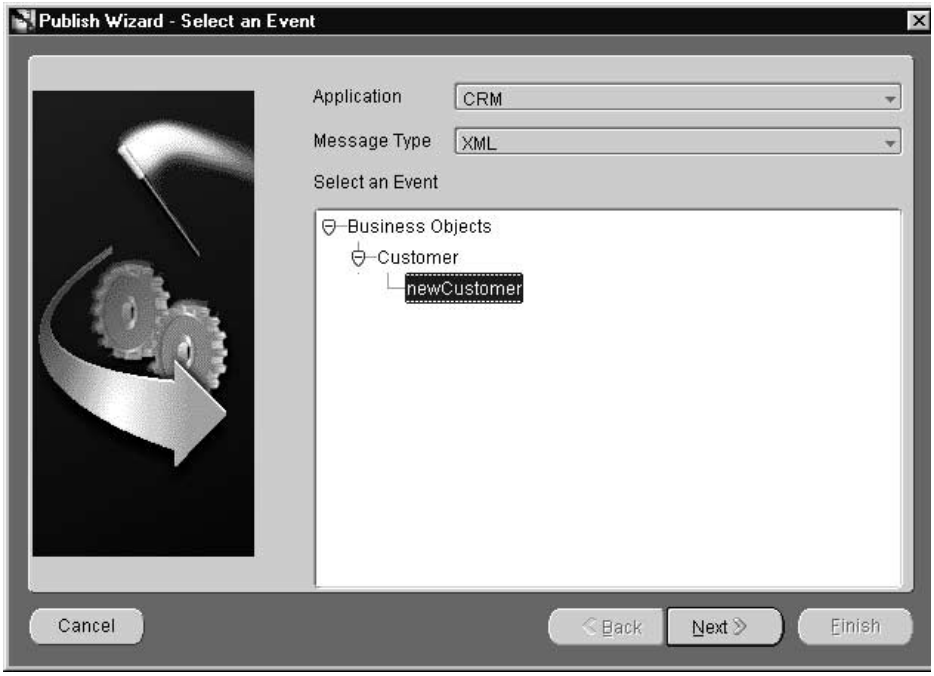
2. From the File menu, click New in the pull-down menu and select Application. The Create Application dialog displays:


3. Enter a name for the application in the Application Name field.
4. Click OK.

### Publishing an Event

To publish an event in an application:

1. From the iStudio main menu panel, select Event, then click Publish in the pull-down menu. The Publish Wizard dialog displays:




The Event Publish Wizard provides a series of pages to follow for creating a publish event. This wizard provides a series of pages for creating a publish event.

2. **Select an Event Page**

   a. Enter information in the following fields:

   - Application—The name of the application which is publishing the event.
   - Message Type—This field specifies the mode of communication between the Adapter and the application, for example, XML.

Select from the following message types:

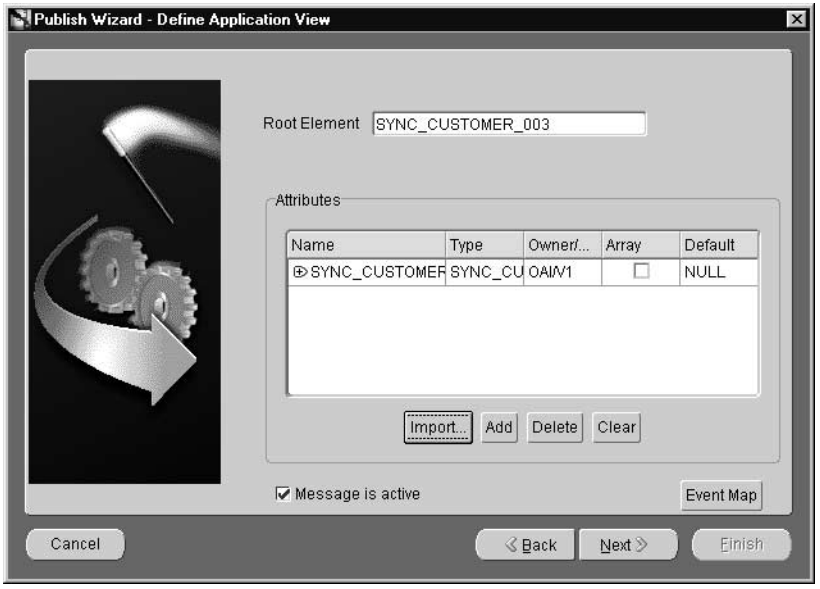
* Database—The Adapter selects the message data from the database.
* SAP BAPI—The Adapter communicates with the application using Business API.
* SAP IBP—The Adapter communicates with the application using IBP.
* SAP IDOC—The Adapter communicates with SAP using IDOC.
* XML—The Adapter communicates with the application using XML.
* Generic—The Adapter communicates with application using a user-defined bridge.

**b.** Select the event name.

**c.** Click Next in the Publish Wizard.

**3. Define Application View Page**

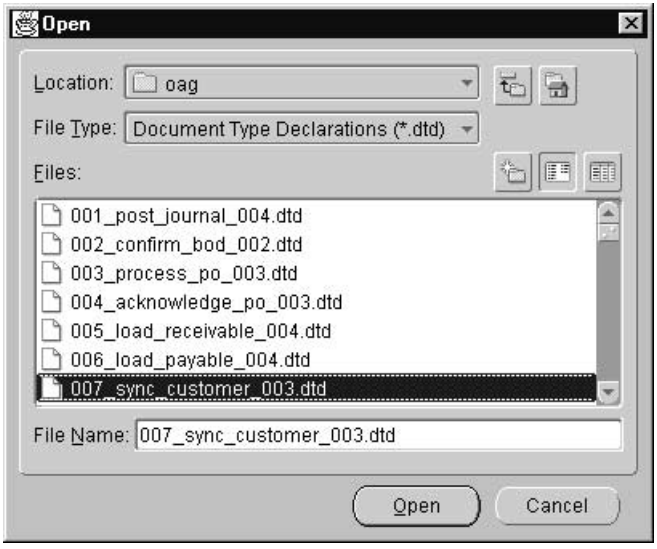
Using this page, the application view is defined. This page is initially an empty table. Define the attributes using Add or import the definitions from a database or an API Repository using Import.



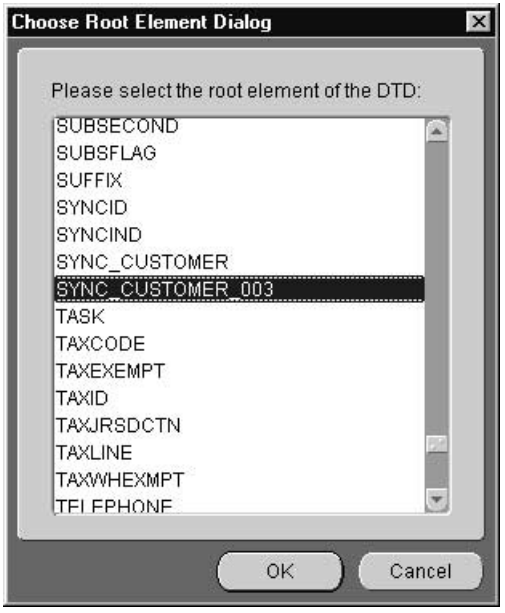
**a.** Enter information in the following fields:

* Name—The name of the attribute.
* Type—The attribute type. Types include integer, string, date, float, or double data type.
* Owner / Version—The owner and version of this application view.
* Array—Check this box if it is an array field. Only user-defined data types can be of type array.
* Default—The default value for the field is NULL.

**b.** To import an XML DTD, a file dialog is displayed:

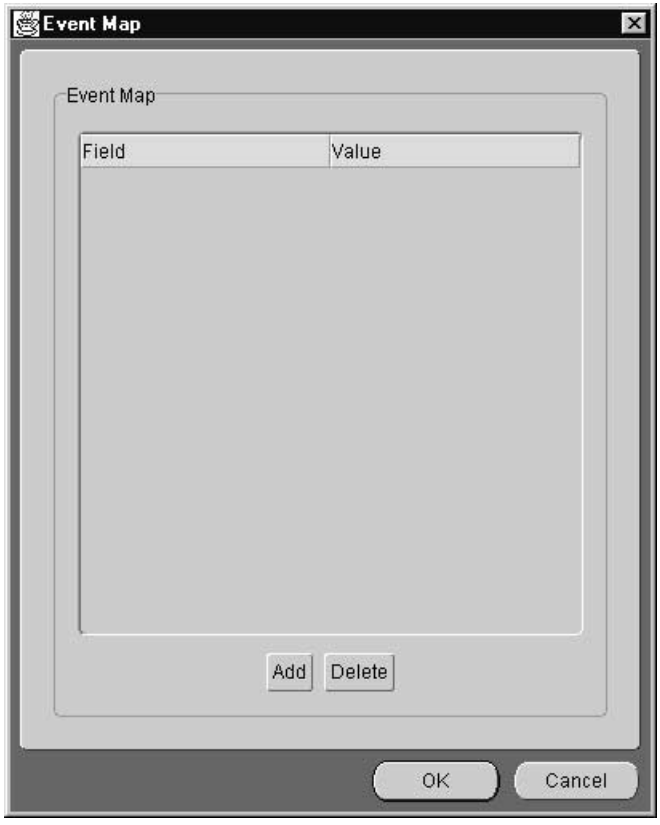


**c.** Select a DTD file and click Open. The Choose Root Element dialog displays:



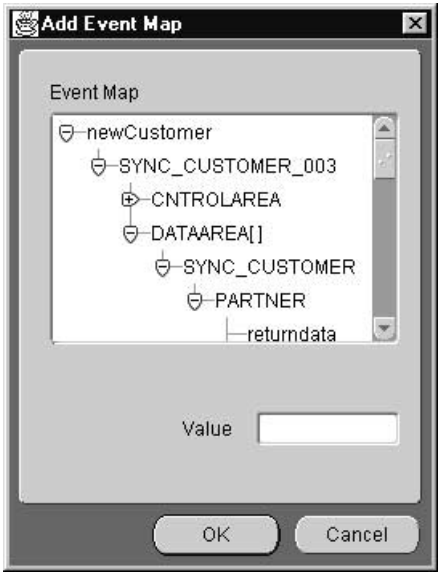

**d.** Select a root DTD element and click OK.

If this is a XML type message, the Event Map button is enabled. To define the event map, click Event Map. The Event Map dialog displays:



To add an event map attribute:

**a.** Click Add. The New Event Map dialog displays:




**b.** Choose an attribute from the tree and enter a value in the text field.

**c.** Click OK on the Add Event Map dialog.

**d.** Click OK to continue to the Publish Event Wizard.

To delete an event map item:

**a.** Select the event map item to delete and click Delete.

**b.** Click Next in the Publish Event Wizard.

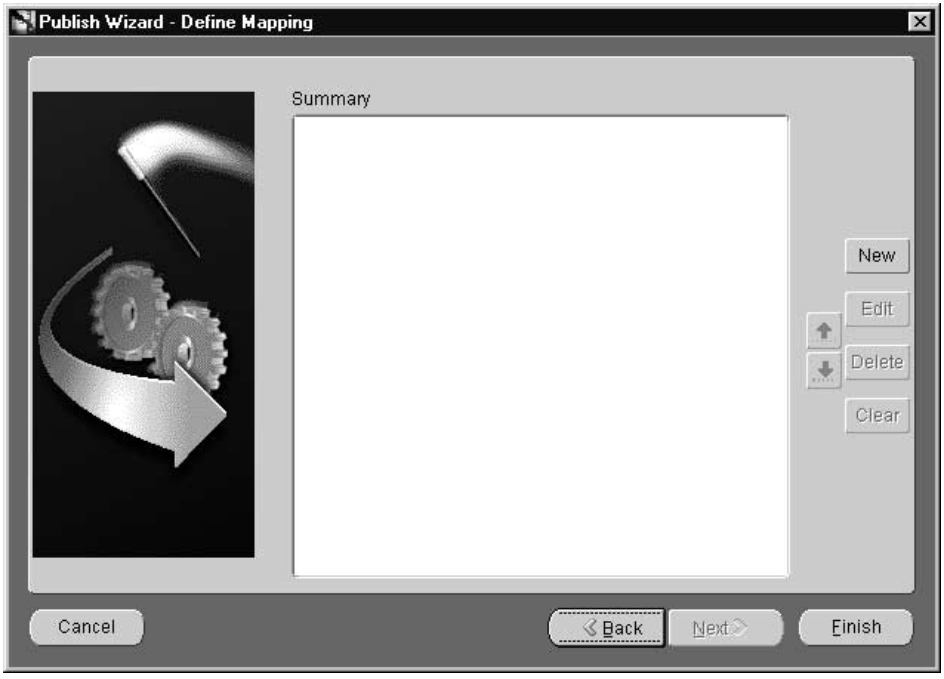
For more information on adding attributes, see "Adding Attributes One by One" on page 2-24.

For more information on importing attributes, see "Importing Attributes" on page 2-25.

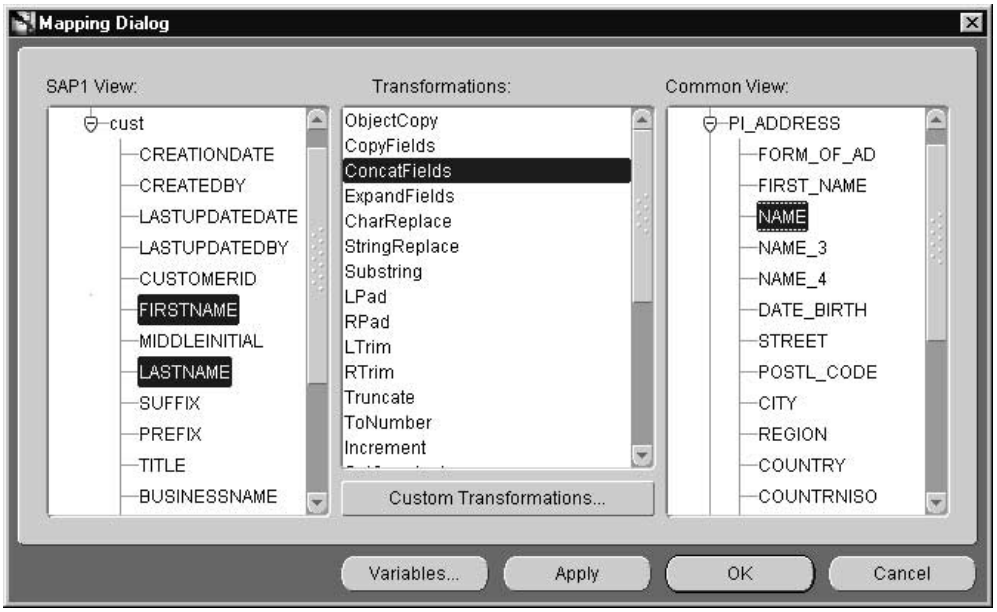
For more information on deleting and clearing attributes, see "Deleting and Clearing Attributes" on page 2-27.

### 4. Define Mapping Page

Mapping can either involve copying the individual fields, or simple shape-change transformations.



**a.** To define new mappings, click New. The Mapping dialog displays:

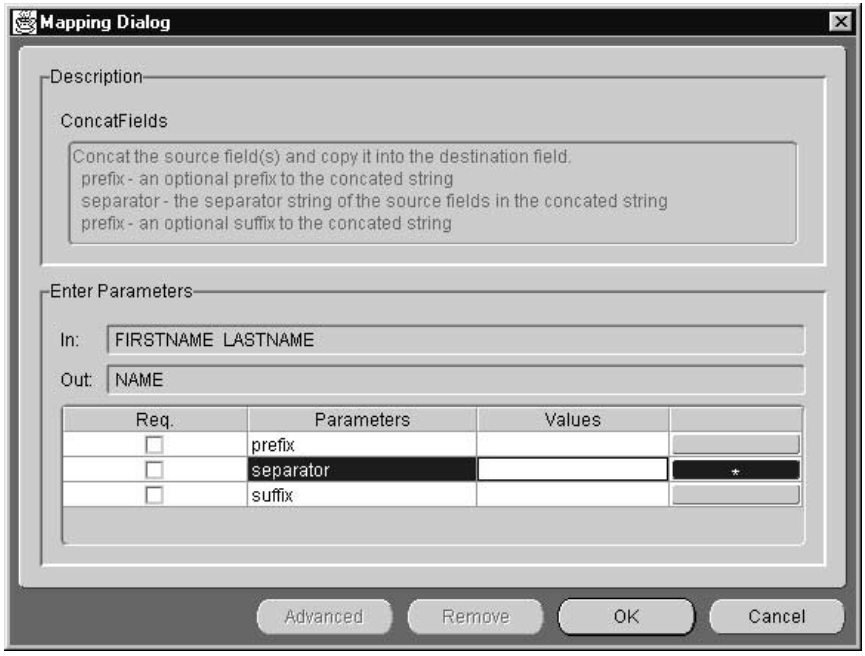



To map fields in the application view to fields in the common view, use the `concat` transform. For example, to map fields in the `FirstName` and `LastName` in the application view to `Name` in the common view, use the concat transform.

The following steps illustrate this example:

1. Select fields to map from in the application view. Use the left mouse button to select multiple fields in a view.
2. Select the transformation to perform, for example, `ConcatFields`. For information on custom transformations, see "Adding Custom Transformations" on page 2-73.
3. Select the fields to map to in the common view. Use the left mouse button to select multiple fields in a view.
4. Click Apply to confirm selection and continue specifying additional mappings.
5. When all mappings have been made, click OK.

