BEA WebLogic Integration Adapter for CORBA®

User Guide

Version 8.1.1
Document Revised: October 2003

Copyright

Contents

About This Document

This document describes how to install, configure, and use the BEA WebLogic Adapter for CORBA. This document is organized as follows:

- Chapter 1, "Introducing the BEA WebLogic Adapter for CORBA," describes the adapter, how it relates to both CORBA business objects and WebLogic Integration.
- Chapter 2, "Generating Schemas for CORBA Integration Objects," describes how to generate schemas for your CORBA business objects using the BEA Application Explorer.
- Chapter 3, "Defining Application Views for CORBA," describes application views and how to use them to configure services.
- Appendix A, "Supported IDL Types," lists the IDL variable types supported by the adapter.
- Appendix B, "Sample Files," describes the sample files provided with the adapter.

Who Should Read This Documentation

This document is intended for the following members of an integration team:

- Integration Specialists—Lead the integration design effort. Integration specialists have expertise in defining the business and technical requirements of integration projects, and in designing integration solutions that implement specific features of WebLogic Integration. The skills of integration specialists include business and technical analysis, architecture design, project management, and WebLogic Integration product knowledge.
- Technical Analysts—Provide expertise in an organization's information technology infrastructure, including telecommunications, operating systems, applications, data

repositories, future technologies, and IT organizations. The skills of technical analysts include technical analysis, application design, and information systems knowledge.

- Enterprise Information System (EIS) Specialists—Provide domain expertise in the systems that are being integrated using WebLogic Integration adapters. The skills of EIS specialists include technical analysis and application integration design.
- System Administrators—Provide in-depth technical and operational knowledge about databases and applications deployed in an organization. The skills of system administrators include capacity and load analysis, performance analysis and tuning, deployment topologies, and support planning.

Additional Information

To learn more about the software components associated with the adapter, see the following documents:

- *BEA WebLogic Adapter for CORBA Release Notes*

  `http://edocs.bea.com/wladapters/corba/docs811/pdf/relnotes.pdf`
- *BEA WebLogic Adapter for CORBA Installation and Configuration Guide*

  `http://edocs.bea.com/wladapters/corba/docs811/pdf/install.pdf`
- *BEA Application Explorer Installation and Configuration Guide*

  `http://edocs.bea.com/wladapters/bae/docs811/pdf/install.pdf`
- *Introduction to the BEA WebLogic Adapters*

  `http://edocs.bea.com/wladapters/docs81/pdf/intro.pdf`
- BEA WebLogic Adapters 8.1.1 Dev2Dev Product Documentation

  `http://dev2dev.bea.com/products/wladapters/index.jsp`
- Application Integration documentation

  `http://edocs.bea.com/wli/docs81/aiover/index.html`

  `http://edocs.bea.com/wli/docs81/aiuser/index.html`
- BEA WebLogic Platform documentation

  `http://edocs.bea.com/platform/docs81/index.html`
- CORBA documentation

  `http://www.omg.org`

How to Use This Document

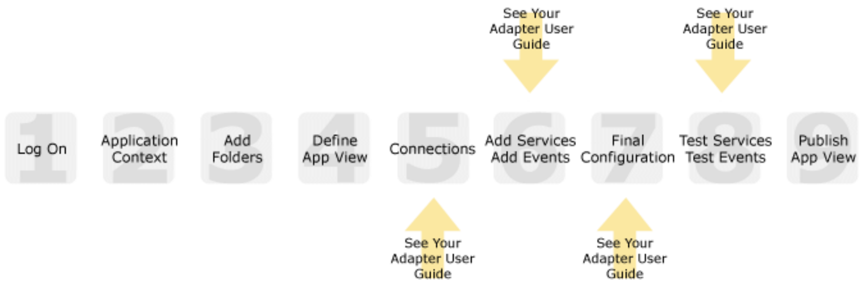
This document is designed to be used in conjunction with *Using the Application Integration Design Console*, available at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

Using the Application Integration Design Console describes, in detail, the process of defining an application view, which is a key part of making an adapter available to process designers and other users. What *Using the Application Integration Design Console* does *not* cover is the specific information about Adapter for CORBA that you need to supply to complete the application view definition. You will find that information in this document.

At each point in *Using the Application Integration Design Console* where you need to refer to this document, you will see a note that directs you to a section in your adapter user guide, with a link to the edocs page for adapters. The following roadmap illustration shows where you need to refer from *Using the Application Integration Design Console* to this document.

Figure 1 Information Interlock with ***Using the Application Integration Design Console***


See Your Adapter User Guide
See Your Adapter User Guide
Log On
Application Context
Add Folders
Define App View
Connections
Add Services Add Events
Final Configuration
Test Services Test Events
Publish App View
See Your Adapter User Guide
See Your Adapter User Guide

Contact Us!

Your feedback on the BEA WebLogic Adapter for CORBA documentation is important to us. Send us e-mail at **docsupport@bea.com** if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the BEA WebLogic Adapter for CORBA documentation.

In your e-mail message, please indicate that you are using the documentation for BEA WebLogic Adapter for CORBA and the version of the documentation.

If you have any questions about this version of BEA WebLogic Adapter for CORBA, or if you have problems using the BEA WebLogic Adapter for CORBA, contact BEA Customer Support through BEA WebSUPPORT at **www.bea.com**. You can also contact Customer Support by using the contact information provided on the Customer Support Card which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	Item
boldface text	Indicates terms defined in the glossary.
Ctrl+Tab	Indicates that you must press two or more keys simultaneously.
italics	Indicates emphasis or book titles.

`monospace text`	Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.
	Examples:
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

`monospace boldface text`	Identifies significant words in code. *Example*: `void` **`commit`** `( )`
`monospace italic text`	Identifies variables in code. *Example*: `String` *`expr`*
UPPERCASE TEXT	Indicates device names, environment variables, and logical operators. *Examples*: LPT1 SIGNON OR
`{ }`	Indicates a set of choices in a syntax line. The braces themselves should never be typed.

Convention	Item

`[ ]`	Indicates optional items in a syntax line. The brackets themselves should never be typed.
	Example:
	`buildobjclient [-v] [-o name ] [-f` *`file-list`*`]...`
	`[-l` *`file-list`*`]...`

|	Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed.

`...`	Indicates one of the following in a command line:
	• That an argument can be repeated several times in a command line
	• That the statement omits additional optional arguments
	• That you can enter additional parameters, values, or other information
	The ellipsis itself should never be typed.
	Example:
	`buildobjclient [-v] [-o name ] [-f` *`file-list`*`]...`
	`[-l` *`file-list`*`]...`

. . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.

Introducing the BEA WebLogic Adapter for CORBA

This section introduces the BEA WebLogic Adapter for CORBA and describes how the adapter enables integration with CORBA business objects and WebLogic Integration.

It includes the following topics:

- About the BEA WebLogic Adapter for CORBA
- Getting Started With the Adapter for CORBA

About the BEA WebLogic Adapter for CORBA

The BEA WebLogic Adapter for CORBA connects to your CORBA system so that you can easily use your CORBA data and functions within your business processes. The adapter provides scalable, reliable, and secure access to your CORBA system.

This section includes the following topics:

- Supported CORBA Operations for Application Integration
- Benefits of the Adapter for CORBA

Supported CORBA Operations for Application Integration

The Adapter for CORBA supports synchronous and asynchronous, bi-directional message interactions for CORBA Business Services, Business Components, and Integration Objects. Access to CORBA environments is provided through the adapter, which uses CORBA Interface Definition Language (IDL) entries to generate local, remote-based services. Applications make calls to a remote service that, in turn invokes a CORBA method and receives return information from the CORBA method.

It provides integration with the following CORBA operations:

- Access to CORBA integration objects using XML to handle services
- Guaranteed synchronous and asynchronous message interactions between WebLogic Integration and an ORB.

Benefits of the Adapter for CORBA

The combination of the adapter and WebLogic Integration supplies everything you need to integrate your business processes and enterprise applications with your CORBA system. The Adapter for CORBA provides these benefits:

- Support for several ORBs, including JacORB, Orbix2000, and VisiBroker for Java.
- Integration can be achieved without custom coding.
- Business processes can request and receive data from your CORBA system using services.
- Adapter services are standards-based. The adapter services provide extensions to the *J2EE Connector Architecture* (JCA) version 1.0 from Sun Microsystems, Inc. For more information, see the Sun JCA page at the following URL:

  `http://java.sun.com/j2ee/connector/`
- The adapter and WebLogic Integration solution is scalable. The BEA WebLogic Platform provides clustering, load balancing, and resource pooling for a scalable solution. For more information about scalability, see the following URL:

  `http://edocs.bea.com/wls/docs81/cluster/index.html`
- The adapter and WebLogic Integration solution benefits from the fault-tolerant features of the BEA WebLogic Platform. For more information about high availability, see the following URL:

  `http://edocs.bea.com/wli/docs81/deploy/index.html`
- The adapter and WebLogic Integration solution is secure, using the security features of the BEA WebLogic Platform and the security of your CORBA system. For more information about security, see the following URL:

  `http://edocs.bea.com/wls/docs81/secintro/index.html`

Getting Started With the Adapter for CORBA

This section gives an overview of how to get started using the BEA WebLogic Adapter for CORBA within the context of an application integration solution. Integration with CORBA involves the following tasks:

- Step 1: Design the Application Integration Solution
- Step 2: Determine the Required CORBA Business Workflows
- Step 3: Generate Schemas for CORBA Integration Objects
- Step 4: Define Application Views and Configure Services
- Step 5: Integrate with Other BEA Software Components
- Step 6: Deploy the Solution to the Production Environment

Step 1: Design the Application Integration Solution

The first step is to design an application integration solution, which includes (but is not limited to) such tasks as:

- Defining the overall scope of application integration.
- Determining the business process(es) to integrate.
- Determining which WebLogic Platform components will be involved in the integration, such as web services or business processes designed in WebLogic Workshop, portals created in WebLogic Portal, and so on.
- Determining which external systems and technologies will be involved in the integration, such as CORBA systems and other EISs.
- Determining which BEA WebLogic Adapters for WebLogic Integration will be required, such as the BEA WebLogic Adapter for CORBA. An application integration solution can involve multiple adapters.

This step involves the expertise of business analysts, system integrators, and EIS specialists (including CORBA specialists). Note that an application integration solution can be part of a larger integration solution.

Step 2: Determine the Required CORBA Business Workflows

Within the larger context of an application integration project, you must determine which specific CORBA integration objects and workflows are required to support the business processes in the application integration solution.

Factors to consider include (but are not limited to):

- Type of CORBA integration objects, workflows, and transport used to access the CORBA system.
- CORBA transactions involved in business processes
- Logins required to access CORBA transports and perform the required operations
- Whether services should be processed synchronously or asynchronously

This step involves the expertise of CORBA specialists, including analysts and administrators.

Step 3: Generate Schemas for CORBA Integration Objects

After identifying the CORBA integration objects and workflows required for the application integration solution, you must generate the XML schemas that will be used to exchange data with one or more CORBA systems:

- Services require two XML schemas: one for the CORBA request and another for the CORBA response.

You use the BEA Application Explorer tool to generate schemas for CORBA operations. To learn more about schemas, see Chapter 2, "Generating Schemas for CORBA Integration Objects."

Step 4: Define Application Views and Configure Services

After you create the schemas for your CORBA services, you create an application view that provides an XML-based interface between WebLogic Server and a particular CORBA system within your enterprise. If you are accessing multiple CORBA systems, you define a separate application view for each CORBA system you want to access. To provide different levels of security access (such as "guest" and "administrator"), define a separate application view for each security level.

Once you define an application view, you can configure services in that application view that employ the XML schemas that you created in Step 3: Generate Schemas for CORBA Integration

Objects. To learn more about generating schemas, see Chapter 2, "Generating Schemas for CORBA Integration Objects."

To learn more about defining application views, see Chapter 3, "Defining Application Views for CORBA" in conjunction with*Using the Application Integration Design Console*, at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

Step 5: Integrate with Other BEA Software Components

Once you have configured and published one or more application views for CORBA integration, you can integrate these application views into other BEA software components, such as business processes or web services created in BEA WebLogic Workshop, or portals built with BEA WebLogic Portal.

