BEA WebLogic Adapter for SAP®

User Guide

Release 7.0 with Service Pack 1
Document Date: November 2002

BEA WebLogic Adapter for SAP User Guide

Part Number	Date	Release
N/A	November 2002	7.0 with Service Pack 1

Table of Contents

About This Document

The BEA WebLogic Adapter for SAP is organized as follows:

- Chapter 1, “Introducing the BEA WebLogic Adapter for SAP,” introduces the BEA WebLogic Adapter for SAP and describes SAP business objects and WebLogic Integration.
- Chapter 2, “Creating Schemas for SAP Business Objects,” describes how to use the BEA Application Explorer to generate schemas for your SAP business objects.
- Chapter 3, “Configuring Application View Events and Services,” describes how to configure application view events and services.
- Chapter 4, “Configuring SAP to Send IDocs to an Event,” describes how to configure and test SAP to send IDocs to an application view event.
- Appendix A, “Sample Files,” provides sample request and response documents sent between SAP and the BEA WebLogic Adapter for SAP.

What You Need to Know

This document is written for system integrators who develop client interfaces between SAP and other applications. It describes how to use the BEA WebLogic Adapter for SAP in order to integrate SAP IDocs, RFCs, and BAPIs with WebLogic Integration. It is assumed that readers know Web technologies and have a general understanding of Microsoft Windows and UNIX systems as well as the WebLogic Integration and WebLogic Server infrastructure.

Related Information

The following documents provide additional information for the associated software components:

- *BEA WebLogic Adapter for SAP Installation and Configuration Guide*
- *BEA WebLogic Adapter for SAP Release Notes*
- *BEA Application Explorer Installation and Configuration Guide*
- BEA WebLogic Server installation and user documentation, which is available at the following URL:

  `http://edocs.bea.com/more_wls.html`
- BEA WebLogic Integration installation and user documentation, which is available at the following URL:

  `http://edocs.bea.com/more_wli.html`

Contact Us!

Your feedback on the BEA WebLogic Adapter for SAP 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 SAP documentation.

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

If you have any questions about this version of BEA WebLogic Adapter for SAP, or if you have problems using the BEA WebLogic Adapter for SAP, 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 that 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

Convention	Item
`{ }`	Indicates a set of choices in a syntax line. The braces themselves should never be typed.

`[ ]`	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.

CHAPTER

1 Introducing the BEA WebLogic Adapter for SAP

This section introduces the BEA WebLogic Adapter for SAP and describes SAP business objects and WebLogic Integration. It includes the following topics:

- WebLogic Integration
- How the BEA WebLogic Adapter for SAP Works

You can use the BEA WebLogic Adapter for SAP to mine your existing SAP business procedures and applications for reuse with other applications and to participate in distributed e-business processes. High-speed, low-impact access to SAP exposes the critical business logic, and the data contained within, for reuse. This is the key to building a successful e-business or integrated enterprise.

The BEA WebLogic Adapter for SAP is designed specifically to provide simple, standard access to a set of business operations or functions, such as SAP Remote Function Call (RFC) modules, BAPIs (Business Application Programming Interfaces), and IDocs (Intermediate Documents), which are used to support existing business processes.

An operation that is used to make a request of SAP is referred to as an adapter *service*. An operation that is used to await and respond to an SAP event (for example, a specific message) is referred to as an adapter *event*. Services and events are essentially self-describing objects (that is, the name indicates the business function) that use XML schema to define their input and output.

They work in the following way:

- **Remote Function Call (RFC)** modules are sessions established from the calling application to the SAP system. A user ID is logged on and then a call is issued, triggering processing inside the call. When the call is processed it usually returns information, such as a return code and application data. The calling application waits for processing to complete, then receives the data. It continues processing, taking the result into account. It can even issue multiple RFCs during one session.
- **Business Application Programming Interfaces (BAPIs)** are interfaces within the business framework, which are used to link SAP components to one another or to third-party components. BAPIs are called synchronously and return information. For BAPIs, the client needs to do the appropriate error handling.
- **Intermediate Documents (IDocs)** are documents that are processed asynchronously– that is, no information is returned to the client. As soon as one asynchronous method is involved, the overall communication flow is asynchronous. As a result, the sender should not be on standby awaiting an answer.