The transformation may have parameters. After clicking Apply or OK, the Mapping Parameters dialog may display:



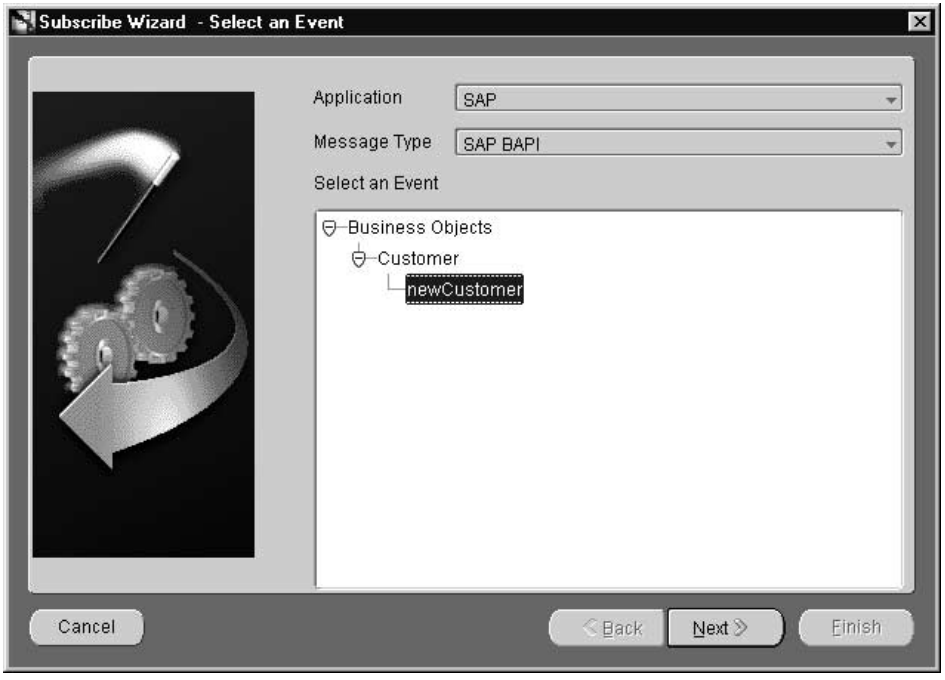

In the Parameters field, enter the values for the transformation parameters. For example, a blank value indicates a value for the separator parameter.

6. Click OK to return to the Publish Event Wizard.
7. Click Finish on the Publish Event Wizard to create the publish event.

### Subscribing to an Event

To subscribe to an event in an application:

1. From the iStudio main menu panel, click Event, then click Subscribe from the pull-down menu. The Subscribe Wizard displays:




2. **Select an Event Page**

   a. Use this page to enter information in the following fields:

   * Application—The name of the application subscribing to the event.

   * Message Type—Specifies the mode of communication between the Adapter and the application, for example SAP Business API. Select from the following message types:

     Database—The Adapter picks the message data from the database.

     SAP BAPI—The Adapter communicates with the application using Business API.

     SAP IBP—The Adapter communicates with the application using IBP.

SAP IDOC—The Adapter communicates with SAP using IDOC.

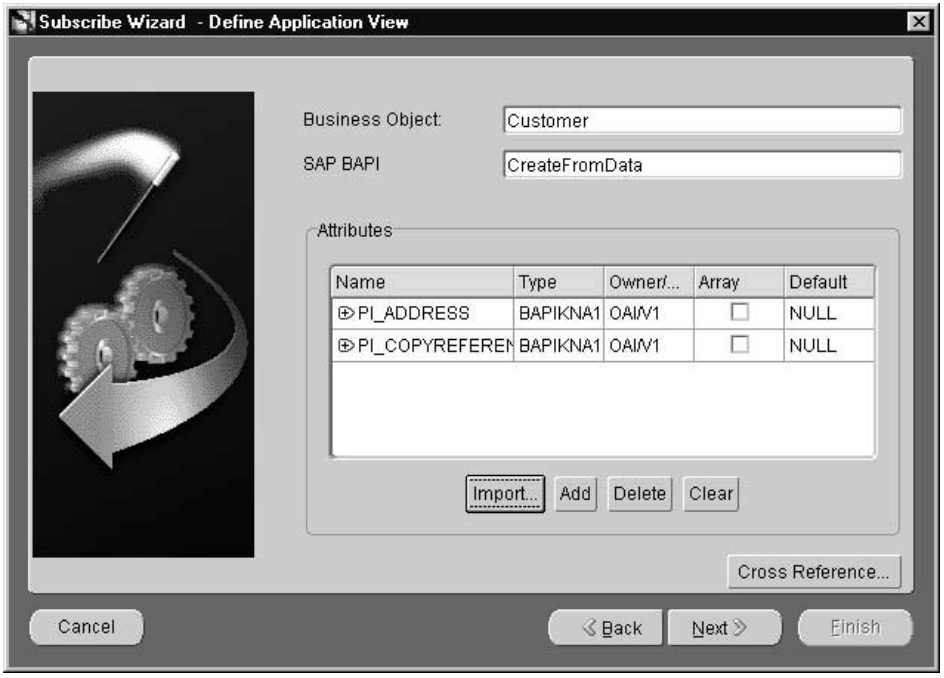
XML—The Adapter communicates with the application using XML.

Generic—The Adapter communicates with application using a user-defined bridge.

**b.** Select the event to subscribe and click Next.

### 3. Define Application View Page

After selecting the event to publish, the application view is defined on the Define Application View Page. The application view window is initially an empty table. Use Add to define attributes or import definitions from a database or API repository using Import.




**a.** Enter information in the following fields:

* Business Object—The name of the business object.
* SAP BAPI—The SAP Business API name which is imported.

**b.** Add, import, delete, or clear attributes in the Attributes box. The following attributes are available:

- Name—The name of the attribute.
- Type—The attribute type. Types include integer, string, date, float, or double data type.
- Owner / Version—The owner and version number of this application view.
- Array—Check this box if it is an array field. Only user-defined data types can be of type array.
- Default—The default value for the field is NULL.

**c.** To populate and look up cross reference tables, click Cross Reference... The Cross Reference dialog displays. For more information, see "Populating Cross Reference Tables" on page 2-69.

**d.** Click Next.

**4. Define Mapping Page**

Mapping can either involve copying the individual fields or simple shape change transformations.

To define mappings, use the Define Mappings page:


**a.** To define new mappings, click New to define mappings.

**b.** Click Finish.

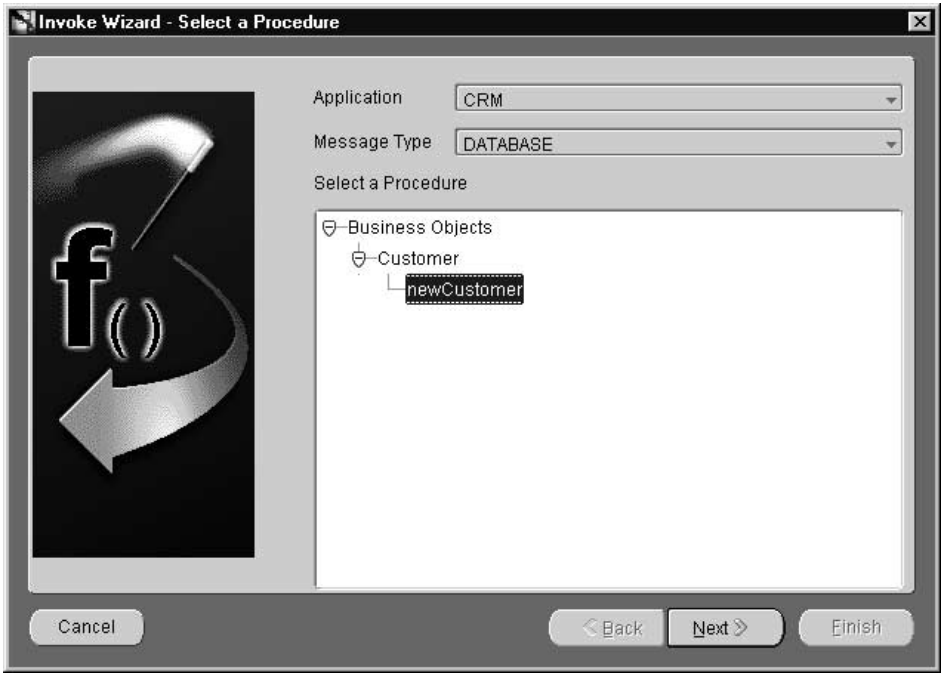
The subscribe event is created.

### Invoking an Procedure

To invoke a procedure in an application, from the iStudio main menu panel, click Procedure. Then click Invoke from the pull-down menu. The Invoke Wizard displays.

1. **Select a Procedure Page**

   When the Invoke Wizard is started, the Select a Procedure page displays:




   **a.** Use this page to enter information for the following fields:

   * Application—The name of the application invoking the procedure, for example iStore.

   * Message Type—Specifies the mode of communication between the Adapter and the application. Select from the following message types:

     Database—The Adapter picks the message data from the database.

     SAP BAPI—The Adapter communicates with the application using Business API.

SAP IDOC—The Adapter communicates with SAP using IDOC.

XML—The Adapter communicates with the application using XML.

Generic—The Adapter communicates with application using a user defined bridge.

---

**Note:** In this release of Applications InterConnect, to invoke a procedure the message type can only be a database, XML, or Generic.

---

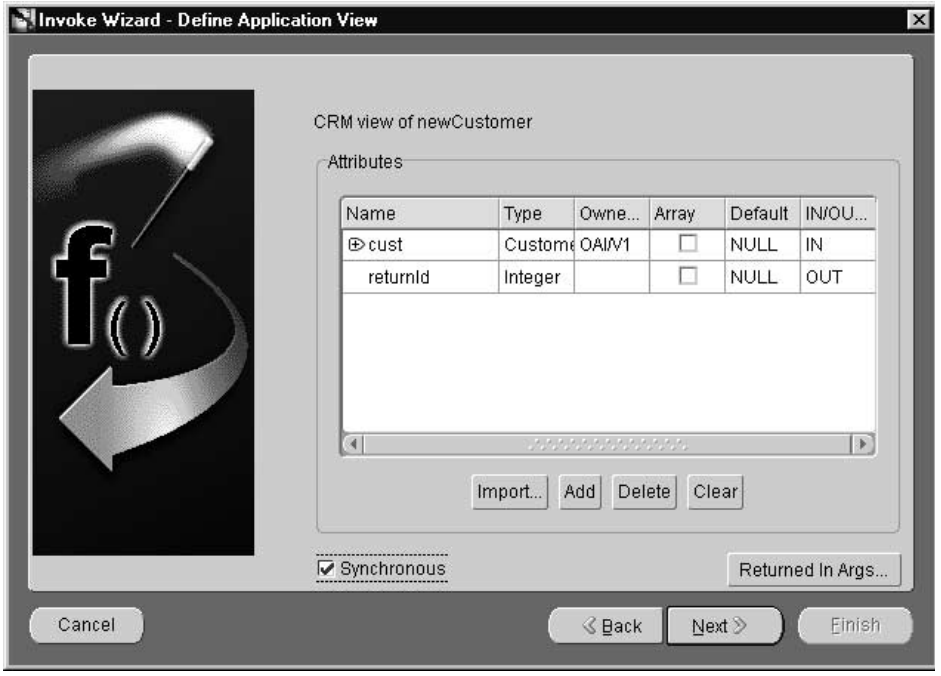
**b.** Select the procedure to invoke in the Select a Procedure box.

**c.** Click Next.

**2. Define Application View Page**

After selecting the procedure to invoke, the application view is defined. The application view dialog is initially an empty table. Attributes are defined by using the add button. Attribute definitions can be imported from a database or an API Repository using Import.

After clicking Next on the Select a Procedure page, the Define Application View page displays:




**a.** Enter information in the following fields:

- Name—the attribute name.
- Type—integer, string, date, float, or double data type.
- Owner / Version—the owner and version of this application view.
- Array—check this box if it is an array field. Only user-defined data types can be of type array.
- Default—The default value for the field is NULL.

**b.** Add or import attributes by clicking Add or Import. The following attributes are included:

* Name—The attribute name.
* Type—The type of attribute. Types include integer, string, date, float, or double data type.
* Owner / Version—The owner and version of this application view.
* Array—Check this box if it is an array field. Only user-defined data types can be of type array.
* Default—The default value for the field is NULL.

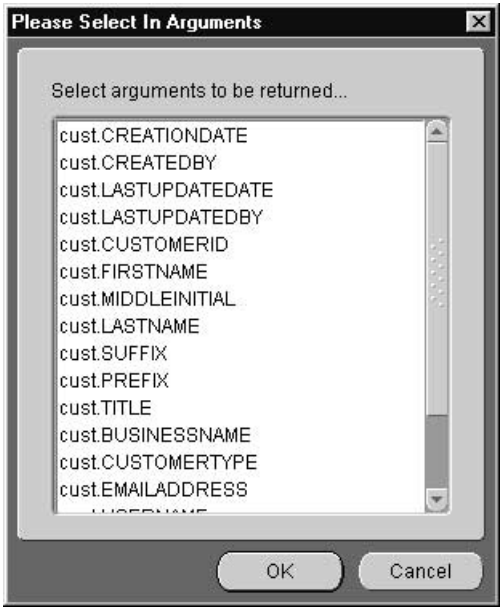
For information on adding attributes, see "Adding Attributes One by One" on page 2-24.

For information on importing attributes, see "Importing Attributes" on page 2-25.

For information on deleting and clearing attributes, see "Deleting and Clearing Attributes" on page 2-27.

**c.** Check the Synchronous box if this is a synchronous invoke, for example, the request will block till it gets a reply.

**d.** To specify IN arguments to be returned, click Returned In Args. The Select Returned In Arguments displays:



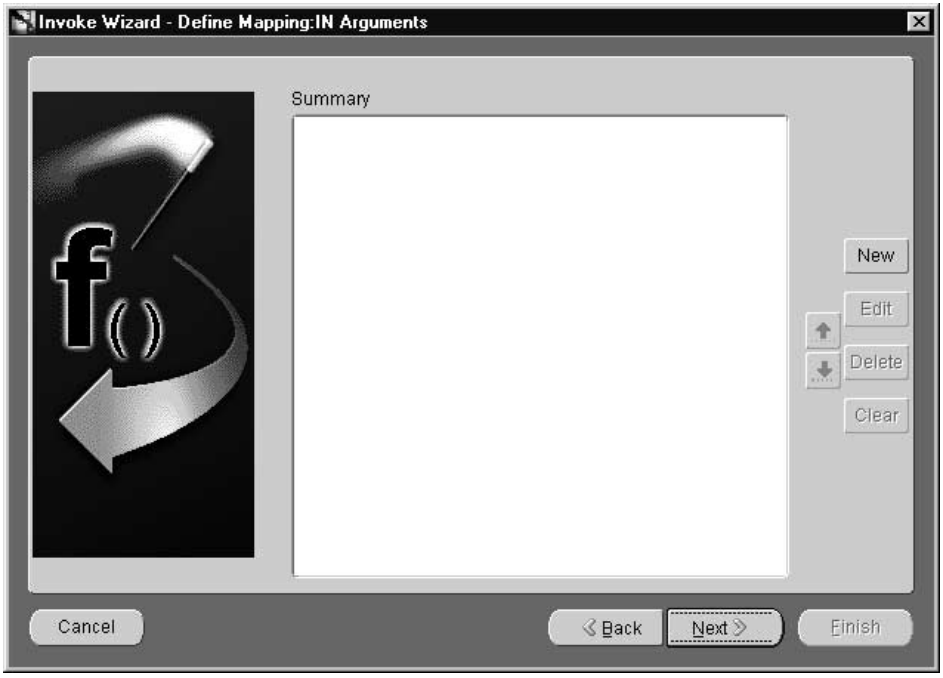

**e.** Select the input and output arguments to be returned. Use the left mouse button to select multiple arguments. Only non user-defined input arguments are shown for selection.

**f.** Click OK on the Select Returned Out Arguments to return to the Define Application View page.

**g.** Click Next.

### 3. Define Mapping IN Arguments Page

Mapping can either involve copying the individual fields or simple shape-change transformations. After clicking Next on the Define Application View page, the Define Mapping IN Arguments page displays:

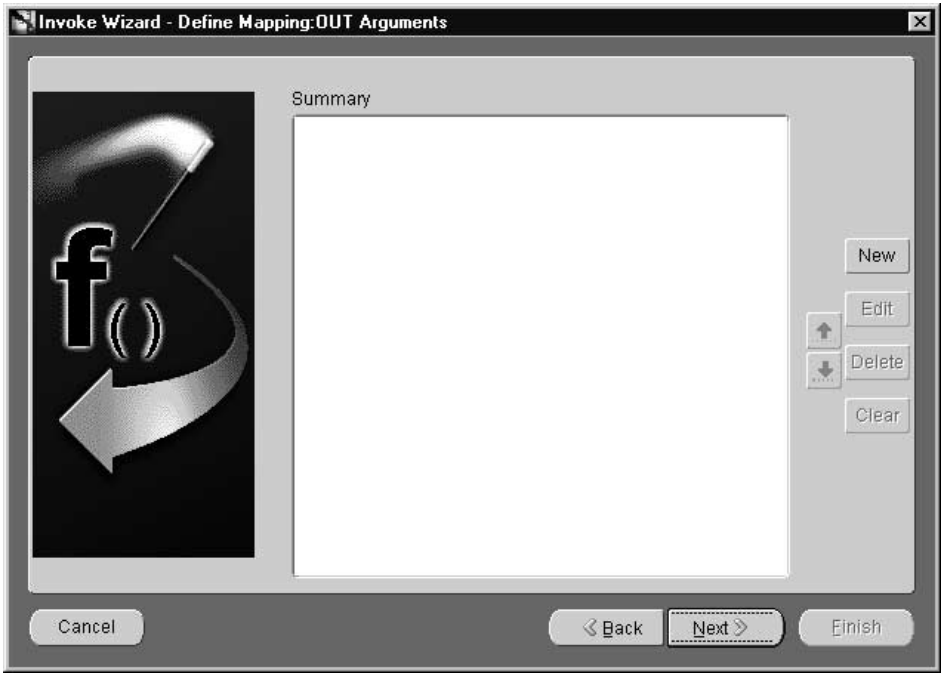



**a.** Click New to define mappings.

**b.** Click Next.

**4. Define Mapping OUT Arguments Page**

Mapping can either involve copying the individual fields, or simple shape-change transformations. You map the common view return arguments to the application view return arguments on this page.