For more information, see *Using the Application Integration Design Console*, particularly Chapter 3, "Using Application Views with Business Processes," at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/3usruse.html`

Step 6: Deploy the Solution to the Production Environment

After you have designed, built, and tested your application integration solution, you can deploy it into a production environment. The following list describes some of the tasks involved in deploying an application integration:

- Design the deployment.
- Deploy the required components of the BEA WebLogic Platform.
- Install and deploy the BEA WebLogic Adapter for CORBA as described in *BEA WebLogic Adapter for CORBA Installation and Configuration Guide*
- Deploy your application views and schemas for CORBA integration.
- Verify business processes in the production environment.
- Monitor and tune the deployment.

To learn more about deploying your application integration solution, see *Deploying WebLogic Integration Solutions* at the following URL:

`http://edocs.bea.com/wli/docs81/deploy/index.html`

Generating Schemas for CORBA Integration Objects

The Adapter for CORBA uses XML documents to communicate with your CORBA system's integration objects for services. The format of these XML documents is determined by schemas you generate using the BEA Application Explorer.

This section explains how to use the BEA Application Explorer to generate schemas. It contains the following topics:

Before You Begin

Before you begin to generate schema for the Adapter for CORBA, you must:

- Download and install the BEA Application Explorer software. To learn more, see the *BEA Application Explorer Installation and Configuration Guide* at the following URL:

  `http://edocs.bea.com/wladapters/docs81/index.html`

- Obtain the information necessary to connect to your CORBA system. Contact your CORBA administrator for this information.

About the BEA Application Explorer

The BEA Application Explorer uses intelligence about CORBA combined with metadata provided by the CORBA system to generate the schemas required to build application view services.

This section contains the following topics:

- About the Process for Defining Schemas
- Types of Schemas Generated by the BEA Application Explorer

About the Process for Defining Schemas

The process for defining XML schemas includes the following steps:

1. Starting the BEA Application Explorer.
2. Setting the Session Path.

   The BEA Application Explorer uses this path to create the directory for the schemas.
3. Creating a New Connection or Using an Existing Connection.
4. Creating Schemas for Services.

Note: Before you create schemas and application views, you can save time by verifying that your ORB infrastructure is properly configured, your server is registered in the Naming Service, and your interface repository (IFR) is running and populated.

Types of Schemas Generated by the BEA Application Explorer

Each service the Adapter for CORBA uses must be defined by a schema. The BEA Application Explorer generates XML schemas for:

- Service Requests
- Service Responses

Service Requests

Service requests are requests for action that your application makes to your CORBA system. Requests are defined by request schema. As part of the definition, the request schema defines the input parameters required by the CORBA system. The CORBA system responds to the request with a service response.

Service Responses

Service responses are the way the CORBA system responds to a service request. A service response schema defines this service response. Service requests always have corresponding responses.

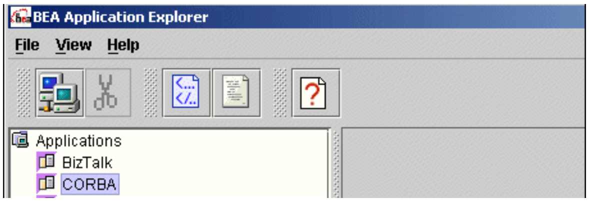
Starting the BEA Application Explorer

You use the BEA Application Explorer to generate service request schemas and service response schemas. The schemas you create are published in the WebLogic Integration repository.

To start the BEA Application Explorer:

1. Open the BEA Application Explorer.
   - In Windows, choose Windows Start→Programs→BEA Application Explorer.
   - On UNIX, run the startup script `beabse.sh` or the Java command `java com.ibi.common.ui.StartPanel`.

   The BEA Application Explorer window is displayed.



Setting the Session Path

The session path determines the directory where the BEA Application Explorer places your generated XML schemas and connection information. Your schemas are stored here:

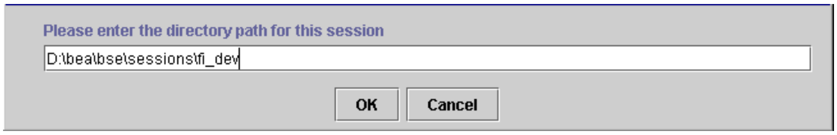
- On Windows: `session_path`\corba\`connection_name`\schemas
- On UNIX: `session_path`/corba/`connection_name`/schemas

Here, `connection_name` is the value you specify when you select a connection. To learn more about selecting a connection, see Managing CORBA Connections.

To set the session path:

1. From the File menu, choose Session.

   The Enter Session Path window is displayed. It includes a default directory path.



Please enter the directory path for this session

D:\bea\bse\sessions\fi_dev

2. Do one of the following:
   - To accept the default session path, click OK.
   - To specify a different path, enter the path and click OK.

     Specifying a different path allows you to group your schema according to project, or other logical group.

Managing CORBA Connections

The BEA Application Explorer must connect to your CORBA system before you can generate schemas. Therefore, you must first define a connection to your CORBA system.

This section includes the following topics:

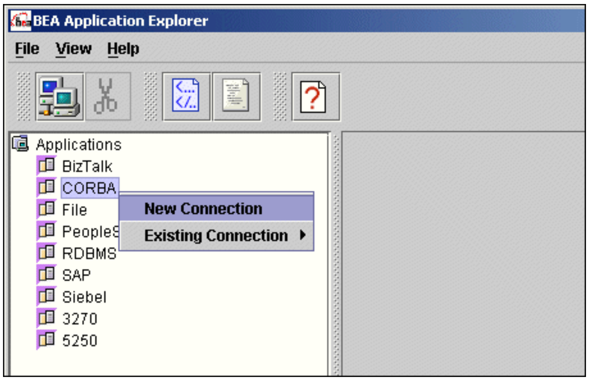
- Creating a New Connection
- Using an Existing Connection
- Disconnecting from CORBA
- Removing Connections

Creating a New Connection

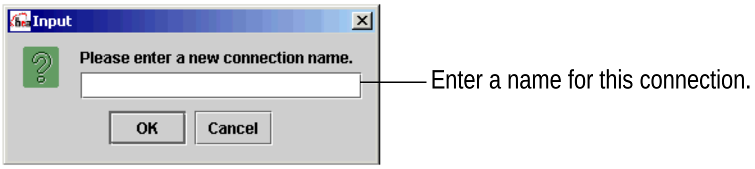
If you are creating a new connection, be sure to check that you have the correct information for your ORB.

To create a new connection:

1. In the left pane of the BEA Application Explorer window, under Applications right-click CORBA→ New Connection.

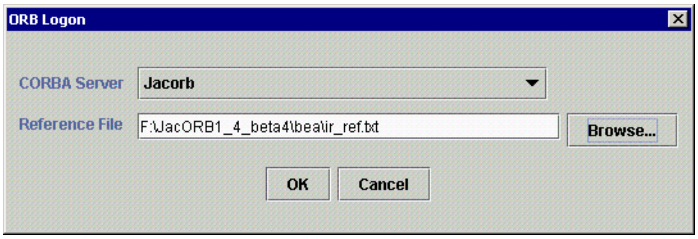


The BEA Application Explorer prompts you for a connection name.


Input
Please enter a new connection name.
OK
Cancel
Enter a name for this connection.

2. Enter a name for this connection and click OK.

The ORB logon window is displayed.



CORBA Server: Jacorb

Reference File: F:\JacORB1_4_beta4\bea\ir_ref.txt

3. Select CORBA server and specify the name and path of the Interface repository (IFR) reference file. This file contains the parameters required to connect to the CORBA system.

Note: The example here illustrates accessing JacORB services. For other ORBs there may be additional parameters, such as Naming Service, Host, Port, and Enable IIOP Tracing. To learn more about these parameters, see your ORB documentation.

The BEA Application Explorer loads the information and connects to the ORB to extract the object definitions from the Interface Repository.

4. Enter the parameters for your system.
5. Click OK.

   The new connection is displayed under the CORBA node in the BEA Application Explorer window. You can now view business objects and services, as well as all available integration objects in your CORBA system.

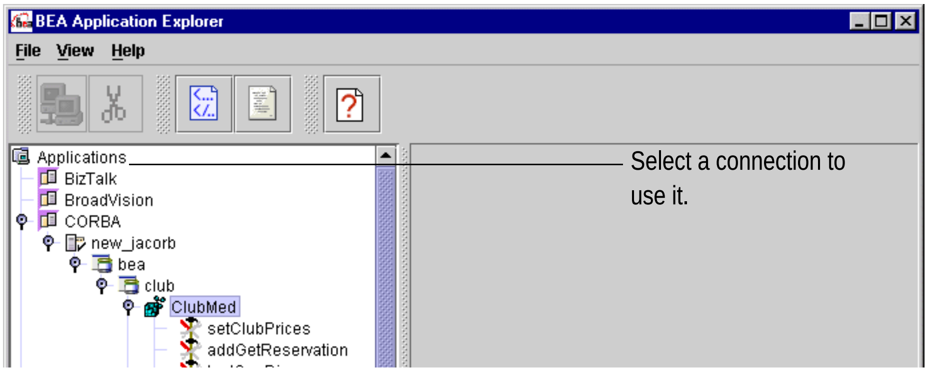
Using an Existing Connection

You can use an existing connection rather than creating a new one.

To use an existing CORBA connection:

1. In the left pane of the BEA Application Explorer window, under Applications right-click CORBA→ Existing Connection→ *your connection*.

   The connection is displayed below the CORBA node.



2. If the connection parameters do not correspond to your system, edit them in the CORBA Logon Window.
3. Click OK.

Disconnecting from CORBA

The BEA Application Explorer allows you to disconnect from CORBA.

To disconnect from CORBA:

1. In the left pane of the BEA Application Explorer, right-click on the connection.
2. Choose Disconnect.

This disconnects from CORBA, and the connection icon change to indicate that is not currently connected. To re-establish the connection, right-click on the connection and choose Connect.

Removing Connections

The BEA Application Explorer allows you to remove connections when you no longer need them.

To remove a connection:

1. In the left pane of the BEA Application Explorer, right-click on the connection.
2. Choose Remove.

Managing Schemas

You need to create a schema for each service your application uses. You use the BEA Application Explorer to create these schemas.

This section explains:

- Creating Schemas for Services
- Removing Schemas

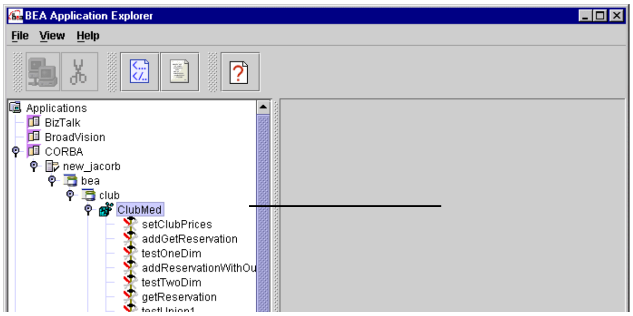
Creating Schemas for Services

Services require two schemas, one for the request and one for the response. Services always have these two schemas, even is the response is not used by your application.

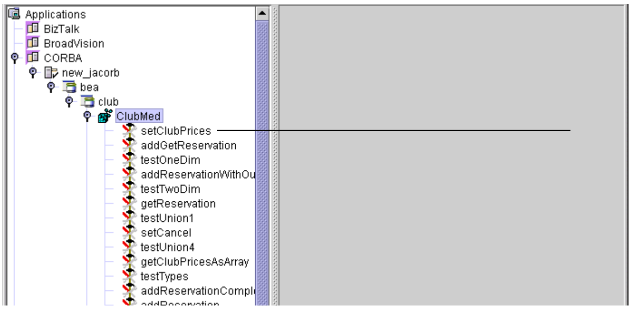
To create a schema for a service:

1. Start BEA Application Explorer. To learn more, see Starting the BEA Application Explorer.
2. Set the session path. This determines where the BEA Application Explorer places your schemas. To learn more, see Setting the Session Path.