The BEA WebLogic Adapter for SAP quickly and easily integrates your SAP IDocs, RFCs, and BAPIs via WebLogic Integration workflows. The adapter and WebLogic Integration provide all the functionality you need to integrate your mission critical SAP system with other enterprise applications. Adapter benefits include:

- Eliminating the need for custom coding.
- Running SAP IDocs, BAPIs, and RFCs both synchronously and asynchronously from WebLogic Integration.
- Allowing SAP to initiate bidirectional business process management workflows using the application view events.
- Creating application views directly from SAP metadata using BEA Application Explorer.
- Standard JCA and JMS-based adapter services and events, ensuring reusability from the entire WebLogic Server platform.
- Integrating SAP events and services with WebLogic Integration.

WebLogic Integration

WebLogic Integration is a single solution that delivers application server, application integration, business process management, and B2B integration for the enterprise. With its comprehensive business process management capabilities, WebLogic Integration provides a powerful J2EE, EJB, and XML-based business process engine that enables customers to design, execute, and optimize enterprise-wide business processes involving systems, applications, and human decision makers.

These enterprise-wide solutions require integration with both external and internal systems in order for projects to be successful. Some of these systems are packaged applications in which organizations have made a substantial investment of time and money. To justify the investments, these systems must be accessible from WebLogic Integration. While some user organizations attempt to manually integrate JCA-based connections to the packaged applications, and even achieve limited success in these efforts, most organizations take the recommendations of industry analysts in seeking out vendor-supplied application adapters.

SAP R/3 is probably the most widely used packaged application that must be accessible from WebLogic Integration for companies to successfully complete their integration projects. The BEA WebLogic Adapter for SAP allows an organization to fully integrate its SAP R/3, mySAP.com, SAP Markets, or SAP Portals application systems with virtually any other legacy system, DBMS, EDI, B2B, ERP, CRM, or SCM application on any platform.

How the BEA WebLogic Adapter for SAP Works

The paradigm that the BEA WebLogic Adapter for SAP uses includes application views, services, and events.

An application view is a standard self-describing interface to an application. The BEA WebLogic Adapter for SAP services are exposed in WebLogic Integration Studio using design elements, or plug-ins, known as nodes. These include Task nodes, which specify the operations to be performed by a BEA WebLogic Adapter for SAP, and Event nodes, which set the business processes that occur when a specific event is "pushed" from the adapter.

For outbound processing, the BEA WebLogic Adapter for SAP is invoked from the Action node and will, in turn, perform a transaction against SAP using the IFR XML, BAPI, RFC, or IDoc interfaces. For inbound processing, the adapter converts the specific SAP event into an XML document that triggers the start of a business process.

The BEA WebLogic Adapter for SAP interfaces are exposed as application views, providing the XSD XML schemas for event, request, and response document schemas that are imported into the WebLogic Integration repository. Once WebLogic Integration knows of these documents, they can be used in WebLogic Integration Studio and other WebLogic Integration tools. In addition, since application views are supported by the WebLogic Server strategy, the same BEA WebLogic Adapter for SAP can be leveraged by other WebLogic Server JCA-based applications to increase ROI.

The BEA WebLogic Adapter for SAP enables users to execute SAP IFR XML, IDocs, BAPI calls, and custom RFCs from WebLogic Integration as application views. To do this, the user creates the event, request, and response XML document schemas using BEA Application Explorer, which is implemented as a stand-alone Java Swing GUI. This GUI exposes all the components of your SAP system and enables you to select the ones for which you want to create an application view. By connecting the BEA Application Explorer to your SAP system, you can ensure that all the necessary communication and security information is gathered using SAP calls, and then stored in a WebLogic Integration Connection Factory database, to be used at execution time by the BEA WebLogic Adapter for SAP. This allows the application views to separate the business logic—contained in the XML event, request, and response documents—from the physical connection data, which is stored in the WebLogic Integration repository. This shields users from the details of executing SAP IFR XML, IDoc, BAPIs, and RFCs.

The deployed application view from BEA WebLogic Adapter for SAP has the following features:

- Support for Remote Function Calls (RFC), Business Application Programming Interfaces (BAPI), and Intermediate Documents (IDoc) interfaces to SAP. RFCs and BAPIs are called synchronously by the adapter and always return data (either technical error information or a well-formed response document). IDocs are processed asynchronously.
- Consistent data representation—a standard XML representation of event and service request/response documents for SAP. The developer is freed from the specific details of the SAP interface (BAPI, RFC, IDoc, IFR XML) and the specific configuration details of the target SAP system.
- XML validation. The schemas used by WebLogic Integration are validated against SAP Business Object Repository (BOR) to ensure that each message conforms to the correct configuration of the target SAP system. Since the schemas are built dynamically from the target SAP system, this all but eliminates the possibility of errors in formatting or executing SAP requests.
- Adheres to SAP ABAP serialization rules and SAP Interface Repository standards published by SAP AG.