**a.** Click New to define mappings.

**b.** Click Next.

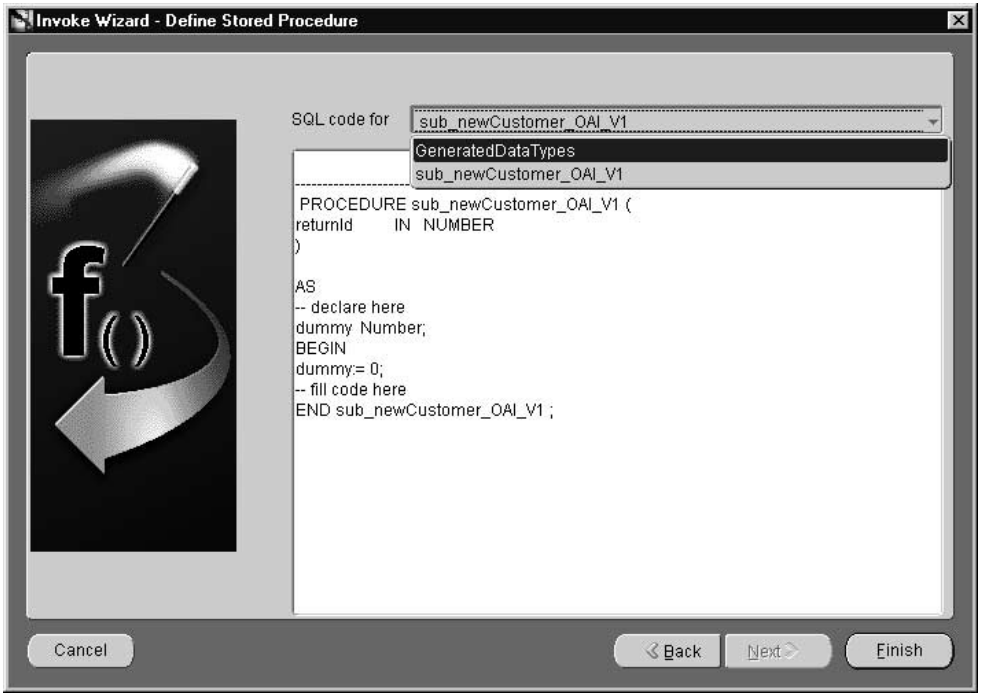
**5. Define Stored Procedure Page**

If the message type selected was database, the data is received by a stored procedure. In this stored procedure, the action performed when the values are returned to the application can be specified. The adapter invokes the stored procedure at runtime with the appropriate data.

The following arguments will be returned:

- All OUT arguments.
- All IN arguments specified to be returned as part of the reply.

After clicking Next on the Define Mapping OUT Arguments page, the Define Stored Procedure page displays:




**a.** Select a generated procedure from the SQL code for drop down list.

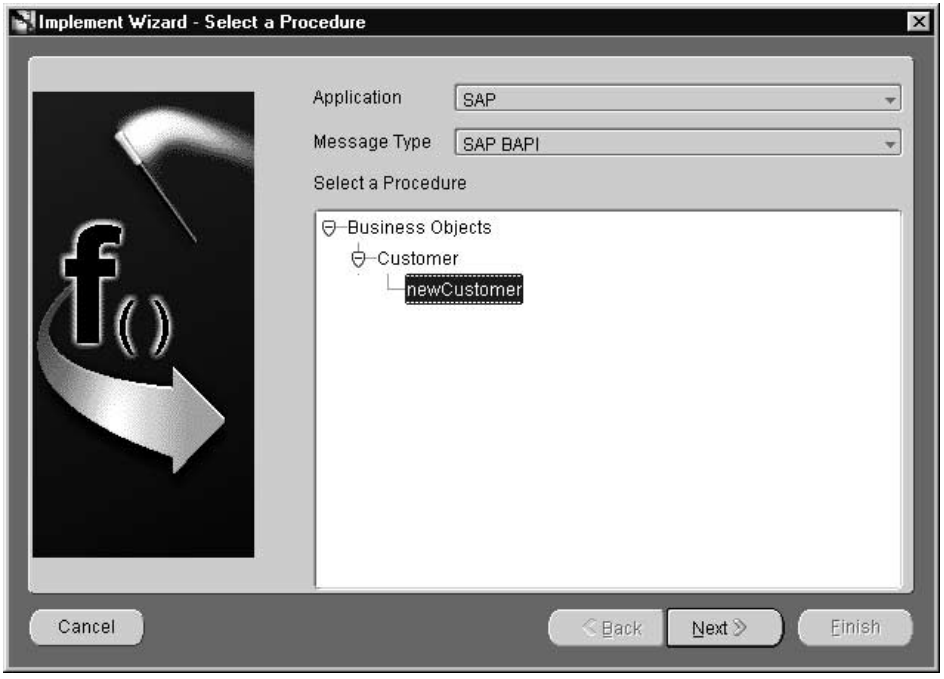
**b.** Click Finish.

The procedure invoke message is created.

### Implementing a Procedure

To implement a procedure:

1. From the iStudio main menu panel, select Procedure, then select Implement from the drop down list. The Select a Procedure page of the Implement Wizard displays:




2. **Select a Procedure Page**

   Use this page to select a procedure to implement.

   a. Enter information in the following fields:

      * Application—The name of the application implementing the procedure, for example SAP.

      * Message Type—Specifies the mode of communication between the Adapter and the application. Message types include:

        Database—The Adapter retrieves the message data from the database.

SAP BAPI—The Adapter communicates with the application using Business API.

SAP IBP—The Adapter communicates with the application using IBP.

SAP IDOC—The Adapter communicates with SAP using IDOC.

XML—The Adapter communicates with the application using XML.

Generic—The Adapter communicates with the application using a user-defined bridge.

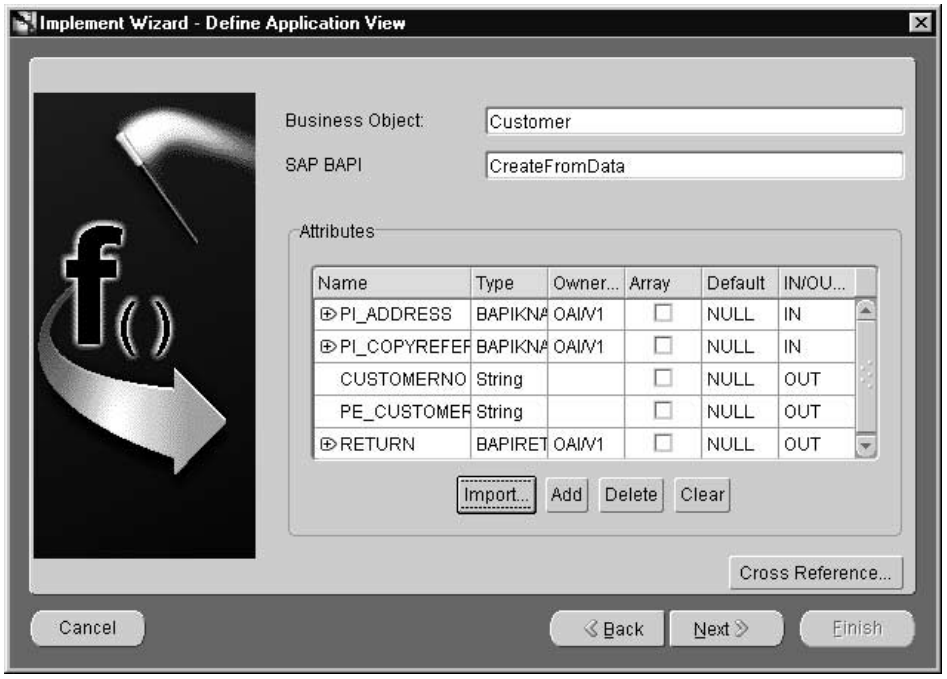
**b.** Select the procedure to invoke.

**c.** Click Next.

**3. Define Application View Page**

After selecting the procedure to implement, the application view is defined. The Define Application View page is initially an empty table. Attributes can be defined by using Add. Attribute definitions can be imported from a database or an API Repository by using Import.

After clicking Next on the Select a Procedure page, the Define Application View page displays:




**a.** Enter information in the following fields:

- Business Object—The name of the business object.
- SAP BAPI—The imported SAP Business API name.

**b.** Add or import attributes by clicking Add or Import. The following attributes are included:

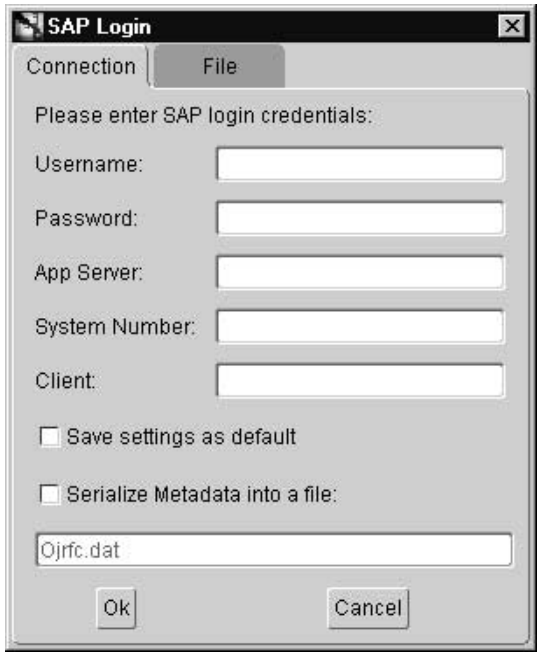
- Name—The attribute name.
- Type—The type of attribute. Types can be integer, string, date, float, or double data type.
- Owner / Version—The owner and version of this application view.
- Array—Check this box if it is an array field. Only user-defined data types can be of type array.
- Default—The default value for the field is NULL.

For information on adding attributes, see "Adding Attributes One by One" on page 2-24.

For information on importing attributes, see "Importing Attributes" on page 2-25.

For information on deleting and clearing attributes, see "Deleting and Clearing Attributes" on page 2-27.

**c.** If SAP Business API information is being imported, the SAP Login dialog displays:

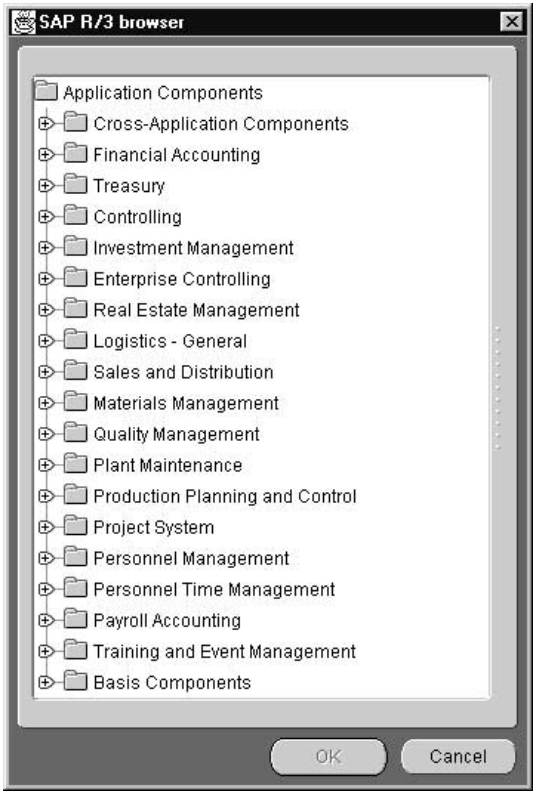



**d.** On the SAP Login dialog, enter the following information:

* User Name—The SAP username.
* Password—The password for the username entered.
* System Number—The number of the system being logged into.
* App Server—The application server name.
* Client—The client host name.

* Save settings as default—Check this box to save workspace settings as the default.
* Serialize Metadata into a file—Check this box to save SAP metadata to a file. Enter a file name in the field provided.

**e.** Click OK. The SAP application browser displays:



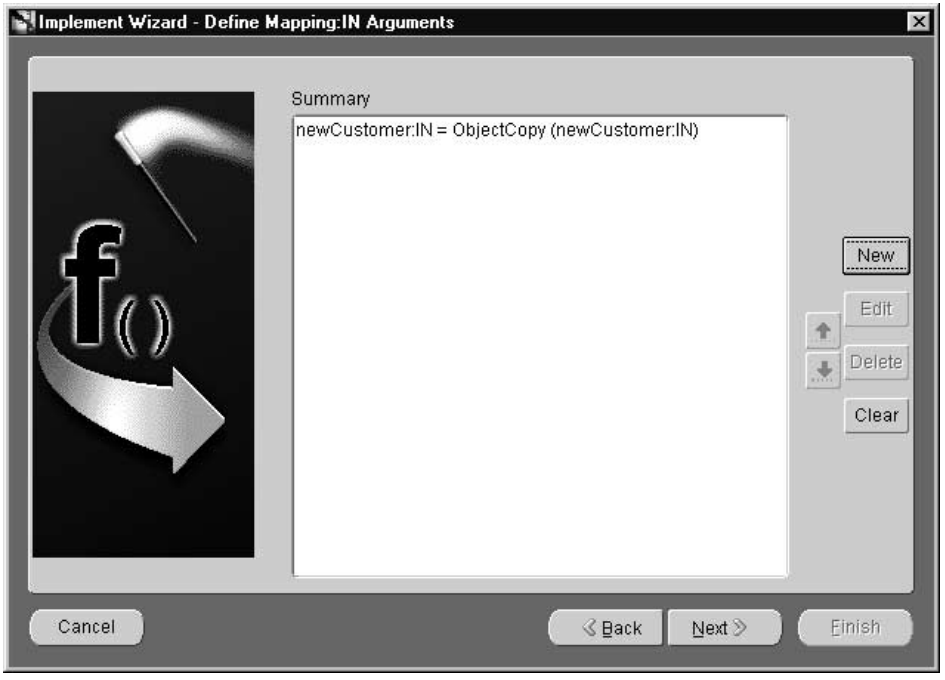

**f.** Select the SAP Business API and click OK to return to the Define Application View page of the Implement Wizard. The SAP Business API attributes are imported into the application view.

**g.** Click Cross Reference... to populate cross reference tables. For more information see "Populating Cross Reference Tables" on page 2-69.

**h.** Click Next.

### 4. Define Mapping IN Arguments Page

Mapping may involve copying individual fields, or simple shape-change transformations. After clicking next on the Define Application View page, the Define Mapping IN Arguments page displays:



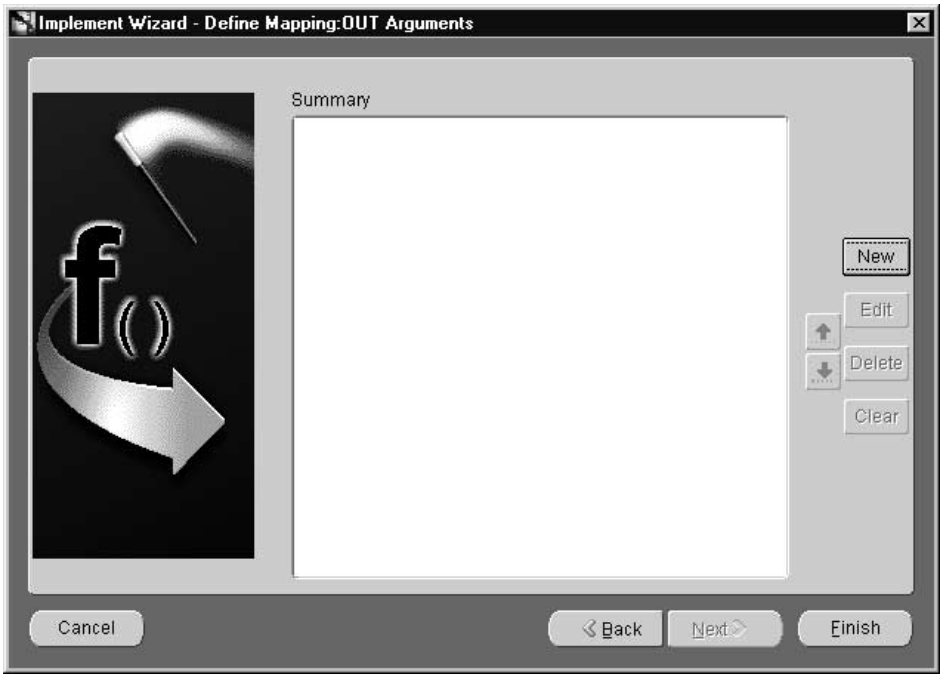

Use this page to define mapping IN arguments.

**a.** Click New to define mappings.

**b.** Click Next.

**5. Define Mapping OUT Arguments Page**

After clicking Next on the Define Mapping IN Arguments page, the Define Mapping OUT Arguments page displays:




Use this page to define mapping IN arguments.

**a.** Click New to define mappings.

**b.** Click Finish.

The procedure implement message is created.