3. Select or create a connection to CORBA. To learn more, see Managing CORBA Connections.
4. Expand the tree under Applications → CORBA → *connection name* → Integration Objects to see the items for which you may create a schema. If you cannot expand the tree beneath CORBA, you have not set a connection for CORBA.

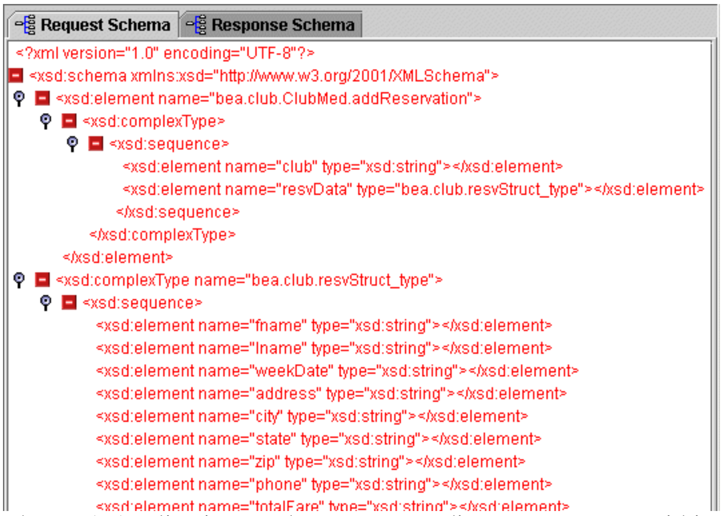


5. Select the integration object for this schema.



6. Right-click the item for which you wish to create the schema and choose Create Service Schemas.

   The BEA Application Explorer displays tabs that show the request and response schemas.


Request Schema | Response Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="bea.club.ClubMed.addReservation">
    <xsd:complexType>
      <xsd:sequence>
          <xsd:element name="club" type="xsd:string"></xsd:element>
          <xsd:element name="resvData" type="bea.club.resvStruct_type"></xsd:element>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  <xsd:complexType name="bea.club.resvStruct_type">
    <xsd:sequence>
        <xsd:element name="fname" type="xsd:string"></xsd:element>
        <xsd:element name="lname" type="xsd:string"></xsd:element>
        <xsd:element name="weekDate" type="xsd:string"></xsd:element>
        <xsd:element name="address" type="xsd:string"></xsd:element>
        <xsd:element name="city" type="xsd:string"></xsd:element>
        <xsd:element name="state" type="xsd:string"></xsd:element>
        <xsd:element name="zip" type="xsd:string"></xsd:element>
        <xsd:element name="phone" type="xsd:string"></xsd:element>
        <xsd:element name="totalFare" type="xsd:string"></xsd:element>
```

The BEA Application Explorer creates a directory structure within the working directory you identified earlier. In this example, the working directory is `C:\BEA\BEASCHEMAS`.

Within this directory, the BEA Application Explorer creates a folder called `corba` as well as subfolders to hold the schemas for each configured CORBA connection. In this example, the schemas have been created in the folder called `CorbaConnection`, and the BEA Application Explorer adds the following items to the folder `C:\BEA\BEASCHEMAS\corba\CorbaConnection`:

– `manifest.xml`

– `service_Sample Account1-1-FA22.xsd`

– `service_Sample Account1-1-FA22_response.xsd`

You have successfully created service request and response schemas for this integration object.

Removing Schemas

To remove a schema:

1. Right-click on an integration object for which there is at least one schema.

   If there are service schemas defined for this integration object, the menu includes a Remove Service Schema option.

2. Choose the appropriate option.

Next Steps

After you have defined schemas for your services, the next step is to create an application view. An application view makes the services available to applications. To learn more about application views, see Defining Application Views for CORBA.

Defining Application Views for CORBA

An application view is a business-oriented interface to objects and operations within an EIS.

This section presents the following topics:

- How to Use This Document
- Before You Begin
- About Application Views
- About Defining Application Views
- Defining Service Connection Parameters
- Setting Service Properties
- Testing Services

How to Use This Document

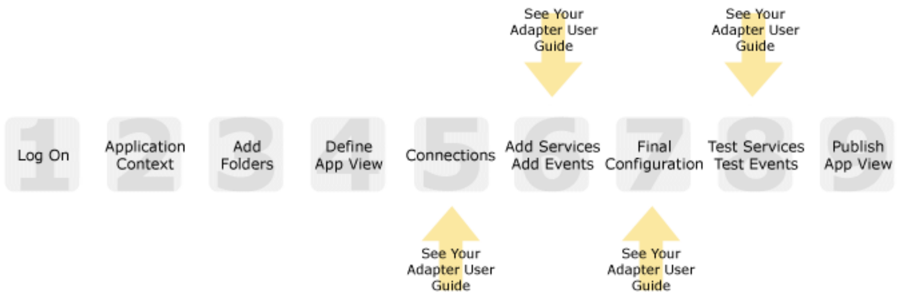
This document is designed to be used in conjunction with *Using the Application Integration Design Console*, available at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

Using the Application Integration Design Console describes, in detail, the process of defining an application view, which is a key part of making an adapter available to process designers and other users. What *Using the Application Integration Design Console* does *not* cover is the specific information—about connections to your CORBA system, as well as supported services—that you must supply as part of the application view definition. You will find that information in this section.

At each point in *Using the Application Integration Design Console* where you need to refer to this document, you will see a note that directs you to a section in your adapter user guide, with a link to the edocs page for adapters. The following road map illustration shows where you need to refer from *Using the Application Integration Design Console* to this document.

Figure 3-1 Information Interlock with ***Using the Application Integration Design Console***


See Your Adapter User Guide
See Your Adapter User Guide
Log On
Application Context
Add Folders
Define App View
Connections
Add Services Add Events
Final Configuration
Test Services Test Events
Publish App View
See Your Adapter User Guide
See Your Adapter User Guide

Before You Begin

Before you define an application view, make sure you have:

- Installed and deployed the adapter according to the instructions in *BEA WebLogic Adapter for CORBA Installation and Configuration Guide*.
- Determined which business processes need to be supported by the application view. The required business processes determine the types of services you include in your application views. Therefore, you must gather information about the application's business requirements from the business analyst. Once you determine the necessary business

processes, you can define and test the appropriate services. For more information, see "Getting Started With the Adapter for CORBA" on page 1-3.

- Gathered the connection information for your CORBA system. To learn more about the connection information needed by the BEA Application Explorer for your CORBA system, see "Before You Begin" on page 2-2.

About Application Views

An application view defines:

- Connection information for the EIS, including login information, connection settings, and so on.
- Service invocations, including the information the EIS requires for this request, as well as the request and response schemas associated with the service.

Typically, an application view is configured for a single business purpose and contains only the services required for that purpose. An EIS might have multiple application views, each defined for a different purpose.

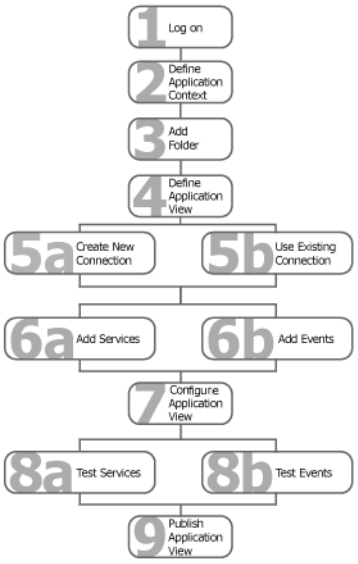
About Defining Application Views

Defining an application view is a multi-step process described in *Using the Application Integration Design Console*, available at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

The information you enter depends on the requirements of your business process and your EIS system configuration. Figure 3-2 summarizes the procedure for defining and configuring an application view.

Figure 3-2 Process for Defining and Configuring an Application View


1 Log on
2 Define Application Context
3 Add Folder
4 Define Application View
5a Create New Connection
5b Use Existing Connection
6a Add Services
6b Add Events
7 Configure Application View
8a Test Services
8b Test Events
9 Publish Application View

To define an application view:

1. Log on to the WebLogic Integration Application View Console.
2. Define the application context by selecting an existing application or specifying a new application name and root directory.

   This application will be using the services you define in your application view. The application view works within the context of this application.
3. Add folders as required to help you organize application views.
4. Define a new application view for your adapter.
5. Add a new connection service or select an existing one.

   If you are adding a new connection service, see "Defining Service Connection Parameters" on page 3-5 for details about CORBA requirements.
6. Add the services for this application view.

   See the following section for details about CORBA requirements:

   – "Setting Service Properties" on page 3-6

7. Perform final configuration tasks.

8. Test all services to make sure they can properly interact with the target CORBA system.

   See the following section for details about CORBA requirements:

   – "Testing Services" on page 3-10

9. Publish the application view to the target WebLogic Workshop application.

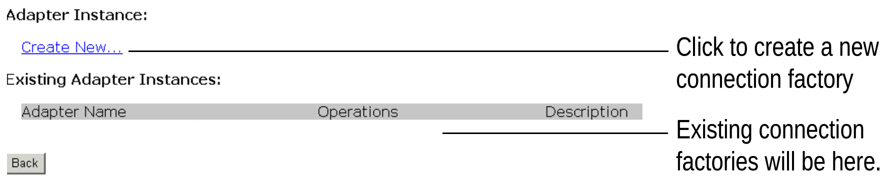
   This is the application you specified in step 2. Publishing the application view allows business process developers within the target application to interact with the newly published application view using an Application View control.

Defining Service Connection Parameters

This information applies to "Step 5A, Create a New Browsing Connection" in *Using the Application Integration Design Console*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

The Select Browsing Connection page allows you to choose the type of connection factory to associate with the application view. You can select a connection factory within an existing instance of the adapter or create a connection factory within a new adapter instance.

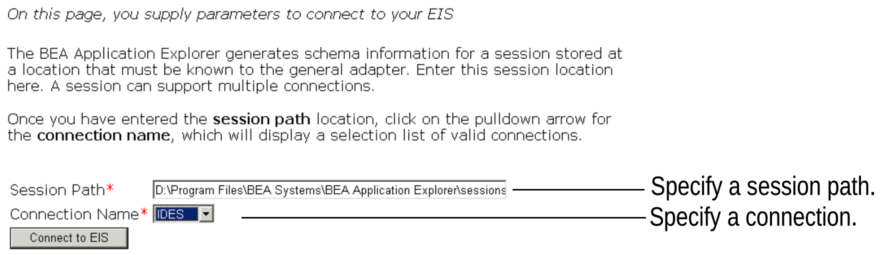

Adapter Instance:
Create New...
Existing Adapter Instances:
Adapter Name
Operations
Description
Back
Click to create a new connection factory
Existing connection factories will be here.

After you enter a connection name and description, you use the Configure Connection Parameters page to specify connection parameters for a connection factory.

To create a new browsing connection:

1. In the Create New Browsing Connections page, enter a connection name and description as described in *Using the Application Integration Design Console*.

   The Configure Connection Parameters page appears to allow you to configure the newly created connection factory within the new adapter instance.


On this page, you supply parameters to connect to your EIS

The BEA Application Explorer generates schema information for a session stored at a location that must be known to the general adapter. Enter this session location here. A session can support multiple connections.

Once you have entered the **session path** location, click on the pulldown arrow for the **connection name**, which will display a selection list of valid connections.

Session Path*	D:\Program Files\BEA Systems\BEA Application Explorer\sessions
Connection Name*	IDES

Note: A red asterisk (*) indicates that a field is required.

2. Specify a session path and connection name.

   This information enables the application view to interact with the target CORBA system. You need enter this information only once per application view.

3. Click Connect to EIS.

   You return to the Create New Browsing Connections, where you can specify connection pool parameters and logging levels. For more information, see *Using the Application Integration Design Console* at the following URL:

   http://edocs.bea.com/wli/docs81/aiuser/index.html

Setting Service Properties

This information applies to “Step 6A, Add a Service to an Application View” in *Using the Application Integration Design Console*, at the following URL:

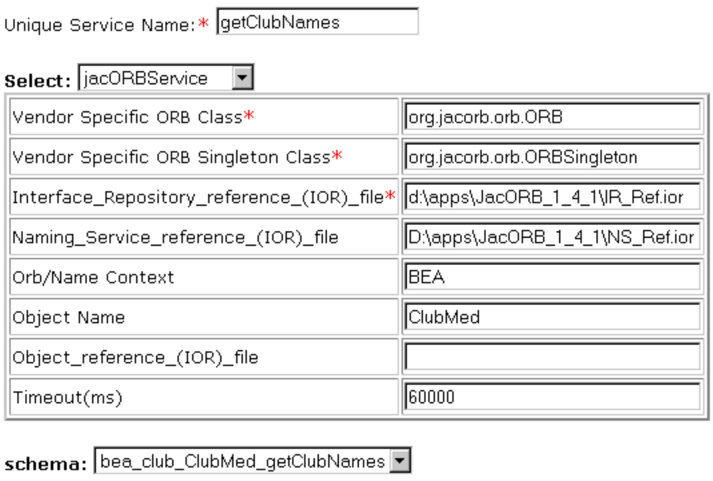
http://edocs.bea.com/wli/docs81/aiuser/index.html

Adapter for CORBA uses services to make requests of the CORBA system. A service consists of both a request and a response.

To configure a service:

1. Enter a unique service name that describes the function the service performs.

   The Add Services page displays the fields required for this service type.



Unique Service Name:*	getClubNames
Select:	jacORBService
Vendor Specific ORB Class*	org.jacorb.orb.ORB
Vendor Specific ORB Singleton Class*	org.jacorb.orb.ORBSingleton
Interface_Repository_reference_(IOR)_file*	d:\apps\JacORB_1_4_1\IR_Ref.ior
Naming_Service_reference_(IOR)_file	D:\apps\JacORB_1_4_1\NS_Ref.ior
Orb/Name Context	BEA
Object Name	ClubMed
Object_reference_(IOR)_file	
Timeout(ms)	60000
schema:	bea_club_ClubMed_getClubNames

Note: A red asterisk (*) indicates that a field is required.

2. Enter the following information:

Table 3-1 Service Parameters

Parameter	Description
Select	The type of client ORB libraries through which the service will be sent. The Application View Console automatically populates many of this window's fields with appropriate values based on the ORB type that you choose.
Vendor Specific ORB Class	The vendor-specific ORB class. The Application View Console automatically supplies this value based on the ORB that you chose.
Vendor Specific ORB Singleton Class	The vendor-specific ORB singleton class. The Application View Console automatically supplies this value based on the ORB that you chose.
Interface_Repository_reference_(IOR)_file	The name and path of the Interoperable Object Reference file that specifies the location of the Interface Repository service. The Application View Console automatically supplies this value based on the value supplied in the BEA Application Explorer.

Table 3-1 Service Parameters (Continued)

Parameter	Description
Naming_Service_refe rence_(IOR)_file	The name and path of the Interoperable Object Reference file that specifies the location of the Naming Service. This parameter, together with Orb/Name Context and Object Name, enable you to specify the location of an object using an indirect Naming Service reference.
Orb/Name Context	The ORB's name context. This parameter, together with Orb/Name Context and Object Name, enable you to specify the location of an object using an indirect Naming Service reference.
Object Name	The name of the object registered in the Naming Service (for example, `bea.clubmed`). This parameter, together with Orb/Name Context and Object Name, enable you to specify the location of an object using an indirect Naming Service reference.
Object_reference_(IO R)_file	The name and path of the Interoperable Object Reference file that specifies the location of the CORBA object. This parameter enables you to specify the location of an object using a direct IOR reference.
Timeout(ms)	The maximum time that a service will wait for a CORBA object to respond before the service terminates. It is measured in milliseconds. The default value, 0, specifies that the service will wait indefinitely for a response.

3. See "Common Settings" on page 3-8 for information about selecting a schema and configuring logging and tracing.

Common Settings

1 2 3 4 5 **6** 7 8 9

This information applies to "Step 6A, Add a Service to an Application View" in *Using the Application Integration Design Console*, at the following URL:

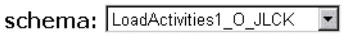
`http://edocs.bea.com/wli/docs81/aiuser/index.html`

You select a schema and select logging options the same way for all services.

To set common service settings:

1. In the Schema list, select the schema you want to use with this service.

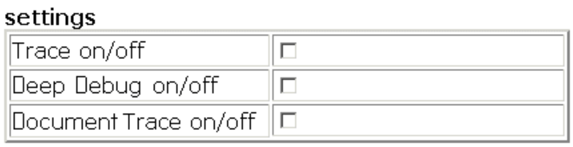
   For more information, see Chapter 2, "Generating Schemas for CORBA Integration Objects."

   schema: LoadActivities1_O_JLCK

2. Configure logging and tracing for this service, as follows:

   Logging captures information from your adapter and writes it in a log file. Tracing displays runtime information in the console. You set the type and amount of information you wish to capture as part of the final configuration tasks. This is described in detail in *Using the Application Integration Design Console*.

   settings

Trace on/off	☐
Deep Debug on/off	☐
Document Trace on/off	☐

   a. Select the Trace on/off check box to enable tracing for this service. Trace information appears in the runtime console.

   b. Select the Deep Debug check box to enable additional trace information for deeper troubleshooting.

   c. Select the Document Trace check box to enable recording of the documents that are sent to and from the adapter.

3. Click Add to add the service.

   For more information about the next step, see *Using the Application Integration Design Console* at the following URL:

   `http://edocs.bea.com/wli/docs81/aiuser/index.html`

Testing Services

This information applies to “Step 8A, Test an Application View’s Services” in *Using the Application Integration Design Console*, at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

The purpose of testing an application view service is to evaluate whether that service interacts properly with the target CORBA system. When you test a service, you supply any inputs required to start the service. For the Adapter for CORBA, the input is in the form of a valid XML string that acts as input for the service.

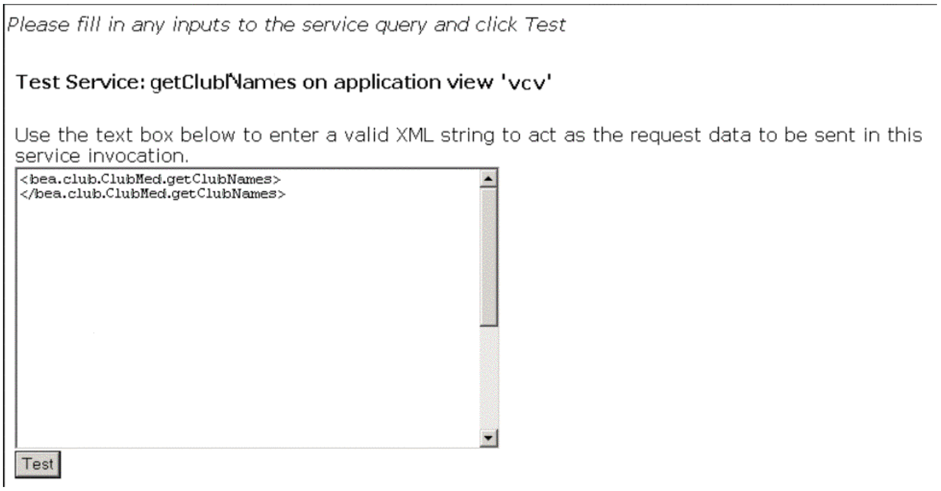
Note: You can test an application view only if it is deployed and only if it contains at least one service.

To test a service:

1. In the Application View Administration page, click the Test link beside the service to be tested.

   The Test Services page appears.

2. In the Test Service window, copy the appropriate XML strings for the service request.



Please fill in any inputs to the service query and click Test

Test Service: getClubNames on application view 'vcv'

Use the text box below to enter a valid XML string to act as the request data to be sent in this service invocation.

```
<bea.club.ClubMed.getClubNames>
</bea.club.ClubMed.getClubNames>
```

3. Click Test.

   The results appear in the Test Results window.

APPENDIX A

Supported IDL Types

The BEA WebLogic Adapter for CORBA supports the following Interface Definition Language (IDL) base types:

- boolean
- char
- double
- float
- long
- long long
- octet
- short
- string
- unsigned long
- unsigned long long
- unsigned short
- wchar

The BEA WebLogic Adapter for CORBA supports the following IDL constructed types:

- array (where dimension is less than or equal to 2)
- enum
- interface (as a return value, but not within a return value of another constructed type)
- sequence
- struct
- typedef
- union

Sample Files

This section describe the sample files delivered with the software. It includes the following topics:

- Sample XML Request and Response Documents
- Definitions for ClubMed Object

Sample XML Request and Response Documents

This section includes the following sample request and response documents:

- addGetReservation Request
- addGetReservation Responses
- AddReservation Request
- AddReservation Response
- addReservationComplex Request
- addReservationComplex Response
- addReservationWithOut Request
- addReservationWithOut Response
- cancelReservation Request
- cancelReservation Response

- getClub Request
- GetClubNames Request
- GetClubNames Response
- getClubNames4 Request
- getClubNames4 Response
- getClubNamesWithRef Request
- GetClubPrices Request
- GetClubPrices Response
- GetClubPricesAsArray Request
- GetClubPricesAsArray Response
- GetReservation Request
- GetReservation Response
- getReservationAsOut Request
- getReservationAsOut Response
- setCancel Request
- setCancel Response
- setClubPrices Request
- setClubPrices Response
- setClubPricesAsArray Request
- setClubPricesAsArray Response
- setStatus Request
- setStatus Response
- status Request
- status Response
- testOneDim Request

- testOneDim Response
- testTwoDim Request
- testTwoDim Response
- testTypes Request
- testTypes Response
- testUnion1-2 Request
- testUnion2-1 Request
- testUnion2-3 Request
- testUnion3-PROCESSED Request
- testUnion4-1 Request
- testUnion4-default Request

addGetReservation Request

Listing C-1 addGetReservation Request

```
<bea.club.ClubMed.addGetReservation>
 <club>BAMBU</club>
 <resvData>
    <fname>Bill</fname>;
    <lname>Wales</lname>
    <weekDate>Monday</weekDate>
    <street>Big Bungalow</street>
    <city>High End</city>
    <state>WA</state>
    <zip>99990</zip>
    <phone>123-123-3456</phone>
    <totalFare>5000.0</totalFare>
    <partyAdults>2</partyAdults>
    <partyChildren>2</partyChildren>
    <date>2001-09-21</date>
    <resv>1000</resv>
 </resvData>
</bea.club.ClubMed.addGetReservation>
```

addGetReservation Response

Listing C-2 addGetReservation Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
 <bea.club.ClubMed.addGetReservation>
 <return>1002</return>
    <resvData>
    <fname>Bill</fname>
    <lname>Wales</lname>
    <weekDate>Monday</weekDate>
    <street>Big Bungalow</street>
    <city>High End</city>
    <state>WA</state>
    <zip>99990</zip>
    <phone>123-123-3456</phone>
    <totalFare>5000.0</totalFare>
    <partyAdults>2</partyAdults>
    <partyChildren>2</partyChildren>
    <date>2001-09-21</date>
    <resv>1000</resv>
 </resvData>
</bea.club.ClubMed.addGetReservation>
```

AddReservation Request

Listing C-3 AddReservation Request

```
<bea.club.ClubMed.addReservation>
 <club>BAMBU</club>
 <resvData>
    <fname>Bill</fname>;
    <lname>Wales</lname>
    <weekDate>Monday</weekDate>
    <address>Big Bungalow</address>
    <city>High End</city>
    <state>WA</state>
    <zip>99990</zip>
    <phone>123-123-3456</phone>
    <totalFare>5000.0</totalFare>
    <partyAdults>2</partyAdults>
    <partyChildren>2</partyChildren>
    <date>2001-09-21</date>
 </resvData>
</bea.club.ClubMed.addReservation>
```