## Enabling Infrastructure

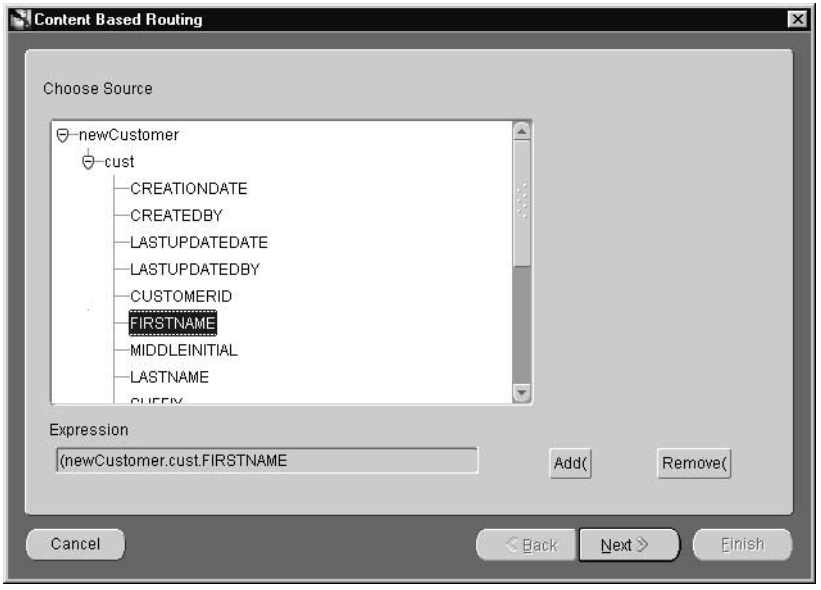
### Working with Content-Based Routing

For more information on content based routing, see "Content-Based Routing" on page 2-15. To modify content based routing for an event or procedure:

1. Right-click the event or procedure under the Content-Based Routing node in the iStudio design tree, then click Edit. The Content Based Routing Rules dialog displays:

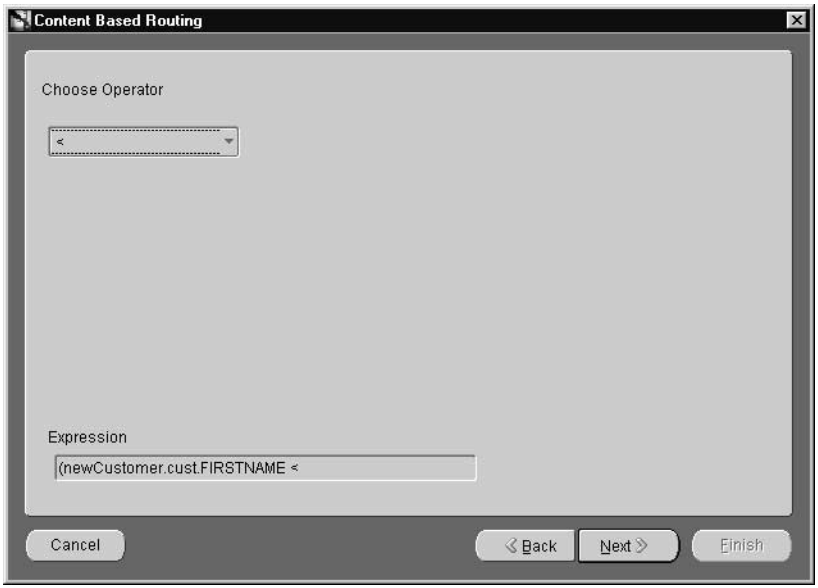

2. Click New to display the Content Based Routing Wizard. This wizard provides a series of pages to follow for editing content based routing. When the wizard starts, the Choose Source page displays.

### 3. Choose Source Page



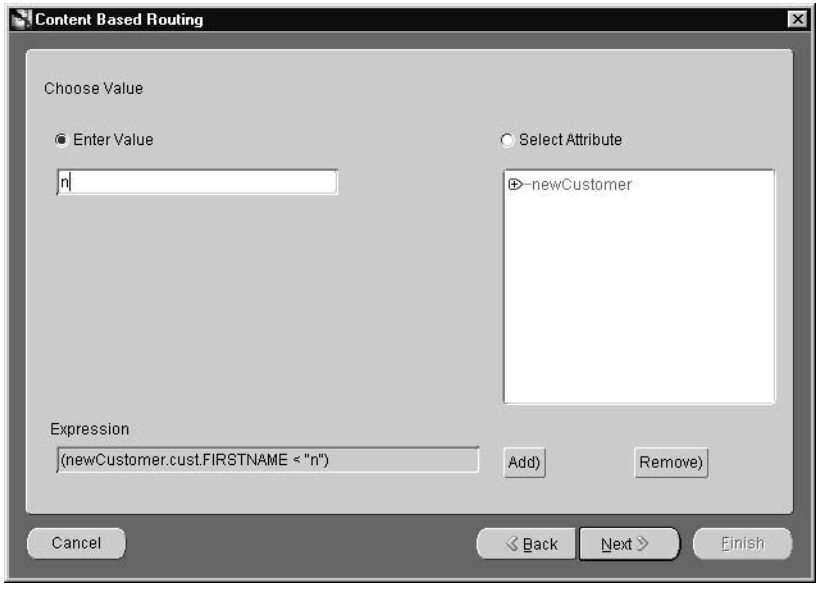

**a.** Choose the source event attribute in the Choose Source box and click Next.

4. **Choose Operator Page**



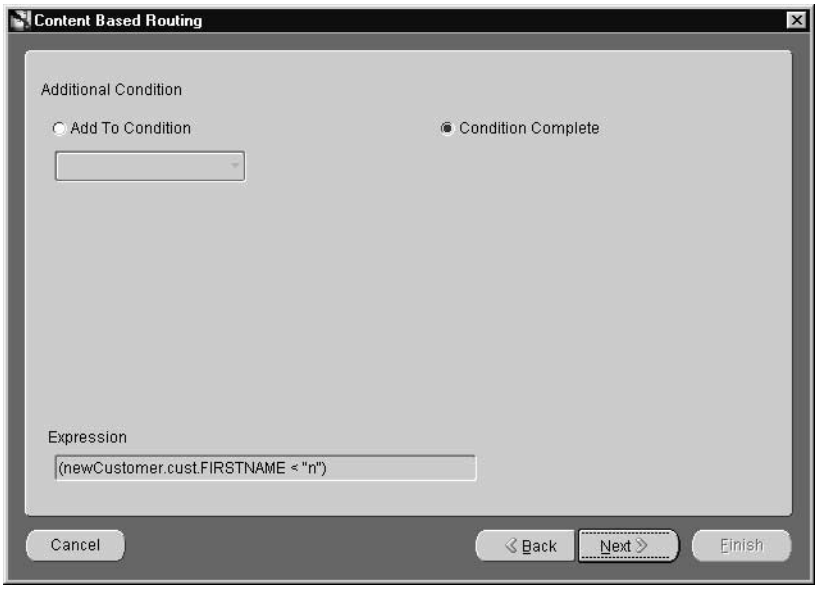

**a.** Select the operator from the drop-down list and click Next.

### 5. Choose Value Page




**a.** Select Enter Value to enter a value in the text field or click Select Attribute to select an attribute from the tree and click Next.

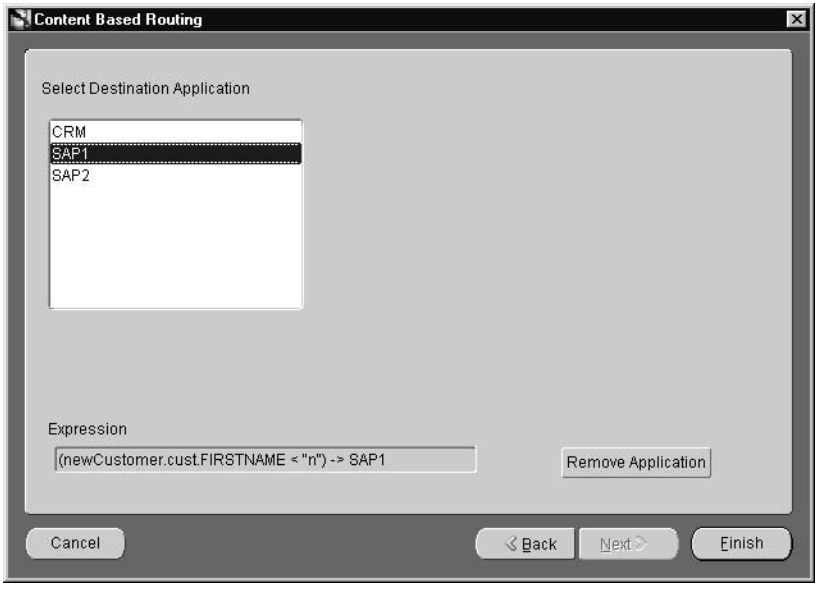
### 6. Additional Condition Page




**a.** Click Add To Condition and select a boolean operator from drop-down dialog to add additional conditions.

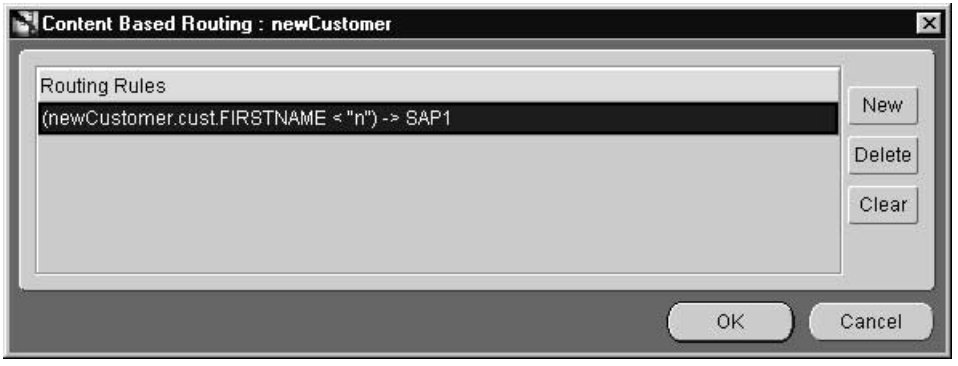
**b.** Click Next. Depending on which radio button is selected, the Choose Source page or Select Destination Application page displays.

7. **Select Destination Application Page**




   a. Select an application from the Select Destination Application box and click Finish.

8. The Content Based Routing Rule is created and displays on the Content Based Routing dialog:



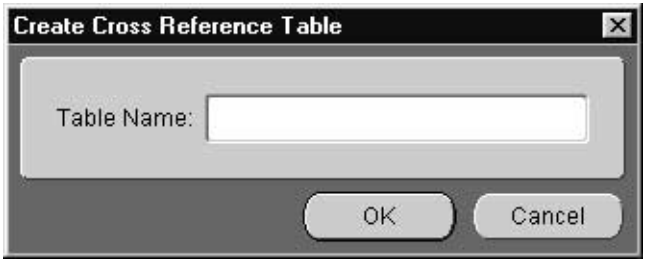

9. Click OK.

### Working with Cross Reference Tables

For more information on cross reference tables, see "Cross Reference Tables" on page 2-15.

To create a cross reference table:

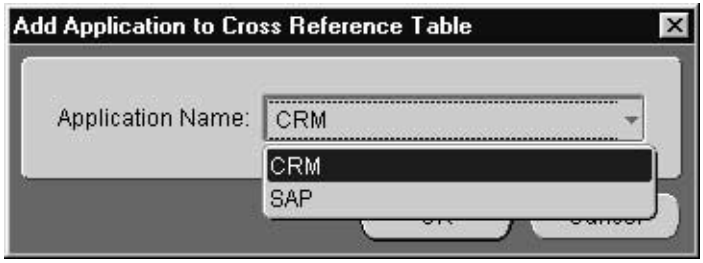
1. From the File menu, click New.
2. From the New pull-down menu, select Cross Reference Tables. The Create X_ Ref Table dialog displays:




3. Enter a name for the cross reference table in the Table Name field.
4. Click OK.

**Adding Applications to Cross Reference Tables** To add applications to the cross reference table:

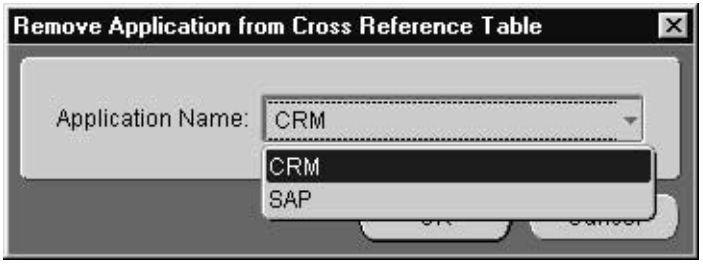
1. In the project tree, select the appropriate cross reference table and right-click it to display the context menu.
2. From the context menu, select Add App. The Add Application to XRef Table dialog displays:



3. From the drop down list, select an application name.
4. Click OK.

**Removing Applications From Cross Reference Tables** To remove applications from the cross reference table:

1. From the project tree, select the appropriate cross reference table and right-click it to display the context menu.
2. From the context menu, select Remove App. The Remove Application from XRef Table dialog displays:

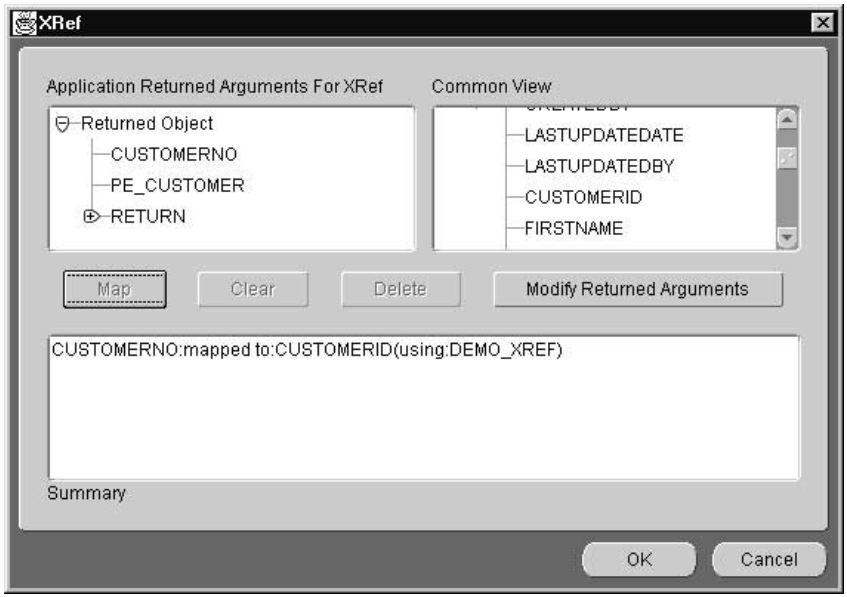



3. From the drop down list, select an application name.
4. Click OK.

**Populating Cross Reference Tables** To populate the cross reference tables, returned arguments must first be defined. Returned arguments are the arguments returned by the subscribe or implement code for populating the cross reference table.

To populate cross reference tables:

1. Click Cross Reference... on the Define Application View page on the Subscribe Wizard. The XRef dialog displays:




   The Application Returned Arguments box displays the returned arguments. This information is initially populated with any OUT arguments from the application view.

2. Click Modify Return Arguments to modify the returned arguments list.

3. Select the corresponding attributes in the Application Returned Arguments and Common View windows and click Map.

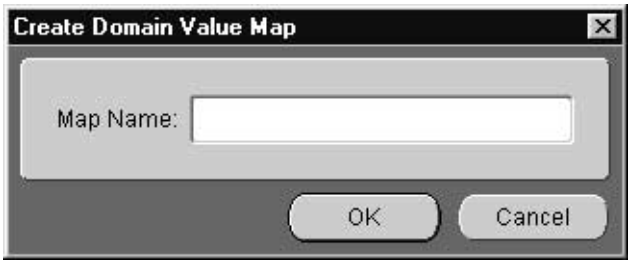
   Specify the Cross Reference Table name to be populated using these attributes' values.

### Working with Domain Value Mappings

For more information on domain value mappings, see "Domain Value Mapping" on page 2-15.

To create a domain value mappings table:

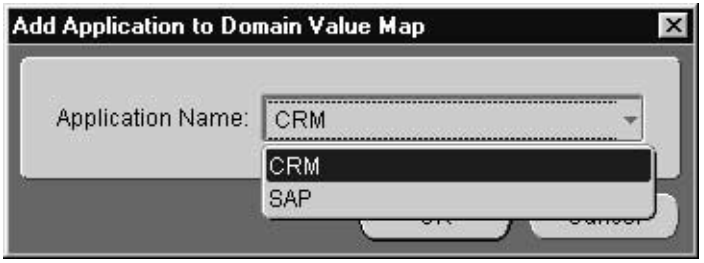
1. From the File menu on the iStudio main menu panel, click New.
2. From the New pull-down menu, select Domain Value Mapping. The Create DVM Table dialog displays:




3. Enter a name for the domain value map in the Map Name field.
4. Click OK.

**Adding Applications to Domain Value Mappings** To add applications to domain value mappings:

1. In the project tree, select a domain value mapping and right-click it to display the context menu.
2. From the context menu, select Add App. The Add Application to Domain Value Map dialog displays:



3. Select an application name from the drop down list.
4. Click OK.

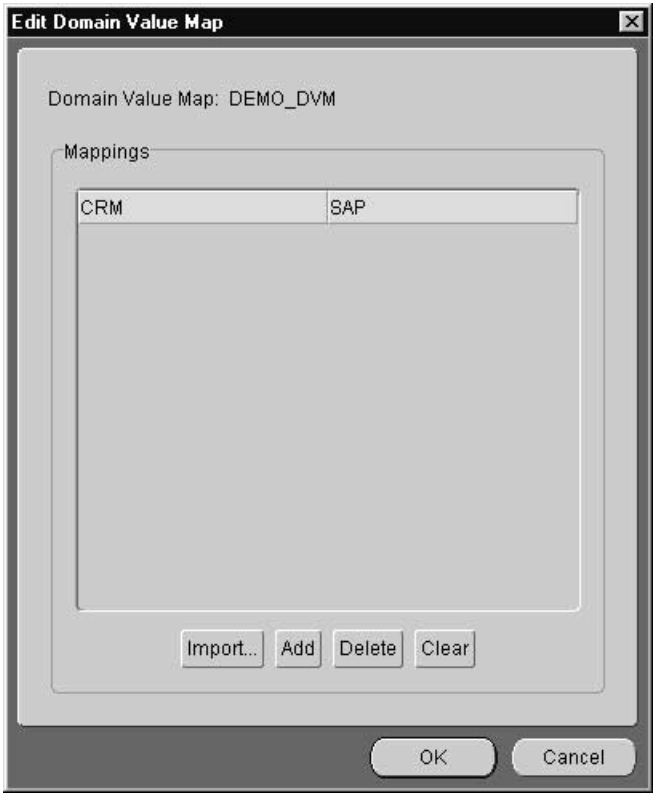
**Removing Applications From Domain Value Mappings** To remove applications from the domain value mappings:

1. In the project tree, select a domain value mapping and right-click it to display the context menu.
2. From the context menu, click Remove App. The Remove Application from Domain Value Mapping dialog displays
3. Select the Application Name to remove from the list.
4. Click OK.

**Modifying Data in Domain Value Mappings** To add data to domain value mappings:

1. In the project tree, select a domain value mapping and right-click it to display the context menu.

2. On the context menu, select Edit Domain Value Map. The Edit Domain Value Map Dialog displays:




3. Enter the mapping values and click Add.
4. Click OK.

**Deleting Domain Value Mappings** To delete a selected domain value mapping:

- Select the domain value mapping to delete and click Delete**.**

**Deleting Domain Value Mapping Tables** To delete the domain value mapping table:

1. Select the domain value mapping table to be deleted and right-click to display the context menu.
2. From the context menu, select Delete.