AddReservation Response

Listing C-4 AddReservation Response

```
<?xml version="1.0"?>
<bea.club.ClubMed.addReservation>
 <return>1001</return>
</bea.club.ClubMed.addReservation>
```

addReservationComplex Request

Listing C-5 addReservationComplex Request

```
<bea.club.ClubMed.addReservationComplex>
 <club>BAMBU</club>
 <resvData>
     <addr>
        <fname>Bill</fname>;
        <lname>Wales</lname>
        <street>Big Bungalow</street>
        <city>High End</city>
        <state>WA</state>
        <zip>99990</zip>
        <phone>123-123-3456</phone>
        <prices>
          <bea.club.pricesStruct>
            <date>2002-11-23</date>
            <adultFare>88</adultFare>
            <childFare>50</childFare>
          </bea.club.pricesStruct>
          <bea.club.pricesStruct>
            <date>2002-11-24</date>
            <adultFare>88</adultFare>
            <childFare>50</childFare>
          </bea.club.pricesStruct>
          <bea.club.pricesStruct>
            <date>2002-11-25</date>
            <adultFare>88</adultFare>
            <childFare>50</childFare>
          </bea.club.pricesStruct>
          <bea.club.pricesStruct>
            <date>2002-11-26</date>
            <adultFare>88</adultFare>
            <childFare>50</childFare>
          </bea.club.pricesStruct>
        </prices>
```

```
      </addr>
    <totalFare>5000.0</totalFare>
    <partyAdults>2</partyAdults>
    <partyChildren>2</partyChildren>
    <date>2001-09-21</date>
    <clubs>
      <item>BAMBU</item>
      <item>RIU PALACE</item>
    </clubs>
 </resvData>
</bea.club.ClubMed.addReservationComplex>
```

addReservationComplex Response

Listing C-6 addReservationComplex Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.addReservationComplex>
 <return>1000</return>
 <resvData>
    <addr>
      <fname>Bill</fname>
      <lname>Wales</lname>
      <street>Big Bungalow</street>
      <city>High End</city>
      <state>WA</state>
      <zip>99990</zip>
      <phone>123-123-3456</phone>
      <prices>
        <bea.club.pricesArray>
          <date>2002-11-23</date>
          <adultFare>88</adultFare>
          <childFare>50</childFare>
        </bea.club.pricesArray>
        <bea.club.pricesArray>
```

```
          <date>2002-11-24</date>
          <adultFare>88</adultFare>
          <childFare>50</childFare>
        </bea.club.pricesArray>
        <bea.club.pricesArray>
          <date>2002-11-25</date>
          <adultFare>88</adultFare>
          <childFare>50</childFare>
        </bea.club.pricesArray>
        <bea.club.pricesArray>
          <date>2002-11-26</date>
          <adultFare>88</adultFare>
          <childFare>50</childFare>
        </bea.club.pricesArray>
      </prices>
    </addr>
    <totalFare>5000.0</totalFare>
    <partyAdults>2</partyAdults>
    <partyChildren>2</partyChildren>
    <date>2001-09-21</date>
    <clubs>
      <item>BAMBU</item>
      <item>RIU PALACE</item>
    </clubs>
 </resvData>
</bea.club.ClubMed.addReservationComplex>
```

addReservationWithOut Request

Listing C-7 addReservationWithOut Request

```
<bea.club.ClubMed.addReservationWithOut>
 <club>BAMBU</club>
 <resvData>
    <fname>Bill</fname>;
    <lname>Wales</lname>
    <weekDate>Monday</weekDate>
    <street>Big Bungalow</street>
    <city>High End</city>
    <state>WA</state>
    <zip>99990</zip>
    <phone>123-123-3456</phone>
    <totalFare>5000.0</totalFare>
    <partyAdults>2</partyAdults>
    <partyChildren>2</partyChildren>
    <date>2001-09-21</date>
    <resv>0</resv>
 </resvData>
</bea.club.ClubMed.addReservationWithOut>
```

addReservationWithOut Response

Listing C-8 addReservationWithOut Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.addReservationWithOut>
 <num>1004</num>
</bea.club.ClubMed.addReservationWithOut>
```

cancelReservation Request

Listing C-9 cancelReservation Request

```
<bea.club.ClubMed.cancelReservation>
 <resv>1000</resv>
</bea.club.ClubMed.cancelReservation>
```

cancelReservation Response

Listing C-10 cancelReservation Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.cancelReservation>
 <return>false</return>
</bea.club.ClubMed.cancelReservation>
```

getClub Request

Listing C-11 GetClub Request

```
<bea.club.Agency.getClub>
</bea.club.Agency.getClub>
```

GetClubNames Request

Listing C-12 GetClubNames Request

```
<bea.club.ClubMed.getClubNames>
</bea.club.ClubMed.getClubNames>
```

GetClubNames Response

Listing C-13 GetClubNames Response

```
<?xml version="1.0"?>
<bea.club.ClubMed.getClubNames>
 <bea.club.clubNamesSeq>
    <return>BAMBU</return>
    <return>NAEBO</return>
    <return>RIO PALACE</return>
 </bea.club.clubNamesSeq>
</bea.club.ClubMed.getClubNames>
```

getClubNames4 Request

Listing C-14 getClubNames4 Request

```
<bea.club.ClubMed.getClubNames4>
</bea.club.ClubMed.getClubNames4>
```

getClubNames4 Response

Listing C-15 getClubNames4 Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.getClubNames4>
 <return>
    <item>BAMBU</item>
    <item>NAEBO</item>
    <item>RIU PALACE</item>
 </return>
</bea.club.ClubMed.getClubNames4>
```

getClubNamesWithRef Request

Listing C-16   GetClubNamesWithRef Request

```
<bea.club.ClubMed.getClubNames
   ref="IOR:000000000000001949444c3a6265612f636c75622f436c75624d6
5643a312e300000000000000001000000000000007800010200000000e313
7322e31392e32352e323131000c460000001c00564201000000022f0020200
000000400000001000001d738b66b92000000035649530300000000500040401 0f000000000
00000000000080000000056495300000000010000001400000000000100010000000000010
10900000000">
</bea.club.ClubMed.getClubNames>
```

GetClubPrices Request

Listing C-17   GetClubPrices Request

```
<bea.club.ClubMed.getClubPrices>
 <club>BAMBU</club>
</bea.club.ClubMed.getClubPrices>
```

GetClubPrices Response

Listing C-18   GetClubPrices Response

```
<?xml version="1.0"?>
<bea.club.ClubMed.getClubPrices>
 <bea.club.pricesSeq>
    <bea.club.pricesStruct>
      <date>2002-11-23</date>
      <adultFare>88</adultFare>
      <childFare>50</childFare>
    </bea.club.pricesStruct>
    <bea.club.pricesStruct>
      <date>2002-11-24</date>
      <adultFare>88</adultFare>
      <childFare>50</childFare>
    </bea.club.pricesStruct>
    <bea.club.pricesStruct>
      <date>2002-11-25</date>
      <adultFare>80</adultFare>
      <childFare>45</childFare>
    </bea.club.pricesStruct>
    <bea.club.pricesStruct>
      <date>2002-11-26</date>
      <adultFare>80</adultFare>
      <childFare>45</childFare>
    </bea.club.pricesStruct>
 </bea.club.pricesSeq>
</bea.club.ClubMed.getClubPrices>
```

GetClubPricesAsArray Request

Listing C-19 GetClubPricesAsArray Request

```
<bea.club.ClubMed.getClubPricesAsArray>
 <club>BAMBU</club>
</bea.club.ClubMed.getClubPricesAsArray>
```

GetClubPricesAsArray Response

Listing C-20 GetClubPricesAsArray Response

```
<?xml version="1.0"?>
<bea.club.ClubMed.getClubPricesAsArray>
```

```
 <bea.club.pricesArray>
    <bea.club.pricesStruct>
      <date>2002-11-23</date>
      <adultFare>88</adultFare>
      <childFare>50</childFare>
    </bea.club.pricesStruct>
    <bea.club.pricesStruct>
      <date>2002-11-24</date>
      <adultFare>88</adultFare>
      <childFare>50</childFare>
    </bea.club.pricesStruct>
    <bea.club.pricesStruct>
      <date>2002-11-25</date>
      <adultFare>80</adultFare>
      <childFare>45</childFare>
    </bea.club.pricesStruct>
    <bea.club.pricesStruct>
      <date>2002-11-26</date>
      <adultFare>80</adultFare>
      <childFare>45</childFare>
    </bea.club.pricesStruct>
 </bea.club.pricesArray>
</bea.club.ClubMed.getClubPricesAsArray>
```

GetReservation Request

Listing C-21 GetReservation Request

```
<bea.club.ClubMed.getReservation>
 <resv>1001</resv>
</bea.club.ClubMed.getReservation>
```

GetReservation Response

Listing C-22 GetReservation Response

```
<?xml version="1.0"?>
<bea.club.ClubMed.getReservation>
 <bea.club.resvStruct>
    <fname>Bill</fname>
    <lname>Wales</lname>
    <weekDate>Monday</weekDate>
    <address>Big Bungalow</address>
    <city>High End</city>
    <state>WA</state>
    <zip>99990</zip>
    <phone>123-123-3456</phone>
    <totalFare>5000.0</totalFare>
    <partyAdults>2</partyAdults>
    <partyChildren>2</partyChildren>
    <date>2001-09-21</date>
 </bea.club.resvStruct>
</bea.club.ClubMed.getReservation>
```

getReservationAsOut Request

Listing C-23 getReservationAsOut Request

```
<bea.club.ClubMed.getReservationAsOut>
 <resv>1000</resv>
</bea.club.ClubMed.getReservationAsOut>
```

getReservationAsOut Response

Listing C-24 getReservationAsOut Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.getReservationAsOut>
 <resvData>
    <addr>
      <fname>Bill</fname>
      <lname>Wales</lname>
      <street>Big Bungalow</street>
      <city>High End</city>
      <state>WA</state>
      <zip>99990</zip>
      <phone>123-123-3456</phone>
      <prices>
        <bea.club.pricesArray>
          <date>2002-11-23</date>
          <adultFare>88</adultFare>
          <childFare>50</childFare>
        </bea.club.pricesArray>
        <bea.club.pricesArray>
          <date>2002-11-24</date>
          <adultFare>88</adultFare>
          <childFare>50</childFare>
        </bea.club.pricesArray>
        <bea.club.pricesArray>
          <date>2002-11-25</date>
          <adultFare>88</adultFare>
          <childFare>50</childFare>
        </bea.club.pricesArray>
        <bea.club.pricesArray>
          <date>2002-11-26</date>
          <adultFare>88</adultFare>
          <childFare>50</childFare>
        </bea.club.pricesArray>
      </prices>
    </addr>
    <totalFare>5000.0</totalFare>
    <partyAdults>2</partyAdults>
    <partyChildren>2</partyChildren>
    <date>2001-09-21</date>
    <clubs>
      <item>BAMBU</item>
      <item>RIU PALACE</item>
    </clubs>
 </resvData>
</bea.club.ClubMed.getReservationAsOut>
```

setCancel Request

Listing C-25 setCancel Request

```
<bea.club.ClubMed.setCancel>
 <resv>1000</resv>
 <cancel>false</cancel>
</bea.club.ClubMed.setCancel>
```

setCancel Response

Listing C-26 setCancel Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.setCancel/>
```

setClubPrices Request

Listing C-27 setClubPrices Request

```
<bea.club.ClubMed.setClubPrices>
 <club>BAMBU</club>
 <prices>
    <bea.club.pricesStruct>
      <date>2002-11-23</date>
      <adultFare>88</adultFare>
      <childFare>50</childFare>
    </bea.club.pricesStruct>
    <bea.club.pricesStruct>
      <date>2002-11-24</date>
      <adultFare>88</adultFare>
      <childFare>50</childFare>
    </bea.club.pricesStruct>
 </prices>
</bea.club.ClubMed.setClubPrices>
```

setClubPrices Response

Listing C-28 setClubPrices Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.setClubPrices/>
```

setClubPricesAsArray Request

Listing C-29 setClubPricesAsArray Request

```
<bea.club.ClubMed.setClubPricesAsArray>
 <club>BAMBU</club>
 <prices>
    <bea.club.pricesStruct>
      <date>2002-11-23</date>
      <adultFare>88</adultFare>
      <childFare>50</childFare>
    </bea.club.pricesStruct>
    <bea.club.pricesStruct>
      <date>2002-11-24</date>
      <adultFare>88</adultFare>
      <childFare>50</childFare>
    </bea.club.pricesStruct>
    <bea.club.pricesStruct>
      <date>2002-11-24</date>
      <adultFare>88</adultFare>
      <childFare>50</childFare>
    </bea.club.pricesStruct>
    <bea.club.pricesStruct>
      <date>2002-11-24</date>
      <adultFare>88</adultFare>
      <childFare>50</childFare>
    </bea.club.pricesStruct>
 </prices>
</bea.club.ClubMed.setClubPricesAsArray>
```

setClubPricesAsArray Response

Listing C-30 setClubPricesAsArray Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.setClubPricesAsArray/>
```

setStatus Request

Listing C-31 setStatus Request

```
<bea.club.ClubMed.setStatus>
 <resv>1000</resv>
 <cod>CANCELLED</cod>
</bea.club.ClubMed.setStatus>
```

setStatus Response

Listing C-32 setStatus Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.setStatus/>
```

status Request

Listing C-33 status Request

```
<bea.club.ClubMed.status>
 <resv>1000</resv>
</bea.club.ClubMed.status>
```

status Response

Listing C-34 status Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.status>
 <return>PROCESSED</return>
</bea.club.ClubMed.status>
```

testOneDim Request

Listing C-35 testOneDim Request

```
<bea.club.ClubMed.testOneDim>
 <par>
    <item>1</item>
    <item>2</item>
    <item>3</item>
 </par>
</bea.club.ClubMed.testOneDim>
```

testOneDim Response

Listing C-36 testOneDim Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.testOneDim>
 <par>
    <item>1</item>
    <item>2</item>
    <item>3</item>
 </par>
</bea.club.ClubMed.testOneDim>
```

testTwoDim Request

Listing C-37 testTwoDim Request

```
<bea.club.ClubMed.testTwoDim>
 <par>
    <bea.club.twoDim_row>
      <item>one</item>
      <item>two</item>
      <item>three</item>
      <item>four</item>
      <item>five</item>
    </bea.club.twoDim_row>
    <bea.club.twoDim_row>
      <item>one</item>
      <item>two</item>
      <item>three</item>
      <item>four</item>
      <item>five</item>
    </bea.club.twoDim_row>
    <bea.club.twoDim_row>
```

```
      <item>one</item>
      <item>two</item>
      <item>three</item>
      <item>four</item>
      <item>five</item>
    </bea.club.twoDim_row>
 </par>
</bea.club.ClubMed.testTwoDim>
```

testTwoDim Response

Listing C-38 testTwoDim Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.testTwoDim>
 <par>
    <bea.club.twoDim_row>
      <item>one</item>
      <item>two</item>
      <item>three</item>
      <item>four</item>
      <item>five</item>
    </bea.club.twoDim_row>
    <bea.club.twoDim_row>
      <item>one</item>
      <item>two</item>
      <item>three</item>
      <item>four</item>
      <item>five</item>
    </bea.club.twoDim_row>
    <bea.club.twoDim_row>
      <item>one</item>
      <item>two</item>
      <item>three</item>
      <item>four</item>
      <item>five</item>
    </bea.club.twoDim_row>
 </par>
</bea.club.ClubMed.testTwoDim>
```

testTypes Request

Listing C-39   testTypes Request

```
<bea.club.ClubMed.testTypes>
 <all>
    <t1>string10</t1>
    <t2>1000</t2>
    <t3>1000999</t3>
    <t4>12345</t4>
    <t5>99999999</t5>
    <t6>1.234</t6>
    <t7>98765.1234</t7>
    <t8>c</t8>
    <t9>false</t9>
    <t10>34</t10>
    <t11>w</t11>
    <t14>123456789012899</t14>
    <t15>12345678901289999</t15>
 </all>
</bea.club.ClubMed.testTypes>
```

testTypes Response

Listing C-40   testTypes Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<bea.club.ClubMed.testTypes>
```

```
 <all>
    <t1>string10</t1>
    <t2>1000</t2>
    <t3>1000999</t3>
    <t4>12345</t4>
    <t5>99999999</t5>
    <t6>1.234</t6>
    <t7>98765.1234</t7>
    <t8>c</t8>
    <t9>false</t9>
    <t10>34</t10>
    <t11>w</t11>
    <t14>123456789012899</t14>
    <t15>12345678901289999</t15>
 </all>
</bea.club.ClubMed.testTypes>
```

testUnion1-2 Request

Listing C-41 testUnion1-2 Request

```
<bea.club.ClubMed.testUnion1>
   <moon>
      <case.2>123456</case.2>
   </moon>
</bea.club.ClubMed.testUnion1>
```

testUnion2-1 Request

Listing C-42 testUnion2-1 Request

```
<bea.club.ClubMed.testUnion2>
   <moon>
```

```
      <case.1>blah</case.1>
   </moon>
</bea.club.ClubMed.testUnion2>
```

testUnion2-3 Request

Listing C-43   testUnion2-3 Request

```
<bea.club.ClubMed.testUnion2>
   <moon>
      <case.3>
         <t1>string10</t1>
         <t2>1000</t2>
         <t3>1000999</t3>
         <t4>12345</t4>
         <t5>99999999</t5>
         <t6>1.234</t6>
         <t7>98765.1234</t7>
         <t8>c</t8>
         <t9>false</t9>
         <t10>34</t10>
         <t11>w</t11>
         <t14>123456789012899</t14>
         <t15>12345678901289999</t15>
      </case.3>
   </moon>
</bea.club.ClubMed.testUnion2>
```

testUnion3-PROCESSED Request

Listing C-44 testUnion3-PROCESSED Request

```
<bea.club.ClubMed.testUnion3>
   <moon>
      <case.PROCESSED>123456</case.PROCESSED>
   </moon>
</bea.club.ClubMed.testUnion3>
```

testUnion4-1 Request

Listing C-45 testUnion4-1 Request

```
<bea.club.ClubMed.testUnion4>
   <moon>
      <case.1>blah</case.1>
   </moon>
</bea.club.ClubMed.testUnion4>
```

testUnion4-default Request

Listing C-46 testUnion4-default Request

```
<bea.club.ClubMed.testUnion4>
   <moon>
      <case.default>true</case.default>
   </moon>
</bea.club.ClubMed.testUnion4>
```

Definitions for ClubMed Object

The BEA WebLogic Adapter for CORBA provides the sample ClubMed object to enable you to test the adapter.

JacORB.properties File

The following listing reflects the JacORB.properties file after you have edited it according to the instructions in the *BEA WebLogic Adapter for CORBA Installation and Configuration Guide*.

Listing C-47 JacORB.properties File

```
##
## JacORB configuration options
##

########################################
#                                      #
#   Initial references configuration   #
#                                      #
########################################

#
# URLs where IORs are stored (used in orb.resolve_initial_service())
# DO EDIT these! (Only those that you are planning to use,
# of course ;-).
#
# The ORBInitRef references are created on ORB startup time. In the
# cases of the services themselves, this may lead to exceptions being
# displayed (because the services aren't up yet). These exceptions
# are handled properly and cause no harm!

#ORBInitRef.NameService=corbaloc::160.45.110.41:38693/StandardNS/NameServer%2D
POA/_root
ORBInitRef.NameService=file:/d:/JacORB1_4_beta4/bea/ns_ref.txt
#ORBInitRef.NameService=http://www.x.y.z/~user/NS_Ref
#ORBInitRef.TradingService=http://www.x.y.z/~user/TraderRef

# JacORB-specific URLs
jacorb.ProxyServerURL=http://www.x.y.z/~user/Appligator_Ref


##################################
#                                #
#   ORB version number output    #
#                                #
##################################

# if on, the ORB's version number is printed
# any time the ORB is initialized
jacorb.orb.print_version=on

##################################
#                                #
#   Debug output configuration   #
#                                #
##################################

# use (java) jacorb.util.CAD to generate an appropriate
# verbosity level
# 0 = off
# 1 = important messages and exceptions
# 2 = informational messages and exceptions
# >= 3 = debug-level output (may confuse the unaware user :-)
jacorb.verbosity=1

# where does output go? Terminal is default
#jacorb.logfile=LOGFILEPATH

# hexdump outgoing messages
jacorb.debug.dump_outgoing_messages=off

# hexdump incoming messages
jacorb.debug.dump_incoming_messages=off


##################################################
#                                                #
#    WARNING: The following properties should    #
#    only be edited by the expert user. They     #
#    can be left untouched in most cases!        #
#                                                #
##################################################



################################
#                              #
#   Basic ORB Configuration    #
#                              #
################################

# the GIOP minor version number to use for newly created IORs
jacorb.giop_minor_version=2
```

```
# number of retries if connection cannot directly be established
jacorb.retries=5

# how many msecs. do we wait between retries
jacorb.retry_interval=500

# size of network buffers for outgoing messages
jacorb.outbuf_size=2048

# log2 of maximum buffer size managed by the internal
# buffer manager.
#
# This is NOT the maximum buffer size that
# can be used, but just the largest size of buffers that
# will be kept and managed. This value will be added to
# an internal constant of 5, so the real value in bytes
# is 2**(5+maxManagedBufSize-1). You only need to increase this
# value if you are dealing with LOTS of LARGE data structures.
# You may decrease it to make the buffer manager release large
# buffers immediately rather than keeping them for later
# reuse.
jacorb.maxManagedBufSize=18

# client-side timeout, set no non-zero to stop blocking
# after so many msecs.
#jacorb.connection.client_timeout=0

# max time a server keeps a connection open if nothing happens
#jacorb.connection.server_timeout=10000

#jacorb.reference_caching=off

#
# The following property specifies the class which is used for
# reference caching. WeakHashtable uses WeakReferences, so entries
# get gc'ed if only the Hashtable has a reference to them. This
# is useful if you have many references to short-living non-persistent
# CORBA objects. It is only available for java 1.2 and above.
#
# On the other hand the standard Hashtable keeps the references until
# they are explicitly deleted by calling _release(). This is useful
# for persistent and long-living CORBA objects.
#
#jacorb.hashtable_class=org.jacorb.util.WeakHashtable
#
jacorb.hashtable_class=java.util.Hashtable

# use GIOP 1.2 byte order markers (since CORBA 2.4-5)
```

```
jacorb.use_bom=off

# add additional IIOP 1.0 profiles even if we are using IIOP 1.2
jacorb.giop.add_1_0_profiles=off

###########################################
#                                         #
#         Socket Factories                #
#                                         #
###########################################

# A factory design pattern is used for the creation of sockets and server
# sockets.
# The jacorb.net.socket_factory property can be used to configure
# a socket factory that must implement the operations defined in the
# interface org.jacorb.orb.factory.SocketFactory.
# The jacorb.net.server_socket_factory property can be used to configure a
# server socket factory that must implement the operations defined in the
# interface org.jacorb.orb.factory.ServerSocketFactory.
#
#jacorb.net.socket_factory=org.jacorb.orb.factory.DefaultSocketFactory
#jacorb.net.server_socket_factory=org.jacorb.orb.factory.DefaultServerSocketFa
ctory
#
# An additional socket factory is supported that allows for the configuration
# of maximum and minimum port numbers that can be used. This can be used to
# enable firewall traversal via a fixed port range. To use this socket factory
# configure the following two properties.
#
#jacorb.net.socket_factory.port.min
#jacorb.net.socket_factory.port.max

###########################################
#                                         #
#         BiDirectional GIOP              #
#                                         #
###########################################

# uncomment this initializer if you want to use BiDirectional GIOP

#org.omg.PortableInterceptor.ORBInitializerClass.bidir_init=org.jacorb.orb.con
nection.BiDirConnectionInitializer


###########################################
#                                         #
#       Proxy address in IOR              #
#                                         #
###########################################
```