**Modifying Attribute Mappings** To modify a selected attribute mapping, use the Define Mapping dialog in the Publish Wizard:

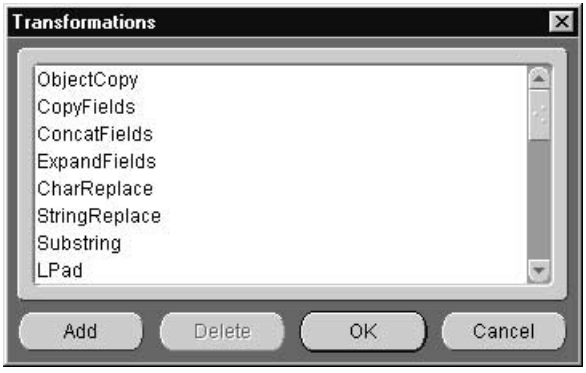
- Select a mapping and click Edit.

**Removing or Clearing Attribute Mappings** To delete a selected attribute mapping, use the Define Mapping dialog in the Publish Wizard:

- Select the mapping to delete and click Remove. To clear all mappings, click Clear.

**Adding Custom Transformations** To create custom transformations:

1. Click Custom Transformations on the Mapping dialog. The Transformations dialog displays:




2. To add user-defined transformations, click Add.
3. Click OK.

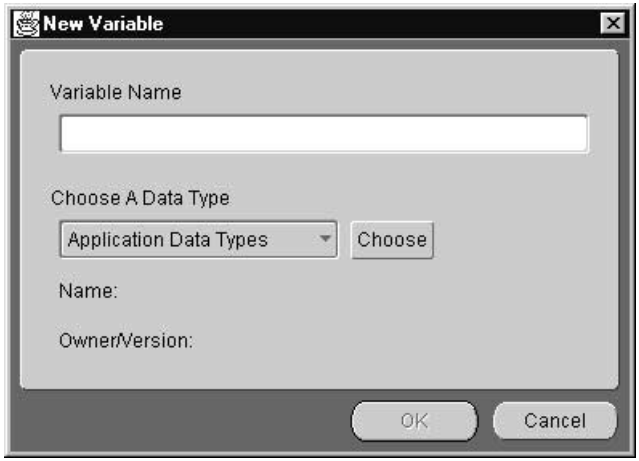
**Deleting Custom Transformations** Using the Transformation dialog, you can delete custom transformations. To delete a selected transformation:

1. Select the transformation to delete and click Delete.
2. Click OK on the Transformation dialog.

**Adding Mapping Variables** To add a mapping variables, use the Mapping dialog.

1. On the Mapping dialog, click Variables. The Variables dialog displays:

2. Click Add to add a variable. The New Variable dialog displays:




3. Enter a name for the variable in the Variable Name field.
4. Select a data type from the drop down list and click Choose. The data type for the variable must be previously defined.
5. Click OK to return to the New Variable dialog.
6. Click OK on the New Variable dialog.

**Deleting a Mapping Variable** To delete a mapping variable, use the Mapping dialog:

1. Select the mapping variable to delete and click Delete.
2. Click OK.

# 3

# Oracle Applications InterConnect and Oracle Workflow

This chapter discusses Oracle Applications InterConnect and Oracle Workflow.

Topics include:

- Oracle Workflow Overview
- Oracle Applications InterConnect Integration with Oracle Workflow
- Using Oracle Workflow with Oracle Applications InterConnect
- Design Business Process

# Oracle Workflow Overview

Oracle Workflow is integrated with Oracle Applications InterConnect. In the context of Oracle Applications InterConnect, Oracle Workflow is used for business process collaborations across two or more applications. A business process collaboration is defined as the conversation between two or more applications in the context of a business process.

In previous versions of Oracle Applications InterConnect, this business process definition was implicit in the messaging definitions through iStudio. Now, Oracle Applications InterConnect leverages the robust design time and runtime Oracle Workflow business process definition and execution support to make these business processes explicit and manageable.

---

**Note:** Knowledge of Oracle Workflow, its tools, and its Business Event System is required to utilize Oracle Applications InterConnect and Oracle Workflow for business process collaboration. For more information on Oracle Workflow, see the *Oracle Workflow Guide*.

---

## Oracle Workflow Solves Common Business Problems

The following are some of the common business problems that can be solved using Oracle Workflow.

### Error Management

If there is a problem in a conversation between two or more applications, the errors arising from this problem can be centrally managed and appropriate remedial actions can be defined. For example, it may be a requirement to keep the data of an order entry system in synch with a backend ERP system. Consider that a new purchase order is created in the order entry system but the ERP system is down at the time the purchase order is created. At a later time, the ERP system comes back up and an attempt is made to create a corresponding new purchase order through messaging using Oracle Applications InterConnect. This attempt fails. To deal with this scenario, the integrator can utilize Oracle Workflow to send a compensating message to the order entry system to undo the creation of the purchase order and notify the user who created the order.

### Human Interaction

In previous versions of Oracle Applications InterConnect, the conversation between two or more applications was based purely on messaging. Now, human interaction can be added to better capture business processes.

In the example above, Oracle Applications InterConnect and Oracle Workflow can be used to model the following for every purchase order that is over $50,000, send a notification to a named approver, and wait for approval. If approved, send the message to the ERP system, otherwise send a message to the order entry system to rollback the order creation.

### Message Junctions

Fan-in and fan-out of messages can be effectively modeled using Oracle Applications InterConnect and Oracle Workflow. Fan-in involves combining two or more messages into one. Fan-out involves splitting one message into two or more.

For an example of fan-in, consider the following. A global organization has a centralized Human Resources ERP application in the United States. Each country has one or more local systems that capture local employee information. If a new employee joins the Japanese branch of this organization, data is entered into a local HR application and a local Benefits application. Each entry launches a message for adding this information to the centralized system. However, the centralized system needs data from both systems combined and will only commit the data if it was entered successfully in both the local systems. Using Oracle Workflow, this process can be modeled so that Oracle Applications InterConnect routes messages from both local systems to Oracle Workflow, Oracle Workflow waits until it receives both messages, combines the data, and launches a single message to be delivered by Oracle Applications InterConnect to the centralized HR system.

### Stateful Routing

Oracle Applications InterConnect provides extensive support for stateless routing through event based and content based routing features. Using Oracle Workflow, stateful routing can be accomplished. In other words, the decision to route can be based on more than the event or the content of the message.

### Composite Services

Using all of the above examples, an internal (organization focused) or external (customer / partner focused) service can be built through a well defined set of business processes involving communication between two or more applications. For example, a brick-and-mortar retail company could provide an on-line

procurement service to their customers. Behind the user interface are several business processes controlling communication across several internal applications to deliver a robust, performant service to the customer.

**Note:** The ability to define explicit business process collaborations is a feature, not a requirement for completing integrations. It is not necessary to utilize Oracle Workflow for integration if the business process definition is simple enough to be implicitly captured in the messaging through the core functionality in iStudio.

# Oracle Applications InterConnect Integration with Oracle Workflow

The following describes how Oracle Applications InterConnect and Oracle Workflow are integrated.

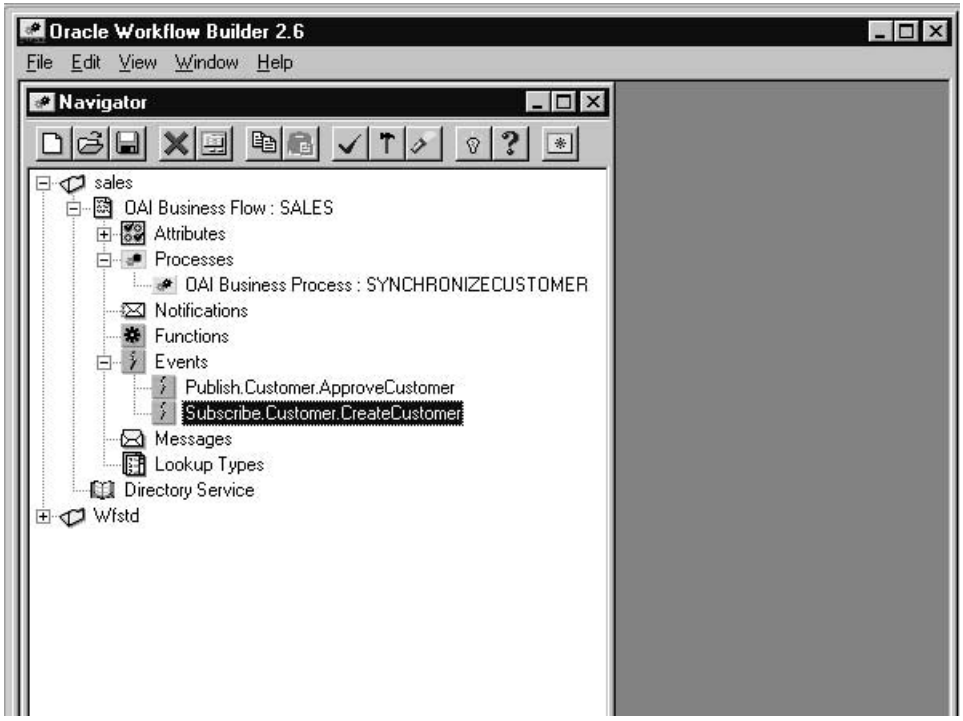
## Design Time Tools

During design time, business process and event definitions in iStudio can be deployed to Oracle Workflow with one click. Consequently, Workflow tools can be launched from within iStudio to graphically create process diagrams in the context of enterprise integration through Oracle Applications InterConnect.

There are two Oracle Workflow tools that can be launched through iStudio:

- Workflow Builder—Use this tool to complete business process definitions defined and deployed through iStudio. This is accomplished by using Workflow Builder to create process diagrams.

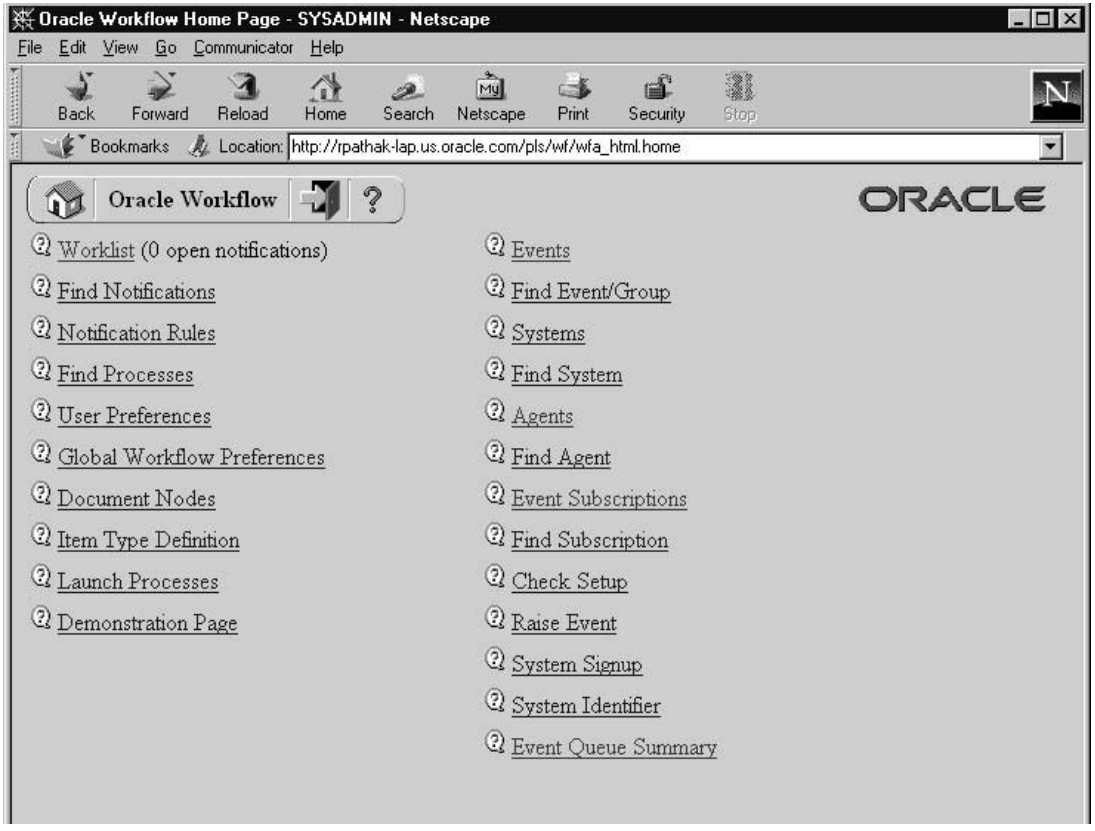
***Figure 3–1 Oracle Workflow Builder***




- Workflow Home Page—Use this tool for centralized access to the web-based features of Oracle Workflow. In the context of Oracle Applications InterConnect, the Business Event System management and administration is the most important feature exposed through this home page.

  For more information on the Business Event System, see the *Oracle Workflow Guide*.

***Figure 3–2 Oracle Workflow Home Page***

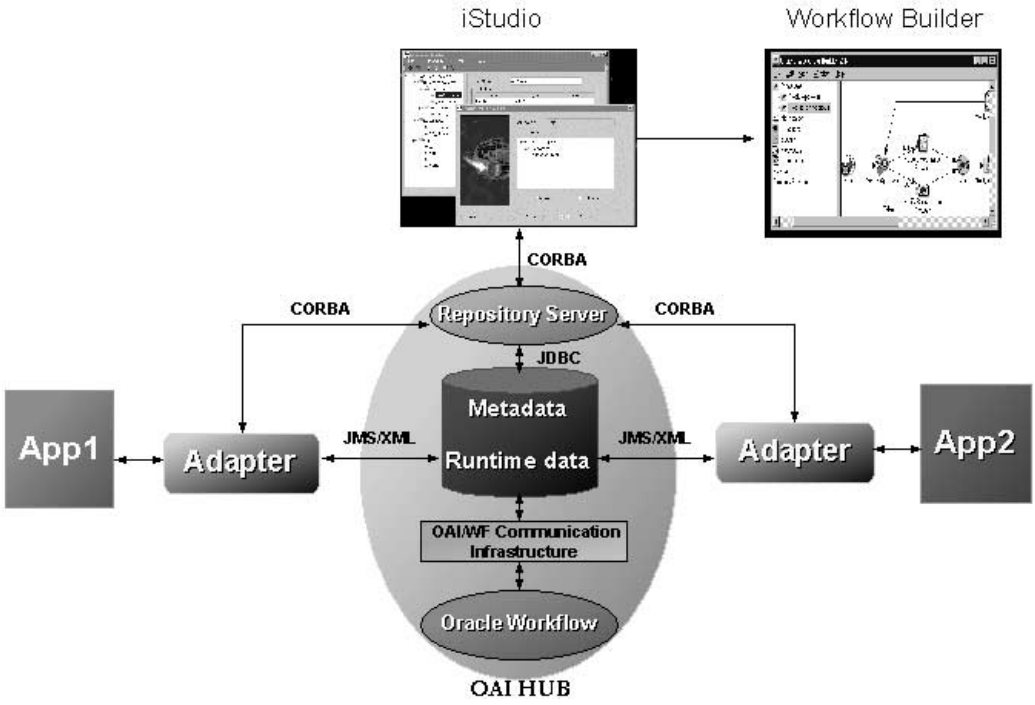



## Runtime

At runtime, Oracle Applications InterConnect integrates with the Business Event System of Oracle Workflow. The Business Event System is an application service that uses the Advanced Queueing infrastructure to communicate business events between systems. Oracle Applications InterConnect registers itself as an external system in Business Event System so the following conditions exist:

- Messages can flow from applications through Oracle Applications InterConnect, in the common view format, to the Business Event System to either trigger or continue Workflow business processes as defined using iStudio (Business Processes) and described using Workflow Builder (Process Diagrams).

- Messages can flow from the Business Event System to Oracle Applications InterConnect in the common view format, to applications to either continue or end Workflow business processes.




At runtime, Oracle Workflow is integrated with Oracle Applications InterConnect at the hub. Messages are passed back and forth between Oracle Applications InterConnect and the Business Event System of Oracle Workflow via Advanced Queues. The Oracle Applications InterConnect Workflow Communication Infrastructure facilitates this communication.

At design time, to keep the integration methodology consistent, iStudio reuses the messaging paradigms of publish/subscribe and request/reply to specify communication between Oracle Applications InterConnect and Oracle Workflow. So, for messages inbound into Workflow, the iStudio user can specify, in the context of a business process, which events Workflow is subscribing to and which procedures Workflow is implementing. For, outbound messages, you can specify events that Workflow can publish and procedures that it can invoke.

For more information on how to set up Oracle Applications InterConnect and Oracle Workflow at design time, see "Using Oracle Workflow with Oracle Applications InterConnect" on page 3-8.

# Using Oracle Workflow with Oracle Applications InterConnect

This section describes using Oracle Applications InterConnect with Oracle Workflow. There are three broad steps.

## Install Oracle Workflow Components

To install the following Oracle Workflow components, see the *Oracle Applications InterConnect Installation Guide*.

- On the iStudio machine, install the following:
  - Workflow Builder
- On the hub machine install the following:
  - Oracle Applications InterConnect Workflow Communication Infrastructure
  - Oracle HTTPS Server
  - Oracle Workflow Server

In addition, follow the Workflow-related post installation steps as described in the *Oracle Applications InterConnect Installation Guide.*

## Design Business Process

- Design process bundles using iStudio.
- Deploy process bundles from iStudio to a `.wft` file.
- Complete process diagrams in Workflow Builder by launching Oracle Workflow Builder from iStudio and using the deployed `.wft` file.

## Deploy Business Processes for Runtime

- Deploy events to the Business Event System from iStudio.
- Deploy a process diagram from a file to the database using Workflow Builder.

# Design Business Process

The following concepts discuss how iStudio and Oracle Workflow work together in Oracle Applications InterConnect. In addition, these topics discuss how to use iStudio and Oracle Workflow step by step during design time for business process collaborations across applications.

## Process Bundle

A process bundle is a set of logically related business processes. This maps one-to-one with a workflow item. For more information on business processes, see "Business Process".

## Business Process

A business process is a set of Oracle Applications InterConnect common view events and / or procedures that must be routed to and from Oracle Workflow in one workflow business process. These events and procedures manifest themselves as workflow business events and can be used to define a process diagram in Oracle Workflow Builder. This maps one-to-one with a workflow business process.