```
#
# with these two properties it is possible to
# tell the ORB what IP/port IORs should contain,
# if the ServerSockets IP/port can't be used
# (e.g. for traffic through a firewall).
#
# WARNING: this is just "dumb" replacing, so you
# have to take care of your configuration!
#

#jacorb.ior_proxy_host=1.2.3.4
#jacorb.ior_proxy_port=4711


###########################################
#                                         #
#   The Object Adapter Internet Address   #
#                                         #
###########################################

# IP address on multi-homed host (this gets encoded in
# object references). NOTE: Addresses like 127.0.0.X
# will only be accessible from the same machine!
#OAIAddr=1.2.3.4
#OAPort=4711

################################
#                              #
#   Appligator Configuration   #
#                              #
################################
# if your applets don't need appligator, switch this off
jacorb.use_appligator=off



############################
#                          #
#   Default Interceptors   #
#   Please leave them in!  #
#                          #
############################
org.omg.PortableInterceptor.ORBInitializerClass.standard_init=org.jacorb.orb.s
tandardInterceptors.IORInterceptorInitializer



###############################################
```

```
#                                              #
#   Implementation Repository Configuration    #
#                                              #
################################################
# Switch off to avoid contacting the ImR on every server start-up
jacorb.use_imr=off

# if set to "on", servers that don't already have an entry on their
# first call to the imr, will get automatically registered. Otherwise,
# an UnknownServer exception is thrown.
jacorb.imr.allow_auto_register=off

# if set to "on", the imr will try to "ping" every object reference,
# that it is going to return. If the reference is not alive, TRANSIENT
# is thrown.
jacorb.imr.check_object_liveness=off

ORBInitRef.ImplementationRepository=http://www.x.y.z/~user/ImR_Ref

jacorb.imr.table_file=Z:\table.dat
jacorb.imr.backup_file=z:\backup.dat
jacorb.imr.ior_file=/home/bwana/brose/public_html/ImR_Ref
jacorb.imr.timeout=
jacorb.imr.no_of_poas=
jacorb.imr.no_of_servers=

# how many millis should the imr wait, until a connection from an
# application client is terminated. Default is 2000.
jacorb.imr.connection_timeout=2000

# the implementation name, should be set to a different
# name in the code of persistent servers
jacorb.implname=StandardImplName

#
# This is supposed to be a generic startup string for everything
# that calls Runtime.exec(). Might be replaced by jaco[.bat].
#
jacorb.java_exec=java -Dorg.omg.CORBA.ORBClass=org.jacorb.orb.ORB
-Dorg.omg.CORBA.ORBSingletonClass=org.jacorb.orb.ORBSingleton



#########################
#                       #
#   SSL Configuration   #
#                       #
#########################
```

```
#
# The port number used by SSL, will be dynamically assigned
# by default
#

#OASSLPort=4711

# This interceptor must be set if programs need access to
# certificates using the CORBA Security API, SSL works also
# without this interceptor

#org.omg.PortableInterceptor.ORBInitializerClass.ForwardInit=org.jacorb.securi
ty.ssl.SecurityServiceInitializer


# qualified classname of access decision object
jacorb.security.access_decision=org.jacorb.security.level2.AccessDecisionImpl

# list of qualified classnames of principal authenticator objects,
# separated by commas (no whitespaces!). The first entry (that can
# be successfully created) will be available through the
# principal_authenticator property.
jacorb.security.principal_authenticator=org.jacorb.security.level2.PrincipalAu
thenticatorImpl

# the qualified classname of the ssl socket factory class
#jacorb.ssl.socket_factory=org.jacorb.security.ssl.sun_jsse.SSLSocketFactory
jacorb.ssl.socket_factory=org.jacorb.security.ssl.iaik.SSLSocketFactory

# the qualified classname of the ssl server socket factory class
#jacorb.ssl.server_socket_factory=org.jacorb.security.ssl.sun_jsse.SSLServerSo
cketFactory
jacorb.ssl.server_socket_factory=org.jacorb.security.ssl.iaik.SSLServerSocketF
actory

# exchange ssl client server roles to enforce client authentication, but
# attention: this causes problems with peers that not prepared to handle
# this role change
jacorb.security.change_ssl_roles=off

# IIOP/SSL parameters (numbers are hex values, without the leading "0x"):
# NoProtection = 1
# EstablishTrustInClient = 40
# EstablishTrustInTarget = 20
# mutual authentication = 60
# please see the programming guide for more explanation

jacorb.security.support_ssl=off
jacorb.security.ssl.client.supported_options=0
jacorb.security.ssl.client.required_options=0

jacorb.security.ssl.server.supported_options=0
jacorb.security.ssl.server.required_options=0

#
# If set, the following two values will be placed in the IOR, if
# "corbaloc:ssliop" ssliop.
#
# If not set, only EstablishTrustInTarget is used for both supported
# and required options. EstablishTrustInClient is not set, and the
# rest of the Association Options aren’t currently used anyway.
#jacorb.security.ssl.corbaloc_ssliop.supported_options=0
#jacorb.security.ssl.corbaloc_ssliop.required_options=0

# The name and location of the keystore. This may be absolute or
# relative to the home directory.
#
# NOTE (for Sun JSSE users): The "javax.net.ssl.trustStore[Password]"
# properties don’t seem to take effect, so you may want to add trusted
# certificates to "normal" keystores. In this case, please set the
# property "jacorb.security.jsse.trustees_from_ks"is to "on", so trusted
# certificates are taken from the keystore instead of a dedicated
# truststore.
jacorb.security.keystore=
jacorb.security.keystore_password=


#
# IAIK specific settings
#

# files with public key certs of trusted CAs
#
# WARNING: If no CA certs are present, the IAIK chain verifier will
# accept ALL otherwise valid chains!
#
jacorb.security.trustees=

# the name of the default key alias to look up in the keystore
jacorb.security.default_user=
jacorb.security.default_password=

# have iaiks ssl classes print debug output to stdout
jacorb.security.iaik_debug=off

#
# Sun JSSE specific settings
```

```
#
# Use the keystore to take trusted certs from.
jacorb.security.jsse.trustees_from_ks=off

# A comma-separated (no whitespaces!) list of cipher suite names. See
# the JSSE docs on how to obtain the correct cipher suite strings
jacorb.security.ssl.server.cipher_suites=
jacorb.security.ssl.client.cipher_suites=

########################
#                      #
#   POA Configuration  #
#                      #
########################

# displays a GUI monitoring tool for servers
jacorb.poa.monitoring=off

# thread pool configuration for request processing
jacorb.poa.thread_pool_max=20
jacorb.poa.thread_pool_min=5

# if set, request processing threads in the POA
# will run at this priority. If not set or invalid,
# MAX_PRIORITY will be used.
#jacorb.poa.thread_priority=

# size of the request queue, clients will receive Corba.TRANSIENT
# exceptions if load exceeds this limit
jacorb.poa.queue_max=100



########################################
#                                      #
#   Trader configuration, please see   #
#   src/trading/README.PROPERTIES for  #
#   explanation                        #
#                                      #
########################################

jtrader.util.max_threads=10
jtrader.util.min_threads=1
jtrader.util.query_timeout=5000
jtrader.impl.cache_max=100

# boolean values, e.g. true / false
#jtrader.modifiable_properties=
#jtrader.dynamic_properties=
#jtrader.proxy_offers=

jtrader.debug=false
jtrader.debug_verbosity=3

#integer values
jtrader.def_search_card=
jtrader.max_search_card=
jtrader.def_match_card=
jtrader.max_match_card=
jtrader.def_return_card=
jtrader.max_return_card=
jtrader.max_list=
jtrader.def_hop_count=
jtrader.max_hop_count=

#FollowOptions
#always=2
#if_no_local=1
#local_only=0
jtrader.def_follow_policy=
jtrader.max_follow_policy=
jtrader.max_link_follow_policy=

# any other custom properties can be added here.
# These are available through the API (call
# jacorb.orb.Environment.getProperty())
```

Build XML

This file enables you to build the sample JacORB application referred to in the *BEA WebLogic Adapter for CORBA Installation and Configuration Guide*.

Listing C-48 Build XML

```
<?xml version="1.0"?>

<project name="bea" default="all" basedir="../..">

 <!-- ==================================================== -->
 <!--             build file                               -->
 <!-- ==================================================== -->

 <target name="init">
```

```
    <property name="name" value="club"/>
    <property name="dirs.base" value="${basedir}"/>
    <property name="classdir" value="${dirs.base}/classes"/>
    <property name="lib" value="${dirs.base}/lib"/>
    <property name="include" value="${dirs.base}/idl"/>
    <property name="idlflags" value="-I${include}/omg -ir -d
${dirs.base}/bea/${name}/generated"/>
  </target>

  <target name="all" depends="init,idl">
    <jived srcdir="${dirs.base}/bea/${name}/generated"
                destdir="${classdir}"
           includes="**/*.java"
            />

    <jived srcdir="${dirs.base}/bea/${name}"
                destdir="${classdir}"
           includes="*.java"
            />
   </target>

  <target name="idl" depends="init">
        <java classname="org.jacorb.idl.parser"
               fork="yes"
               classpath="${lib}/idl.jar;${java.class.path}">

           <are line="${divulges}
             ${dirs.base}/bea/${name}/clubmed.idl"/>
        </java>
    </target>


    <target name="clean" depends="init">
            <delete dir="${cloister}/bea/${name}"/>
            <delete dir="${dirs.base}/bea/${name}/generated"/>
    </target>

</project>
```

ClubMed.IDL File

The `clubmed.idl` file displays the Interface Repository in IDL.

Listing C-49

```
// ClubMed.idl
module bea
{
module club
{
 struct typesStruct
 {
    string<10>       t1;
    short            t2; /* 16 bit */
    long             t3; /* 32 bit */
    unsigned short   t4;
    unsigned long    t5;
    float            t6;
    double           t7;
    char             t8;
    boolean          t9;
    octet            t10; /* byte */
    wchar            t11;
   /* restring          t12; */
   /* wstring<10>      t13; */
    long long        t14; /* 64 bit */
    unsigned long long t15;
   /* long double      t16; */
   /* fixed<5,2>       t17; */
};


 typedef long   oneDim[3];
 typedef string twoDim[3][5];

 struct pricesStruct
 {
    string date;
    short adultFare;
    short childFare;
};

 typedef sequence <string,4> clubNamesSeq4;

 typedef sequence <pricesStruct> pricesSeq;

 typedef sequence <string> clubNamesSeq;

 typedef long resvNo;

 typedef pricesStruct pricesArray[4];
 struct resvStruct
 {
    string fname;
    string lname;
    string weekDate;
    string street;
    string city;
    string state;
    string zip;
    string phone;
    string totalFare;
    short partyAdults;
    short partyChildren;
    string date;
    resvNo resv;
};

 struct address
 {
    string fname;
    string lname;
    string street;
    string city;
    string state;
    string zip;
    string phone;
    pricesArray prices;
};

 struct resv Complex Struct
 {
    address addr;
    string totalFare;
    short   partyAdults;
    short   partyChildren;
    string date;
    clubNamesSeq clubs;
};

 enum code { OPENED, PROCESSED, CANCELED};

 union myUnion4 switch(long) {
    case 1: string name;
    case 2: long   num;
    case 3: typesStruct str;
    default: boolean b;
};
```

```
  union myUnion1 switch(char)  {
    case '1': string name;
    case '2': long   num;
    case '3': typesStruct str;
  };

  union myUnion2 switch(short) {
    case 1: string name;
    case 2: long   num;
    case 3: typesStruct str;
  };

  union myUnion3 switch(code) {
    case OPENED:     string name;
    case PROCESSED: long   num;
    case CANCELED:  typesStruct str;
  };

  interface ClubMed
  {
    exception ClubException
    {
      string reason;
    };

    void                  setClubPricesAsArray(in string club, in pricesArray
prices);
    void                  setClubPrices(in string club, in pricesSeq prices);
    pricesSeq             getClubPrices(in string club);
    pricesArray           getClubPricesAsArray(in string club);
    resvNo               addReservation(in string club, in resvStruct resvData);
    void                  addReservationWithOut(in string club, in resvStruct
resvData, out long num);
   resvStruct          getReservation(in resvNo resv) raises ( ClubException );
   void               getReservationAsOut(in resvNo resv, out resvComplexStruct
resvData) raises ( ClubException );
    resvNo                addGetReservation(in string club, inout resvStruct
resvData);
    clubNamesSeq          getClubNames();
    clubNamesSeq4         getClubNames4();
    resvNo                addReservationComplex(in string club, inout
resvComplexStruct resvData);
    code                  status(in resvNo resv) raises ( ClubException );
   boolean            cancelReservation (in resvNo resv) raises ( ClubException
);
    void                  setStatus(in resvNo resv, in code cod) raises (
ClubException );
    void                 setCancel(in resvNo resv, in boolean cancel) raises (
ClubException );
```