## Activity

Activities in iStudio allows the user to define the common view events and procedures that must be a part of a workflow business process. The following lists the types of activities in iStudio:

- Publish Event—Workflow publishes an Oracle Applications InterConnect common view event. At deployment time, a business event corresponding to the common view event is created in the Business Event System.
- Subscribe Event—Workflow subscribes to an Oracle Applications InterConnect common view event. At deployment time, a business event corresponding to the common view event is created in the Business Event System.
- Invoke Procedure—Workflow invokes an Oracle Applications InterConnect common view procedure. At deployment time, two business events corresponding to the common view procedure are created in the Business Event System: one event for sending the request and one for receiving the reply.
- Implement Procedure—Workflow implements an Oracle Applications InterConnect common view procedure. At deployment time, two business events corresponding to the common view procedure are created in the

Business Event System: one event for receiving the request and one for sending the reply.

The following table describes how iStudio and Oracle Workflow concepts are mapped.

| iStudio Concept | Oracle Workflow Concept | Mapping |
|---|---|---|
| Process Bundle | Item | One-to-one. |
| Business Process | Business Process | One-to-one. |
| Common View Event | Business Event | One-to-one.[1] |
| Common View Procedure | Business Event | Two business events per procedure. |
| Publish Activity | Send Event Activity | One-to-one. |
| Subscribe Activity | Receive Event Activity | One-to-one. |
| Invoke Activity | Send Event Activity (for the request) | |
| | Receive Event Activity (for the reply) | |
| Implement Activity | Receive Event Activity (for the request) | |
| | Send Event Activity (for the reply) | |

[1] Only for all events that are part of a business process in iStudio. Events that are part of the common view but not part of a business process are not instantiated as Oracle Workflow business events. All common view events need not be part of business processes. In other words, depending on the integration, some common view events could be exchanged directly between applications without involving Oracle Workflow using the core functionality of Oracle Applications InterConnect. Other events may need to be part of an explicit business process. It is the latter set of events that become business events in Oracle Workflow. The same is true for common view procedures.

## Creating a Process Bundle

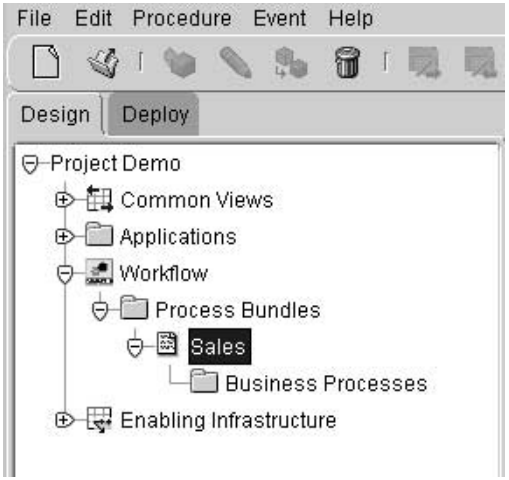
To create a process bundle using iStudio:

1. From the project tree, click Workflow and expand the subtree.
2. Right-click on Process Bundles and select New.

The Create Process Bundle dialog displays:


3. Enter the name of the process bundle in the Process Bundle Name field and click OK.

   A new process bundle is created.

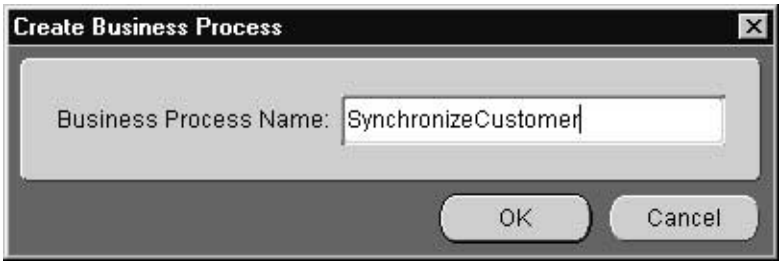



## Creating a Business Process

To create a business process:

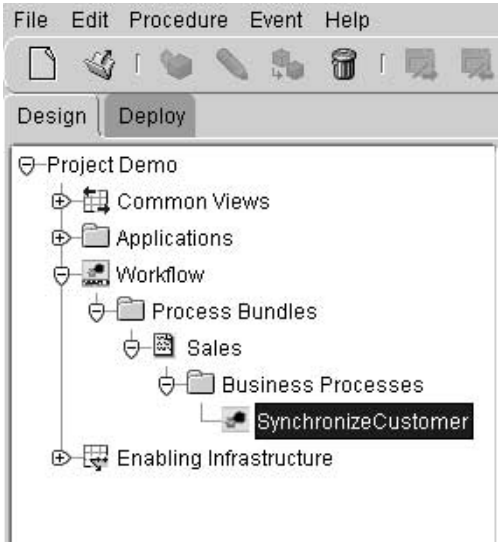
1. From the project tree, expand the process bundle for which the business process is to be created.
2. Right-click on Business Processes and select New.

The Create Business Process dialog displays:




3. Enter a name for the business process in the Business Process Name field and click OK.

   A new business process is created.



## Populating a Business Process with Activities

To populate a business process with activities:

1. From the project tree, select the business process to populate.
2. Right-click the business process and in the context menu, select the activity to be part of the business process. Choose from the following activities:
   - New Publish Activity—Workflow sends a message to Oracle Applications InterConnect in the context of the business process.
   - New Subscribe Activity—Workflow receives a message from Oracle Applications InterConnect.[1]
   - New Invoke Activity—Workflow sends a request message to Oracle Applications InterConnect and receives a reply.
   - New Implement Activity—Workflow receives a request from Oracle Applications InterConnect and sends a reply.

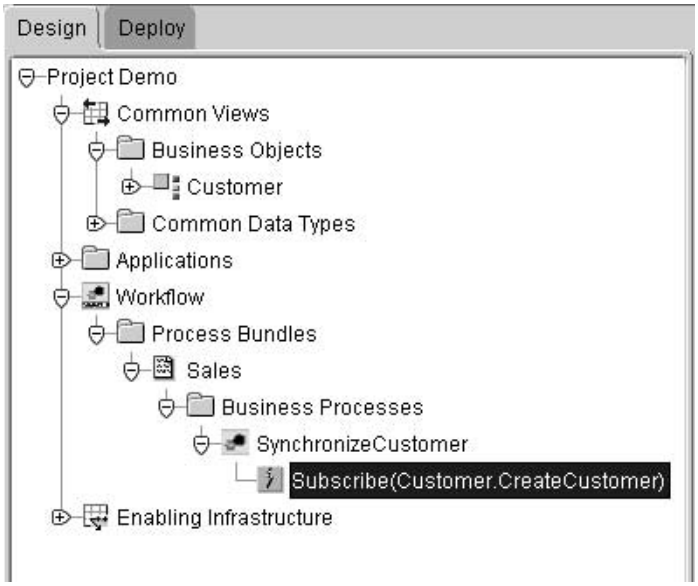
   The Subscribe Activity Wizard displays.

[1] This example is used for explaining the steps. The steps are similar regardless of the selection.

3. Select an event for the activity.


4. Click Finish.

A new activity has been added to the business process.




Repeat these steps for adding other activities to the process.

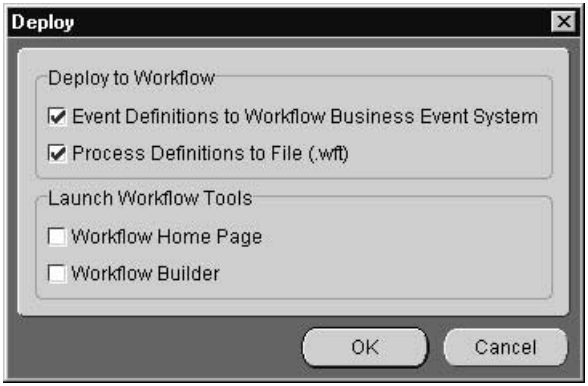
**Note:** When you create multiple activities under a business process, the list of activities is unordered, i.e. the order in which the activities are added is not important. The order can be later decided in Oracle Workflow Builder through a process diagram.

## Deploying to Oracle Workflow

After populating business processes with activities, this information must be deployed to Oracle Workflow to graphically model a business process. To deploy this information to Oracle Workflow:

1. On the Deploy tab in iStudio, right click on Workflow and select Deploy.

The Deploy dialog displays:




2. There are two sets of information that need to be deployed. This can be done independently of each other or together:

   - Workflow Business Events—Business Events need to be created in the Business Event System of Oracle Workflow. This is a requirement for runtime only. Using Workflow Builder, users can decide to not deploy these until all design time work including modeling the business process, is complete.

     **Note:** IStudio checks if an event is already deployed before deploying it. Therefore, users can safely decide to re-deploy all events at anytime. However, deploying events after all the design time work has been completed saves you the effort of redeploying events repeatedly.

     To check if events have been deployed, launch the Oracle Workflow Home page. For more information on the Oracle Workflow Home page, see "Launching the Oracle Workflow Home Page" on page 3-19.

   - Workflow Process Definitions through `.wft` file generation—Information about business processes captured in iStudio provides a foundation for building process diagrams in Oracle Workflow Builder. Deploying process definitions is required for design time.

**Note:** When deploying process definitions, iStudio prompts for a filename. If an existing file is specified, iStudio will **overwrite** the file. Therefore, if there are existing process definitions in a file modified using Workflow Builder, do not select that filename as the target, otherwise all modifications made will be lost.

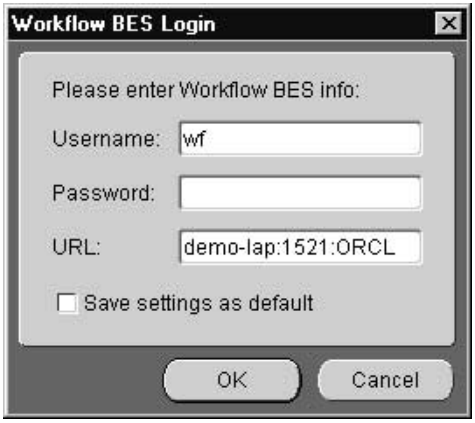
By default both choices are selected. The dialog also allows the following to be automatically launched:

- Workflow Builder—Defines business process diagrams.
- Workflow Home Page—Verifies Business Event deployment.

By default these choices are unselected. Choose to launch these tools with deployment or complete this task at a later time on the Design tab. For more information on launching these tools, see "Launching Oracle Workflow Tools" on page 3-19.

3. Select the desired choices and click OK.

   If you selected deploying event definitions to the Oracle Workflow Business Event System, the following dialog displays:




4. Enter the required information based on the selections made during Oracle Workflow installation and click OK.

If Deploying Process Definitions to a `.wft` file was selected, a file dialog displays:


5. Enter the name and location of the file to create and click OK.

---

**Note:** When deploying process definitions, iStudio prompts for a filename. If an existing file is specified, iStudio will **overwrite** the file. Therefore, if there are existing process definitions in a file modified using Workflow Builder, do not select that filename as the target, otherwise all modifications made will be lost.

---

## Launching Oracle Workflow Tools

The following topics discuss how to launch Oracle Workflow tools in iStudio.

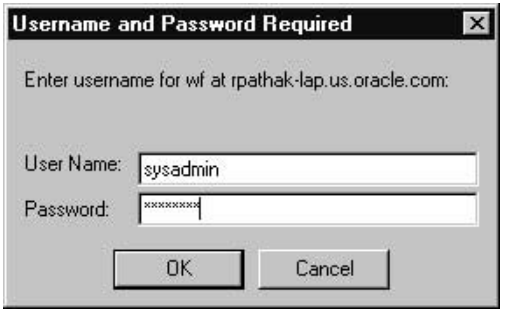
### Launching the Oracle Workflow Home Page

To launch the Oracle Workflow Home page:

1. On the Design tab in iStudio, right click on Workflow and select Launch WF Home Page. The Workflow Home Page dialog displays:


2. Make sure the URL is correct and click on OK.

   The Username and Password Required dialog displays:




3. Enter the login information for Oracle Workflow Home Page and click OK.

   The Workflow Home page is launched using the default browser.

### Launching Oracle Workflow Builder

To launch Oracle Workflow Builder:

1. On the Design tab in iStudio, right click on the process bundle to view in Workflow Builder and select Launch Workflow Builder. The Deploy dialog displays:

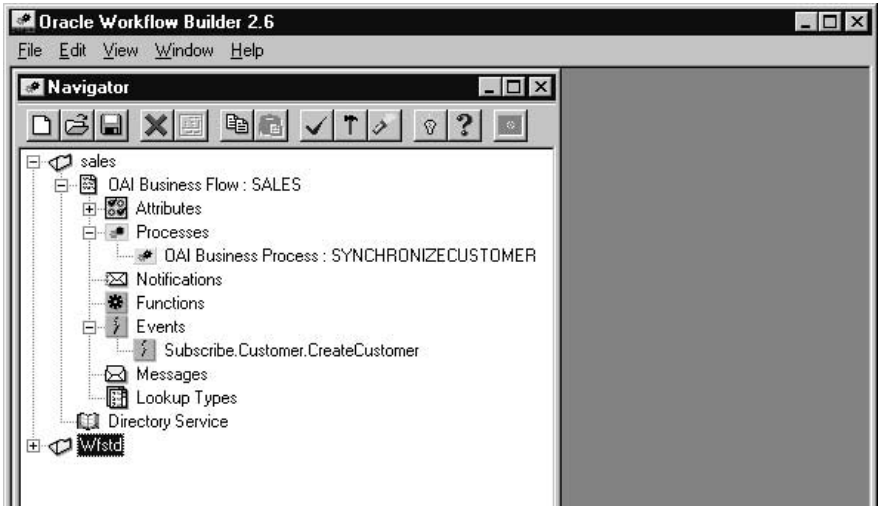

2. In the Deploy dialog, select an existing `.wft` file name to load into Workflow Builder.

---

**Note:** The assumption is that a process definition has already been deployed to a file.

---

Workflow Builder is launched depending on which process definition file selected.




**Note:** To launch Workflow Builder outside of a specific Oracle Applications InterConnect process bundle, right-click on Workflow and select Launch Workflow Builder.

## Modifying Existing Oracle Workflow Processes

When modifying existing Oracle Workflow processes, do not add, modify, or remove Oracle Applications InterConnect event activities directly in Workflow Builder. Always make all event-related process changes in iStudio, redeploy to the file, and import in Workflow Builder.

For example, assume the following steps were completed to create an integration-related Oracle Workflow business process:

1. Create a process bundle in iStudio and create business processes with some activities.
2. Deploy to the `my_process_bundle.wft` file.
3. Import the file into Workflow Builder.

4. Make **non-event** related modifications to the process in Workflow Builder, such as adding other activities like notifications and decision functions to complete the business process.
5. Save the modified process to `my_process_bundle.wft`.

Now, some event related modifications to the business process need to be made. For example, two new events need to be added to the business process. The following steps complete this task:

6. Using iStudio, make the additions to the particular business process.
7. Deploy to a different file such as `changes_to_my_process_bundle.wft`. Do not deploy to `my_process_bundle.wft` because any non-event-related modifications made through Workflow Builder will be lost.
8. Launch Workflow Builder and import both `my_process_bundle.wft` and `changes_to_my_process_bundle.wft`.
9. Drag the required modifications from the process representing `changes_to_my_process_bundle.wft` to the process representing `my_process_bundle.wft`.
10. Save the modified process to `my_process_bundle.wft`.

    The `my_process_bundle.wft` file now contains the updated process definition with both the event and the non-event modifications.

# 4

# Runtime System Concepts and Compents

This chapter describes the runtime concepts, components and processes of Oracle Applications InterConnect.

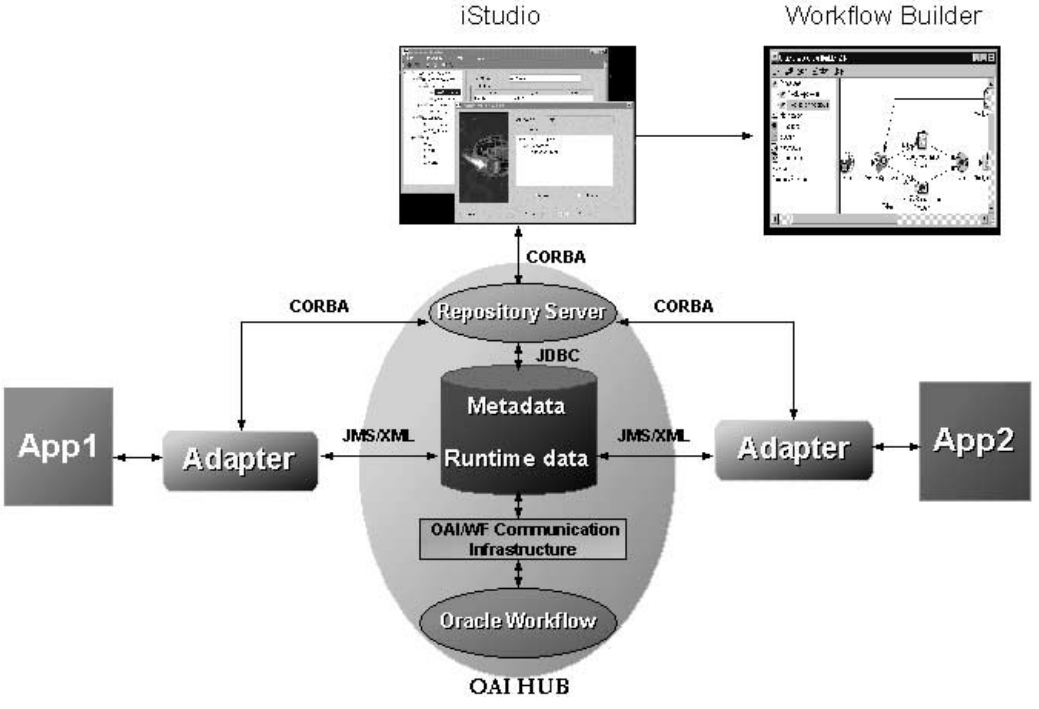
This chapter discusses the following topics:

- Integration Architecture
- Features
- Components

# Integration Architecture

Oracle Applications InterConnect runtime is an event-based distributed messaging system. An event is any action that initiates communication through messaging between two or more applications integrated through Oracle Applications InterConnect. The messaging system can be deployed both within an enterprise or across enterprise boundaries.

The runtime enables inter-application communication through hub and spoke integration. This methodology keeps the applications decoupled from each other by integrating them to a central hub rather than to each other directly. The applications are at the spokes of this arrangement and are unaware of the applications they are integrating with. To them, the target of a message (or the source) is the hub. Since each application integrates with the hub, translation of data between the application and hub (in either direction) is sufficient to integrate two or more applications.



# Features

The Oracle Applications InterConnect runtime features are as follows:

- Messaging Paradigms
- Message Delivery
- Message Retention
- Routing Support
- Error Management
- Scalability and Load Balancing

## Messaging Paradigms

Oracle Applications InterConnect runtime supports three major messaging paradigms:

- Publish / Subscribe
- Request / Reply (synchronous and asynchronous)
- Point-to-Point

These paradigms can be configured on a per integration point basis. Refer to Chapter 2, "Design Time Concepts and iStudio" for detailed explanation on what these paradigms are and how to configure Oracle Applications InterConnect to utilize a specific paradigm for an integration point.

## Message Delivery

**Guaranteed delivery** All messages are guaranteed to be delivered from the source application(s) to the destination application(s).

**Exactly once delivery** The messages are neither lost nor duplicated. The destination application(s) will receive each sent message exactly once.

**In order delivery** The messages are delivered in the exact same order as they were sent.

## Message Retention

Messages remain in the runtime system until they are delivered. Advanced Queues in the hub provide the message retention. The messages are deleted when each application that is scheduled to receive a specific message has received that message. For auditing purposes, you can configure the system to retain all messages even after they have been delivered successfully to each application.

## Routing Support

The current version of Oracle Applications InterConnect has significant improvements over the previous releases for configuring your routing needs. Routing is a function of the Advanced Queues in the hub. By default, there is only one multiconsumer Advanced Queue configured to be the persistent store for all messages for all applications—`oai_hub_queue`. This will handle all your standard as well as content based routing needs. Morever, this queue is created automatically when you install the repository in the hub. The only reason to change this configuration is if Advanced Queues becomes a performance bottleneck. For most scenarios, this is unlikely because most of the message processing is done in the adapters, not in the hub. For information on how to address performance bottlenecks in the adapters, see "Scalability and Load Balancing" on page 4-5.

### Content-Based Routing

Content-based routing allows you to route messages to specific destination applications based on message content. For example, an electronic funds transaction settlement application is designed to transmit bank transactions with a specific bank code to identify the destination bank system. When the electronic funds transfer application publishes each message at runtime, the Oracle Application InterConnect runtime component determines the BankCode value based on objects stored in the repository, and routes the message to the appropriate recipient system.

## Error Management

This release has significant improvements to deal with error conditions in your integration environment:

- Central Logging—If there is an error in the system, it is logged centrally in the repository.
- Central Monitoring—These errors can be monitored through the runtime management console.

### Resubmission

You can resubmit errored-out messages again into your integration environment for processing after modifying them (if required) using the runtime management console.

### Tracing

You can modify the `.ini` files of adapters to turn up the tracing level to troubleshoot errors. You can view the tracing logs by opening up log files through the runtime management console.

### Tracking

Messages can be tracked by specifying tracking fields using iStudio. The runtime system checkpoints state at certain pre-defined points so that you can monitor where the message is currently in the integration environment. This tracking capability can be utilized through the runtime management console.

## Scalability and Load Balancing

At runtime, it is possible that the adapter attached to a particular application becomes a performance bottleneck. You can detect this by monitoring the message througput information through the runtime console. For more information, see Chapter 5, "Runtime Management".

Oracle Applications InterConnect addresses adapter scalability through a well-defined methodology:

Multiple adapters can be attached to one application to share the message load. This can be done in several ways depending upon the needs of your integration environment. For example, Application A publishes three different kinds of events—EventA, EventB, and EventC. Three potential scenarios should be examined to determine how one or more adapters should be attached to the application to meet performance objectives:

### Scenario 1

- Requirement

  The order in which the messages are sent by Application A must be adhered to strictly for the life of all messages. For example, if Application A publishes messages in a specific order, they must be received by the subscribing applications in the exact same order across all the different event types.

- Solution

  In this case, you cannot add more than one adapter to Application A for load balancing.

### Scenario 2

- Requirement

  The order in which messages are sent by Application A must be adhered to but not across different event types. For example, Application A publishes the following messages in order: `M1_EventA`, `M2_EventB`, `M3_EventA`. `M1` and `M3` must be ordered with respect to each other because they correspond to the same event type. However, `M2` has no ordering restrictions with respect to `M1` and `M3`. In addition, `EventA` messages are transformation / size / computation heavy and `EventB` and `EventC` messages are very light.

- Solution

  In this case, you can create message partitions in the Message Capability Matrix. Refer to Chapter 2, "Design Time Concepts and iStudio" for more details on how to configure the Message Capability Matrix. Partition1 can process `EventA` messages, and Partition2 can process `EventB` and `EventC` messages. When you install the adapters, you specify not only the application it is attached to but also the partition it uses. These message partitions can be used to effectively load balance message processing.

### Scenario 3

- Requirement

  There is no message order dependency, even within the same event type.

- Solution

  Since there are no ordering restrictions, two approaches for load balancing can be employed:

  - No message partitions are created. One or more adapters are added utilizing the entire Message Capability Matrix. This means that at runtime any one of the adapters would be available to receive any message, though only one of them would actually receive the message. Which adapter receives the message will be determined solely by which adapter is the first to request the next message for processing.

- Message Partitions can be created based on projections of the number of messages for a particular event type. For example, if there will be three times as many `EventA` messages than `EventB` or `EventC` messages, you could create two partitions—one for handling `EventA` messages, and the other for handling the other two event types.

# Components

There are four major components in the runtime system:

- Adapters
- Repository
- Advanced Queues
- Workflow

## Adapters

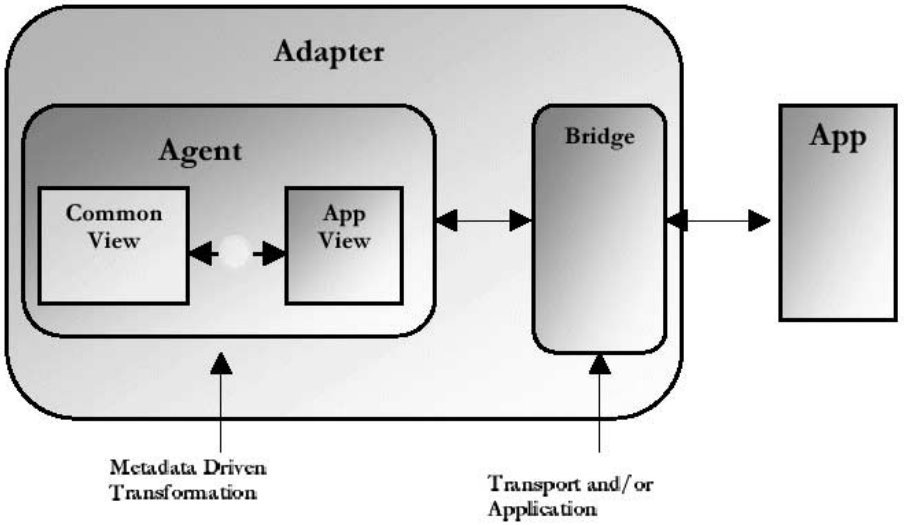
Prepackaged adapters help re-purpose applications at runtime to participate in the integration without any programming effort.

### Agent and Bridge Combination

Adapters are the run-time component for Oracle Applications InterConnect. Adapters have the following responsibilities:

- Application Connectivity—Connect to applications to transfer data between the application and Oracle Applications InterConnect. The logical sub-component within an adapter handles this responsibility is called a bridge. This is the protocol / application-specific piece of the adapter that knows how to communicate with the application. For example, the database adapter is capable of connecting to an Oracle database using JDBC and calling SQL APIs. This sub-component does not know which APIs to call, only how to call them.
- Transformations—Transform data from the application view to common view and vice versa as dictated by the repository metadata. In general, adapters are responsible for carrying out all the runtime instructions captured through iStudio as metadata in the repository. Transformations are an important subset of these instructions. The logical sub-component within an adapter that handles the runtime instructions is called an agent. This is the generic runtime engine in the adapter that is independent of the application to which the adapter connects. It focuses on the integration scenario based on the integration

metadata in the repository. There is no integration logic coded into the adapter itself; all integration logic is stored in the repository. The repository contains the metadata that drives this sub-component. In the database adapter example, this is the sub-component that knows which SQL API's to call, but not how to call them. All adapters have the same agent code. It is the difference in metadata that each adapter receives from the repository that controls and differentiates the behavior of each adapter.




Adapters can be configured to cache the metadata at runtime to address performance needs. There are three settings for caching metadata:

- No Caching—For each message, the adapter will query the repository for metadata. This setting is recommended for an early or unstable integration development environment.
- Demand Caching—The adapter will query the repository only once for each message type and then cache that information. For subsequent messages of the same type, it will use the information from the cache. This setting is recommended for a stable integration development environment.
- Full Caching—At start-up time, the adapter will cache all its relevant metadata. This setting is recommended for a production environment.

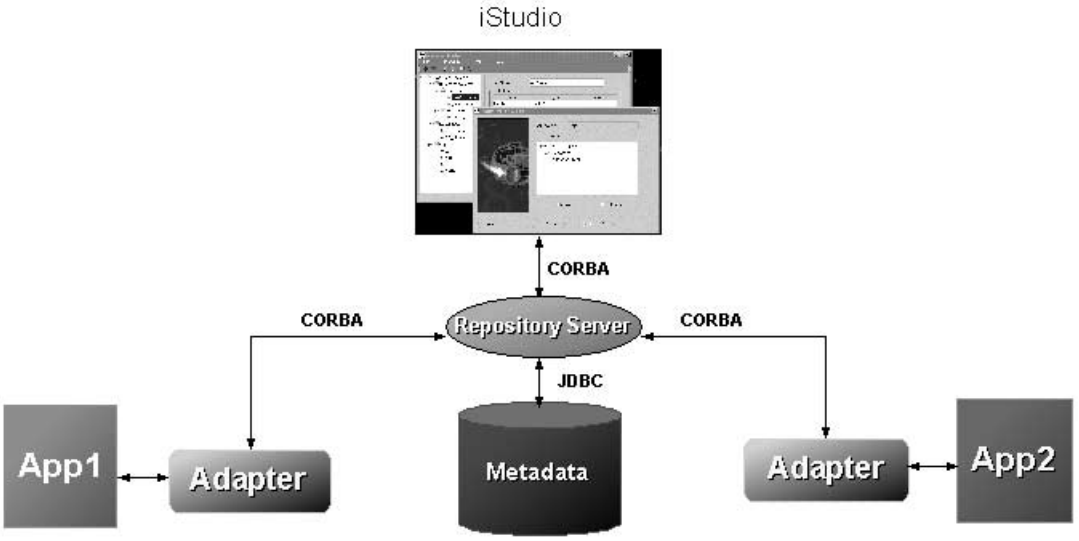
The following adapters are available with Oracle Applications InterConnect:

- Oracle Database Adapter—Allows you to communicate to an Oracle database using JDBC. The database adapter is certified against Oracle database version 7.x to 8.1.7.
- Oracle Advanced Queue Adapter—Allows you to communicate to an Advanced Queue with Raw, XML, and ADT payloads. This adapter can work on Oracle database version 8.1.6 and 8.1.7.
- HTTP/S Adapter—Allows you to communicate across firewalls with an HTTP/S server using XML payloads.
- SAP Adapter—Allows you to communicate to as SAP system using BAPIs, ABAP modules, and Idocs inbound into SAP and Idocs for outbound. This adapter is SAP certified and can work with all SAP versions 3.1H onwards.

## Repository

The repository consists of two components:

- Repository Server—A Java application that runs outside the database. It provides CORBA services to create/modify/delete metadata at design time using iStudio and query during runtime using adapters. Both adapters and iStudio act as CORBA clients to communicate with the repository server.
- Repository Database—The repository server stores metadata in database tables. The server communicates to the database using JDBC.




Adapters have the ability to cache metadata. If the repository metadata is modified after adapters have cached metadata, those adapters are automatically notified by the repository server to purge their caches and re-query the new metadata.

## Advanced Queues

Advanced Queues provide the messaging backbone for Oracle Applications InterConnect in the hub. In addition to being the store and forward unit, they provide message retention, auditing, tracking, and guaranteed delivery of messages. For more information on Advanced Queues, see the *Oracle Database Application Developer's Guide*

## Workflow

Oracle Workflow facilitates integration at the business process level through its Business Event System. Oracle Applications InterConnect and Oracle Workflow are integrated to leverage this facility for business process collaborations across applications.

# 5

# Runtime Management

This chapter describes how to manage Oracle Applications InterConnect components using the Oracle Enterprise Manager Console.

This chapter discusses the following topics:

- Introduction to Runtime Management
- Features

# Introduction to Runtime Management

The Runtime Management Console is an Oracle Enterprise Manager based tool that allows you to manage your integration components at runtime. The Oracle Applications InterConnect components that can be managed through the Console are as follows:

- Adapters
- Repository Server

To be able to manage your integration components, you must

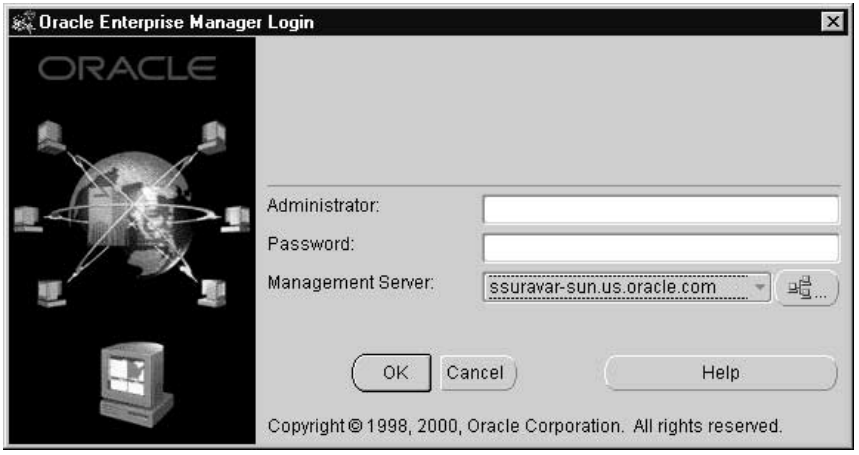
- Install the Enterprise Manager Server on your hub machine (instructions in the install guide). When the management server is started it detects the repository in the hub and all associated adapters.
- Install the Enterprise Manager Console on the machine you wish to manage your integration components from (instructions in the install guide).

For more information on installation, see the *Oracle Applications InterConnect Installation Guide*.

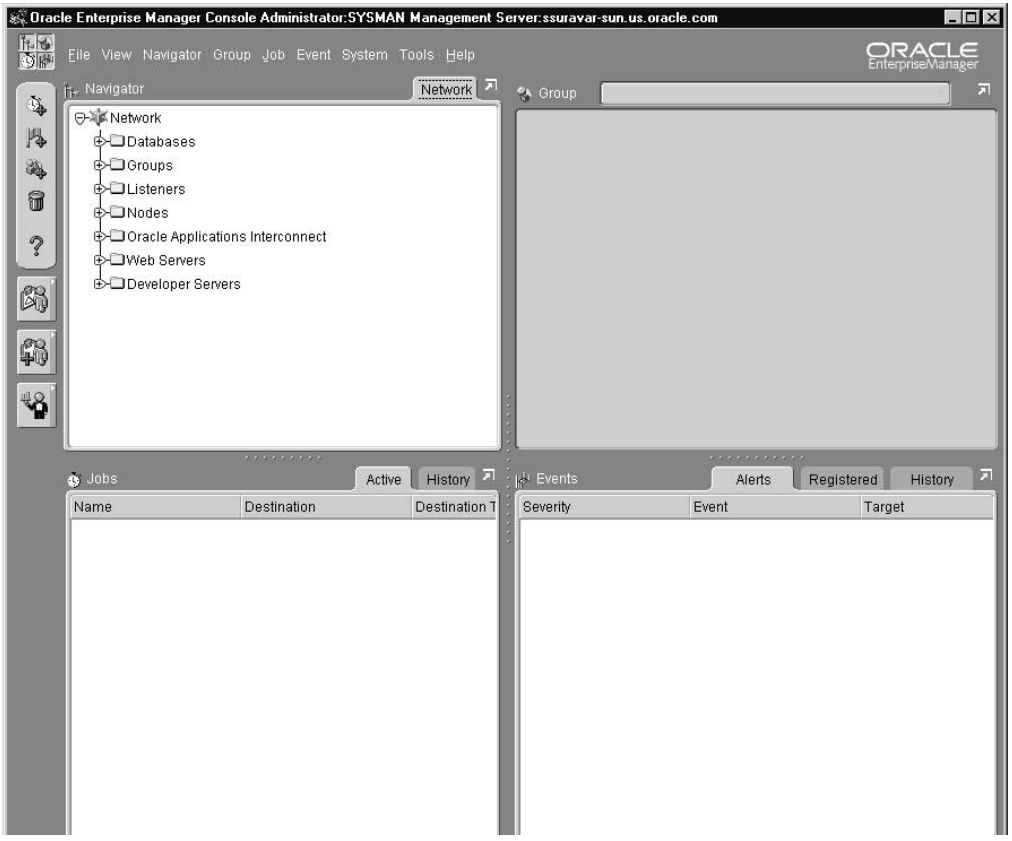
## Starting the Enterprise Manager Console

To start the Enterprise Manager console:

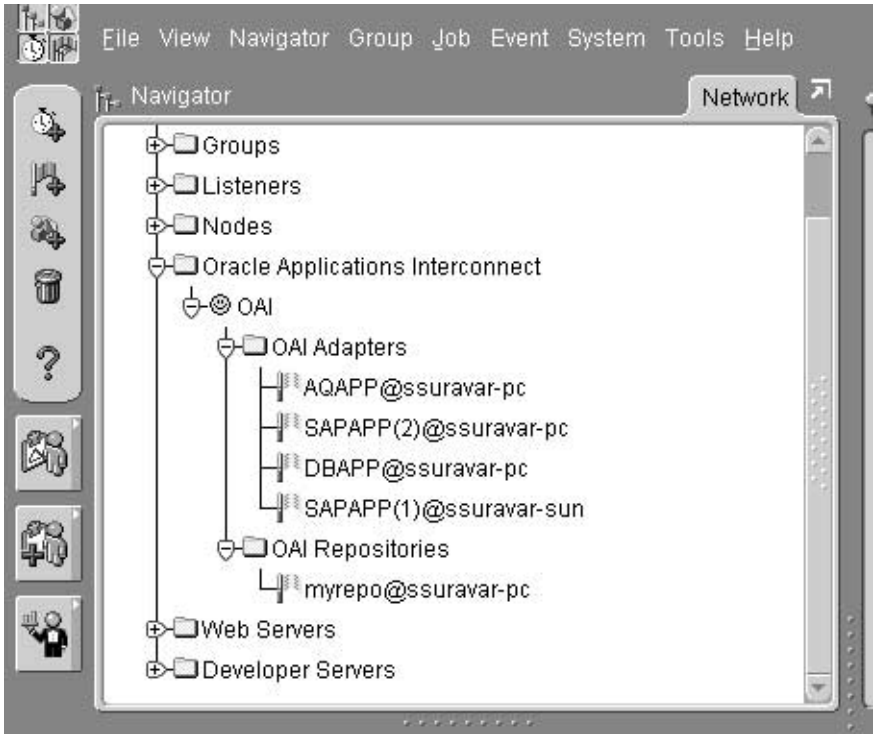
1. Start Enterprise Manager Console. The following dialog displays:



2. Enter the management server you want to connect to, the username, and the password. Click OK. The console comes up:




If you expand the Oracle Applications InterConnect subtree, you see a list of adapters and the repository associated with the particular Oracle Applications InterConnect hub.