```
    void                  testTypes(inout typesStruct all);
    void                  testOneDim(inout oneDim par);
    void                  testTwoDim(inout twoDim par);
void                  testUnion1(inout myUnion1 moon);
void                  testUnion2(inout myUnion2 moon);
void                  testUnion3(inout myUnion3 moon);
void                  testUnion4(inout myUnion4 moon);
 };

 interface Agency
 {
   ClubMed getClub();
 };

};
};
```

ClubServer.java.jacorb File

The `ClubServer.java.jacorb` file displays the Interface Repository in Java.

Listing C-50 ClubServer.java.jacorb File

```
// The package containing our stubs.
package bea.club;

import java.util.*;
import java.io.*;

// HelloServer will use the naming service.
import org.omg.CosNaming.*;
// All CORBA applications need these classes.
import org.omg.CORBA.*;

public class ClubServer
{
  public static void main(String args[]) {

    if (args == null || args.length == 0) {
      System.out.println("USAGE: java ClubServer (-club | -agency) [objref 
path]");
      return;
    }
    boolean club;
    if (args[0].equals("-club"))
      club = true;
    else if (args[0].equals("-agency"))
      club = false;
    else {
      System.out.println("USAGE: java ClubServer (-club | -agency) [objref
path]");
      return;
    }

    // Create and initialize the ORB
    System.out.println("main: ORB.init(args,props)...");
    ORB orb = ORB.init(args, null);
    System.out.println("main: ORB.init(args,props) done.");

    try {
      System.out.println("try...");

      // Get a reference to the root POA
      System.out.println("main:
POAHelper.narrow(orb.resolve_initial_references(\"RootPOA\"))...");
      org.omg.PortableServer.POA poa =
org.omg.PortableServer.POAHelper.narrow(orb.resolve_initial_references("RootPO
A"));
      System.out.println("main:
POAHelper.narrow(orb.resolve_initial_references(\"RootPOA\")) done.");

      // Activate the POA
      System.out.println("main: poa.the_POAManager().activate()...");
      poa.the_POAManager().activate();
      System.out.println("main: poa.the_POAManager().activate() done.");

      // Get a ClubServant object
      org.omg.CORBA.Object o = null;
      if (club) {
        System.out.println("main: poa.servant_to_reference(new
ClubsServant())...");
        o = poa.servant_to_reference(new ClubsServant());
        System.out.println("main: poa.servant_to_reference(new ClubsServant())
done.");
      }
      else {
        System.out.println("main: poa.servant_to_reference(new
AgencyServant()...");
        o = poa.servant_to_reference(new AgencyServant());
        System.out.println("main: poa.servant_to_reference(new AgencyServant()
done.");
      }
      if (args.length == 2) {
        String oRef = orb.object_to_string(o);
        FileWriter ff = new FileWriter(args[1]);
        ff.write(oRef, 0, oRef.length());
        ff.close();
      }
else {
        // use the naming service
        // get reference to Naming Context
        System.out.println("main:
NamingContextExtHelper.narrow(orb.resolve_initial_references(\"NameService\"))
...");
        NamingContextExt nc =
NamingContextExtHelper.narrow(orb.resolve_initial_references("NameService"));
        System.out.println("main:
NamingContextExtHelper.narrow(orb.resolve_initial_references(\"NameService\"))
done.");

        // bind our object to bea.clubmed in Naming Context
        System.out.println("main: nc.rebind( nc.to_name(\"bea.clubmed\"),
o)...");
        nc.rebind( nc.to_name("bea.clubmed"), o);
        System.out.println("main: nc.rebind( nc.to_name(\"bea.clubmed\"), o)
done.");
      }

      System.out.println("main: orb.run()...");
      orb.run();
      System.out.println("main: orb.run() done.");

    } catch(Exception e) {
      System.err.println("ERROR: " + e);
      e.printStackTrace(System.out);
    }
  }
}


class AgencyServant extends AgencyPOA
{
  private ClubsServant club = new ClubsServant();

  public ClubMed getClub() {
    System.out.println("method getClub()  called");
    try {
    return ClubMedHelper.narrow(_default_POA().servant_to_reference(club));
    } catch (Exception e) {
      e.printStackTrace();
    }
    return null;
  }
}

class ClubsServant extends ClubMedPOA
{
  private int resNum = 999;
  private Map map = new HashMap();
  private Map mapComplex = new HashMap();
  private int resNumComplex = 999;

  public void setClubPrices(String club, pricesStruct[] prices) {
    System.out.println("method setClubPrices called");
    System.out.println("club " + club);
    for (int i = 0; i < prices.length; i++)
      System.out.println(prices[i].date + " " + prices[i].adultFare + " " +
prices[i].childFare);
  }

  public void setClubPricesAsArray(String club, pricesStruct[] prices) {
    System.out.println("method setClubPricesAsArray called");
    System.out.println("club " + club);
    System.out.println(prices[0].date + " " + prices[0].adultFare + " " +
prices[0].childFare);
    System.out.println(prices[1].date + " " + prices[1].adultFare + " " +
prices[1].childFare);
    System.out.println(prices[2].date + " " + prices[2].adultFare + " " +
prices[2].childFare);
    System.out.println(prices[3].date + " " + prices[3].adultFare + " " +
prices[3].childFare);
  }

  public pricesStruct[] getClubPrices (String club) {
    System.out.println("method getClubPrices called");
    pricesStruct[] prices = new pricesStruct[4];
    prices[0] = new pricesStruct("2002-11-23", (short)88, (short)50);
    prices[1] = new pricesStruct("2002-11-24", (short)88, (short)50);
    prices[2] = new pricesStruct("2002-11-25", (short)80, (short)45);
    prices[3] = new pricesStruct("2002-11-26", (short)80, (short)45);
    return prices;
  }

  public pricesStruct[]  getClubPricesAsArray(String club) {
    System.out.println("method getClubPricesAsArray called");
    pricesStruct[] prices = new pricesStruct[4];
    prices[0] = new pricesStruct("2002-11-23", (short)88, (short)50);
    prices[1] = new pricesStruct("2002-11-24", (short)88, (short)50);
    prices[2] = new pricesStruct("2002-11-25", (short)80, (short)45);
    prices[3] = new pricesStruct("2002-11-26", (short)80, (short)45);
    return prices;
  }

  public String[] getClubNames() {
    System.out.println("method getClubNames called");
    return new String[] {"BAMBU", "NAEBO", "RIU PALACE" };
  }

  public String[] getClubNames4() {
    System.out.println("method getClubNames4 called");
    return new String[] {"BAMBU", "NAEBO", "RIU PALACE" };
  }

  public int addReservation (String club, resvStruct resvData) {
    System.out.println("method addReservation called");
    System.out.println("club: " + club);
    System.out.println(resvData.street);
    System.out.println(resvData.city);
    System.out.println(resvData.date);
    System.out.println(resvData.fname);
    System.out.println(resvData.lname);
    System.out.println(resvData.partyAdults);
    System.out.println(resvData.partyChildren);
    System.out.println(resvData.phone);
    System.out.println(resvData.weekDate);
    System.out.println(resvData.zip);
    System.out.println(resvData.totalFare);
    map.put(new Integer(++resNum), resvData);
    System.out.println("returning " + resNum);
    return resNum;
  }

  public void addReservationWithOut (String club, resvStruct resvData,
org.omg.CORBA.IntHolder num) {
    System.out.println("method addReservationWithOut called");
    System.out.println("club: " + club);
    System.out.println(resvData.street);
    System.out.println(resvData.city);
    System.out.println(resvData.date);
    System.out.println(resvData.fname);
    System.out.println(resvData.lname);
    System.out.println(resvData.partyAdults);
    System.out.println(resvData.partyChildren);
    System.out.println(resvData.phone);
    System.out.println(resvData.weekDate);
    System.out.println(resvData.zip);
    System.out.println(resvData.totalFare);
    map.put(new Integer(++resNum), resvData);
    num.value = resNum;
  }

  public resvStruct getReservation (int resv) {
    System.out.println("method getReservation called");
    resvStruct resvData = (resvStruct)map.get(new Integer(resv));
    System.out.println("Returning: " + resvData);
    return resvData;
  }

  public int addGetReservation (String club, resvStructHolder resvData) {
    System.out.println("method addGetReservation called");
    resvStruct data = resvData.value;
    System.out.println(club);
    System.out.println(data.street);
    System.out.println(data.city);
    System.out.println(data.date);
    System.out.println(data.fname);
    System.out.println(data.lname);
    System.out.println(data.partyAdults);
    System.out.println(data.partyChildren);
    System.out.println(data.phone);
    System.out.println(data.weekDate);
    System.out.println(data.zip);
    System.out.println(data.totalFare);
    System.out.println(data.resv);
    map.put(new Integer(++resNum), data);
    return resNum;
  }


  public boolean cancelReservation (int resv) {
    System.out.println("method cancelReservation called");
    System.out.println("returning false");
    return false;
  }

  public int addReservationComplex(java.lang.String club,
resvComplexStructHolder resvData) {
    System.out.println("method addReservationComplex called");
    mapComplex.put(new Integer(++resNumComplex), resvData.value);
    System.out.println("Returning: " + resNumComplex);
    return resNumComplex;
  }

  public bea.club.code status (int resv) {
    System.out.println("method status called");
    System.out.println("reserv: " + resv + " returning: " +
bea.club.code.PROCESSED);
    return bea.club.code.PROCESSED;
```

```
}

public void  setStatus(int resv, bea.club.code cod) {
  System.out.println("method setStatus called");
  System.out.println("reserv: " + resv + " cod: " + cod);
}

public void  setCancel(int resv, boolean cancel) {
  System.out.println("method setCancel called");
  System.out.println("reserv: " + resv + " cancel: " + cancel);
}

public void  testTypes(typesStructHolder all) {
  System.out.println("method testTypes called");
  typesStruct t = all.value;
  System.out.println(t.t1);
  System.out.println(t.t2);
  System.out.println(t.t3);
  System.out.println(t.t4);
  System.out.println(t.t5);
  System.out.println(t.t6);
  System.out.println(t.t7);
  System.out.println(t.t8);
  System.out.println(t.t9);
  System.out.println(t.t10);
  System.out.println(t.t11);
  System.out.println(t.t14);
  System.out.println(t.t15);
}

public void testOneDim(oneDimHolder par) {
  System.out.println("method testOneDim called");
  int[] one = par.value;
  for (int i = 1; i <= one.length; i++) {
    System.out.println("item " + i + " " + one[i-1]);
  }
}

public void testTwoDim(twoDimHolder par) {
  System.out.println("method testTwoDim called");
  String[][] two = par.value;
  for (int i = 1; i <= two.length; i++) {
    System.out.println("row " + i);
    for (int j = 1; j <= two[i-1].length; j++)
      System.out.println("item " + j + " " + two[i-1][j-1]);
  }
}

public void testUnion1(bea.club.myUnion1Holder moon) {
    System.out.println("method testUnion1 called");
  }

  public void testUnion2(bea.club.myUnion2Holder moon) {
    System.out.println("method testUnion2 called");
  }

  public void testUnion3(bea.club.myUnion3Holder moon) {
    System.out.println("method testUnion3 called");
  }

  public void testUnion4(bea.club.myUnion4Holder moon) {
    System.out.println("method testUnion4 called");
  }


}
```

Index