To utilize any of the features, right click on the adapter or repository that you would like to manage.

# Features

The following are runtime features for adapters and the repository.

## Common Features for Adapters and the Repository

The following features are provided for both adapters and repository.

- Get State—Shows the component's state. Red indicates the component is stopped. Green indicates the component is running. Yellow indicates the component state is unknown.
- Start—Start the component.
- Stop—Stop the component.

- Alert On Shutdown—Allows you to send a page or email if that particular component goes down. Refer to the *Oracle Enterprise Manager Guide* for setting up the alert.
- Get Ini File—Allows you to browse and modify the `.ini` file for that particular component. If you make changes, the tool prompts you to restart the component. You must restart the component for the changes to take effect.
- Get Log File—Allows you to browse the log files for a particular component. Repositories have one log file each. Adapters have multiple log files. This feature allows you to select the log file you want to look at.
- Get Alive Time—Gets the duration that the component has been up for since it was last started.
- Remove—Removes the component from the list of manageable components for this session of the Oracle Enterprise Manager Console.

## Repository Specific Features

- Get Repo Sessions—Gets a list of all components connected to the repository -- adapters and iStudio.

## Adapter Specific Features

- Get Throughput—Monitor the adapter throughput in terms of messages/second. The steps to get throughput information are as follows:

  1. Right-click on the adapter that you would like to get throughput information for. Select Get Throughput on the context menu.
  2. On the dialog that pops us, click Get Throughput. Throughput information is displayed for inbound and outbound message separately. At startup time the adapter starts recording throughput information by default. If you want to record throughput information in a particular interval, click on disable timer and then click on enable timer. This restarts the recording process. Now you can get fresh throughput information by clicking Get Throughput.

- Track Messages—This allows you to track particular messages at runtime. During design time, in iStudio, you have the ability to specify fields in the application view for publish event or invoke procedure, that you would like to track the message with. For example, a customer related message could be tracked by an application view field -- social security number. To utilize this functionality in the runtime management console:

1. Right-click on any adapter that is attached to the application that is publishing the event or invoking the procedure that you want to track. On the context menu, select Track Messages.
2. A list of trackable events and procedures is displayed. Select the event or procedure you want to track.
3. A dialog pops us asking you for the value(s) of the tracking field(s) that you selected for that particular event or procedure in iStudio. So, in our customer related message example, the dialog would ask for the social security number.

- Get Errors—Get Errors: Shows you all the errors associated with the adapter. Certain errors have a message payload associated with them. These errors are marked with a `yes` in the Editable column. To modify and resubmit a message:
  1. Double-click the description field for such an error. A window pops up with an XML message in the application or common view depending upon the processing stage at which the error occurred.
  2. You can edit the message if you need to. Or you can skip this step.
  3. Click Resubmit. The message now re-enters the integration system for reprocessing.

# A

# Transformations

This appendix provides a list of the Oracle Applications InterConnect transformations.

## Copy Fields

Copy the source field(s) into the destination field(s).

**Parameters:**

None

## Copy Object

Copy the source object into the destination object

**Parameters:**

None

## Concat Fields

Concatenate the source field(s) and copy it into the destination field

**Parameters:**

| | |
|---|---|
| prefix | An optional prefix to the concatenated string. |
| separator | The separator, a string of characters, that separate source fields in the concatenated string. |
| suffix | An optional suffix to the concatenated string. |

## Expand Fields

Expand the source field into the destination fields.

**Parameters:**

| | |
|---|---|
| delimiter | The delimiter or string of characters around which the source field should be broken up. |

## Set Constant

Copy a constant into the destination fields.

**Parameters:**

| | |
|---|---|
| constant | The constant to be copied. |

## True Conditional Lookup XRef

Lookup the source field in a cross reference table. If condition is satisfied, copy it into the destination field.

**Parameters:**

| | |
|---|---|
| condition | The condition for this parameter. |
| table | The cross reference table. |
| pass through | When there is no corresponding cross reference, and this parameter is `true`, the destination field is set to the source field. If this parameter is `false`, the destination field is set to null. |

## True Conditional Lookup DVM

Lookup the source field in a domain value map table. If condition is satisfied, copy it into the destination field.

**Parameters:**

| | |
|---|---|
| condition | The condition for this parameter. |
| table | The domain value map table. |
| pass through | When there is no corresponding domain value map and this parameter is set to `true`, the destination field is set to the source field. If this parameter is set to `false`, the destinations field is set to null. |

## Conditional Copy

Copy the source field(s) into the destination field(s) if expression is satisfied.

**Parameters:**

| | |
|---|---|
| expression | The expression. |
| only copy on true | If this parameter is set to true and the expression evaluates to false, nothing is copied. If this parameter is set to false and the expression evaluates to false, the second input object is copied. |

## True Conditional Copy

Copy the source field(s) into the destination field(s) if condition is satisfied.

**Parameters:**

| | |
|---|---|
| condition | The condition for this parameter. |

## True Conditional Concat

Concatenate the source field(s) into the destination field if the condition is satisfied.

**Parameters:**

| | |
|---|---|
| condition | The condition for this parameter. |
| prefix | An optional prefix to the concatenated string. |
| separator | The separator, a string of characters, that separate source fields in the concatenated string. |
| suffix | An optional suffix to the concatenated string. |

## True Conditional To Number

Convert the sign, value, and precision source fields into a number and copy it into the destination field if condition is satisfied.

**Parameters:**

| | |
|---|---|
| condition | The condition for this parameter. |
| int length | The number of digits before the decimal point excluding the sign. |
| dec length | The number of digits after the decimal point. |
| character | The padding character. |
| DVM | An optional domain value map to lookup decimal point character. |

## False Conditional Copy

Copy the source field(s) into the destination field(s) if condition is NOT satisfied.

**Parameters:**

| | |
|---|---|
| condition | The condition for this parameter. |

## False Conditional Concat

Concatenate the source field(s) into the destination field if the condition is NOT satisfied.

**Parameters:**

| | |
|---|---|
| condition | The condition for this parameter. |
| prefix | An optional prefix to the concatenated string. |
| separator | The separator, a string of characters, that separate source fields in the concatenated string. |

| | |
|---|---|
| suffix | An optional suffix to the concatenated string. |

## False Conditional To Number

Convert the sign, value, and precision source fields into a number and copy it into the destination field if condition is NOT satisfied.

### Parameters:

| | |
|---|---|
| condition | The condition for this parameter. |
| int length | The number of digits before the decimal point excluding the sign. |
| dec length | The number of digits after the decimal point. |
| character | The padding character. |
| DVM | An optional domain value map to lookup decimal point character. |

## To Number

Convert the sign, value, and precision source fields into a number and copy it into the destination field.

### Parameters:

| | |
|---|---|
| int length | The number of digits before the decimal point excluding the sign. |
| dec length | The number of digits after the decimal point. |
| character | The padding character. |
| DVM | An optional domain value map to lookup decimal point character. |

`

Copy a substring of the source field into the destination field.

**Parameters:**

| | |
|---|---|
| begin index | The index at which the substring begins. |
| length | An optional length of the substring. |

## Char Replace

Replace characters in the source field and copy it into the destination field.

**Parameters:**

| | |
|---|---|
| targets | The string of characters to replace. |
| replacements | The string of replacement characters. |

## String Replace

Replace each occurrence of a string in the source field and copy it into the destination field.

**Parameters:**

| | |
|---|---|
| targets | The string of characters to replace. |
| replacements | The string of replacement characters. |

## LTrim

Delete source field characters starting from the left until a character from the set is found and copy the remaining string into the destination field.

**Parameters:**

| | |
|---|---|
| characters | The string of characters to seek that stop the deletion. |

## RTrim

Delete source field characters starting from the right until a character from the set is found and copy the remaining string into the destination field.

### Parameters:

| | |
|---|---|
| characters | The string of characters to seek that stop the deletion. |

## LPad

Pad source field starting from the left for a given length and copy it into the destination field.

### Parameters:

| | |
|---|---|
| length | The padding length. |
| character | An optional character to pad with, default is <space>. |

## RPad

Pad source field starting from the right for a given length and copy it into the destination field.

### Parameters:

| | |
|---|---|
| length | The padding length. |
| character | An optional character to pad with, default is <space>. |

## Lookup XRef

Lookup the source field in a cross reference table and copy it into the destination field.

**Parameters:**

| | |
|---|---|
| table | The cross reference table. |
| pass through | When there is no corresponding cross reference and this parameter is set to `true`, the destination field is set to the source field. If this parameter is set to false, the destination field is set to `null`. |

## Delete XRef

Delete the source field from a cross reference table.

**Parameters:**

| | |
|---|---|
| table | The cross reference table. |

## Lookup DVM

Lookup the source field in a domain value map table and copy it into the destination field.

**Parameters:**

| | |
|---|---|
| table | The cross reference table. |
| pass through | When there is no corresponding domain value map and this parameter is set to `true`, the destination field is set to the source field. If this parameter is set to false, the destination field is set to `null`. |

## Truncate

Truncate source field starting from the right for a given length and copy it into the destination field.

### Parameters:

| | |
|---|---|
| length | The length to truncate. |

## Increment

Increment a counter and copy the incremented value into the destination field.

### Parameters:

| | |
|---|---|
| start value | The initial counter value. |
| counter | Give this counter a name to distinguish it from other counters that may be at different values at a given time and may have a different step size. |
| step size | The increment size. |

# Index


## A

activity, 3-9
  populating a business process with, 3-13
adapters, 1-14
  agents, bridges, 4-7
  common features, 5-4
  prepackaged, 1-9
  sdk, 1-15
agents, 1-9, 4-7
application data types, 2-12
application view, 2-12, 2-31
applications, 2-9
  creating, 2-31
attributes
  adding to common data types, 2-24
  deleting and clearing from common data types, 2-27
  importing for common data types, 2-25
  modifying mappings, 2-73
  removing mappings, 2-73

## B

bridges, 1-9, 4-7
business objects, 2-10
  creating, 2-22
business process, 1-7, 3-9
  creating, 3-11
  populating with activities, 3-13

## C

common data types, 2-11
  adding attributes, 2-24
  creating, 2-23
  deleting and clearing attributes, 2-27
  importing attributes, 2-25
common view, 2-9, 2-22
components, 1-13, 4-7
  adapter sdk, 1-15
  adapters, 1-14, 4-7
  advanced queues, 4-10
  istudio, 1-13
  istudio sdk, 1-15
  repository, 1-14, 4-9
  runtime console, 1-14
  workflow, 4-10
console monitors, 5-2
content-based routing, 2-15, 4-4
  working with, 2-61
cross reference tables, 2-15
  adding applications, 2-67
  populating, 2-69
  removing applications, 2-68
  working with, 2-67
custom transformations
  adding, 2-73
  deleting, 2-73

## D

deployment
  to oracle workflow, 3-15
design time, 2-2
design time tools, 3-4
domain value mapping, 2-15
domain value mapping tables

deleting, 2-72
domain value mappings
adding applications to, 2-70
deleting, 2-72
modifying data in, 2-71
removing, 2-71
working with, 2-70

## E

enabling infrastructure, 2-61
enterprise manager console
features, 5-2
event maps, 2-14
events, 2-10
creating, 2-28
publishing, 2-33
subscribing, 2-42

## F

features, 1-2
adapter, 5-5
error management, 4-4
extensibility using sdks, 1-11
flexible deployment, 1-13
integration lifecycle management, 1-8
integration logic drives platform services, 1-3
integration logic, platform services, 1-3
integration methodology, 1-4
istudio, 1-8
message delivery, 4-3
message retention, 4-4
messaging paradigms, 4-3
messaging services, 1-11
oracle infrastructure, 1-9
prepackaged adapters, 1-9
repository, 5-5
routing support, 4-4
scalability and load balancing, 4-5
value added, 1-12

## H

hub and spoke
how it works, 1-4, 2-4

## I

infrastructure, 1-9
enabling, 2-61
installation
oracle workflow components, 3-8
integration architecture, 4-2
integration logic, 1-3
integration methodology, 1-4, 2-4
integration points, 2-9
integration process, 2-2
design time, 2-2
overview, 2-2
runtime, 2-2
istudio, 1-13, 2-8
adding applications to cross reference tables, 2-67
adding applications to domain value mappings, 2-70
adding custom transformations, 2-73
adding mapping variables, 2-74
application data types, 2-12
application view, 2-12
applications, 2-9
business objects, 2-10
common data types, 2-11
common view, 2-9
concepts, 2-8
content-based routing, 2-15, 2-61
creating a business object, 2-22
creating a new project, 2-17
creating a new workspace, 2-19
creating a procedure, 2-29
creating an application, 2-31
creating an event, 2-28
creating common data types, 2-23
cross reference tables, 2-15, 2-67
deleting custom transformations, 2-73
deleting domain value mapping tables, 2-72
deleting domain value mappings, 2-72
deleting mapping variables, 2-75
design time tool, 2-8
domain value mapping, 2-15

domain value mappings, 2-70
event maps, 2-14
events, 2-10
exporting stored procedures, 2-21
implementing a procedure, 2-54
integration points, 2-9
integration specification, 1-8
invoking a procedure, 2-46
mapping, transformations, 2-12
metadata versioning, 2-13
modifying attribute mappings, 2-73
modifying data in domain value mappings, 2-71
opening a workspace, 2-20
populating cross reference tables, 2-69
procedures, 2-11
projects, 2-9
publishing an event, 2-33
removing applications from cross reference tables, 2-68
removing applications from domain value mappings, 2-71
removing attribute mappings, 2-73
routing, message capability matrix, 2-15
sdk, 1-15
subscribing to an event, 2-42
toolbar, 2-16
tracking fields, 2-16
using, 2-16
using workspaces, 2-19
workspaces, 2-8
workspaces, projects, 2-17

## M

mapping, 2-12
mapping and transformations, 2-12
mapping variables
adding, 2-74
deleting, 2-75
message capability matrix, 2-15
message delivery, 4-3
messaging
middleware services, 1-11
point-to-point, 2-7
publish, subscribe, 2-7
request, reply, 2-7
metadata versioning, 2-13

## O

oracle application interconnect
business process collaboration, 1-7
dependent products, 1-15
oracle database, 1-15
oracle applications interconnect
architecture, 1-16
core components, 1-13
features, 1-2
message delivery, 4-3
overview, 1-2
using oracle workflow with, 3-8
oracle database, 1-15
oracle workflow, 1-15, 3-2
composite services, 3-3
deploy business process for runtime, 3-8
deployment, 3-15
design business process, 3-8, 3-9
design time tools, 3-4
error management, 3-2
human interaction, 3-3
installing components, 3-8
integration with oracle applications interconnect, 3-4
launching oracle workflow builder, 3-20
launching the home page, 3-19
launching tools, 3-19
message junctions, 3-3
modify existing processes, 3-21
overview, 3-2
runtime, 3-6
solves business problems, 3-2
stateful routing, 3-3
using with oracle applications interconnect, 3-8

## P

platform services, 1-3
procedures, 2-11
creating, 2-29

exporting stored procedures, 2-21
implementing, 2-54
invoking, 2-46
process, 2-2
process bundle, 3-9
creating, 3-10
projects, 2-8, 2-9, 2-17
creating, 2-17

## R

repository, 1-14
common features, 5-4
resubmission, 4-5
routing, 2-15
runtime, 2-2, 3-6
features, 4-3
runtime console, 1-14
runtime management, 5-2

## S

sdk
extensibility, 1-11
stored procedures, 2-21

## T

tracing, 4-5
tracking, 4-5
tracking fields, 2-16
transformations, 2-12
', A-6
char replace, A-7
concat fields, A-2
conditional copy, A-4
copy fields, A-2
copy object, A-2
delete xref, A-9
expand fields, A-2
false conditional concat, A-5
false conditional copy, A-5
false conditional to number, A-6
increment, A-10
l trim, A-7
lookup dvm, A-9
lookup xref, A-9
lpad, A-8
r trim, A-8
rpad, A-8
set constant, A-3
string replace, A-7
to number, A-6
true conditional concat, A-4
true conditional copy, A-4
true conditional lookup dvm, A-3
true conditional lookup xref, A-3
true conditional to number, A-5
truncate, A-10

## V

views
application view, 2-12

## W

workspaces, 2-8, 2-17
creating, 2-19
new, 2-8
old, 2-8
opening, 2-20
using, 2-19