Besides being able to run SAP IFR XML, IDocs, BAPIs, and RFCs from WebLogic Integration, the adapter can also receive RFCs and IDocs directly from SAP and make them available to WebLogic Integration. The SAP system can be configured to send an IDoc or RFC out to a logical system when a certain event occurs. The output sent by SAP can be in any of these forms:

- An RFC request—for example, `RFC_CUSTOMER_GET`.
- A BAPI request—for example, `BAPI_COMPANYCODE_GETLIST`.
- An IDoc as an XML document—for example, `DEBMAS01`.
- An IDoc in raw data form.

CHAPTER

2 Creating Schemas for SAP Business Objects

This section describes how to use the BEA Application Explorer to generate schemas that describe your SAP business objects. It contains the following topics:

- Overview
- Generating Schemas Using the BEA Application Explorer

Overview

The BEA WebLogic Adapter for SAP, in order to interact with your SAP business objects, requires schemas describing those objects. You can generate the schemas using the BEA Application Explorer:

1. Specify the directory in which you want the schemas to reside.
2. Browse your SAP system to identify the business object for which you want to create a schema.
3. Generate the schema.

   You can create an event schema describing the data that the SAP system sends to the adapter, or a pair of request and response schemas for service calls from the adapter to SAP.

Note that if you want to create a schema for a user-defined RFC module, contact your BEA customer support representative for help. You cannot use the BEA Application Explorer to generate schemas for user-defined RFC modules.

Note: It is important to understand that the connection information and the event, request, and response schema information that you enter and that is created by the BEA Application Explorer, directly affects the connections, events, and services available to the BEA WebLogic Adapter for SAP.

Service requests are Remote Procedure Calls (RPCs) sent by the adapter to SAP for execution. The request runs a process through the application system connection. The request specifies input parameters that are described by its request schema. For each adapter, the BEA Application Explorer displays summary information and request details. The service request expects a response, called a service response.

Service responses are answer sets returned from the application system connection in response to a service request. SAP uses service responses to return results to the adapter. A service response is described by its service response schema.

Events are requests arriving from SAP that are triggered by SAP activity. For example, a call center worker may enter a purchase order or update a customer record through a GUI screen connected directly to SAP. This SAP event may trigger a process that makes a remote call to the BEA WebLogic Adapter for SAP.

Business Objects are the available SAP RFC modules, BAPI methods, and IDocs that appear in the BEA Application Explorer when you connect to the SAP system.

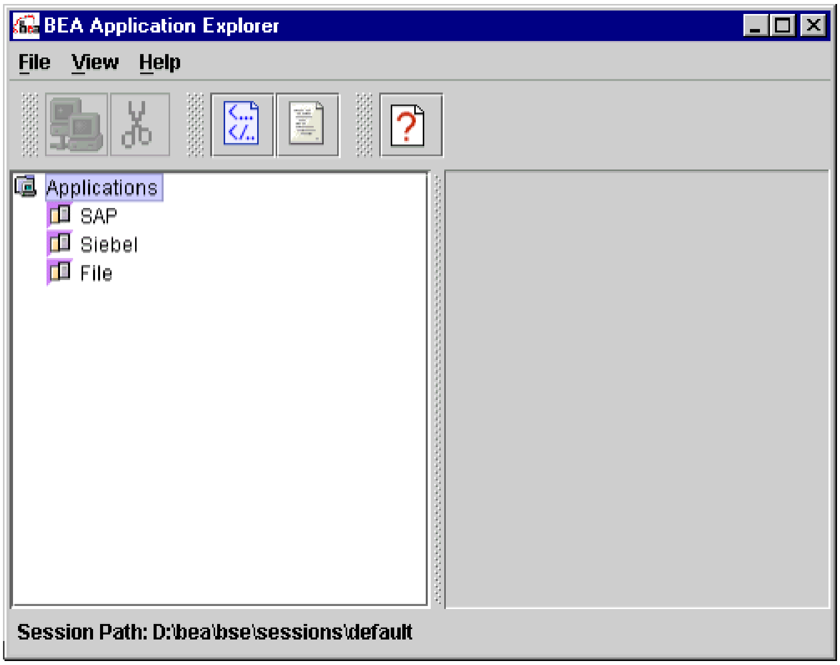
For comprehensive information about the BEA Application Explorer, see the *BEA Application Explorer Installation & Configuration Guide*.

Generating Schemas Using the BEA Application Explorer

To generate schemas for an SAP business object using the BEA Application Explorer:

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

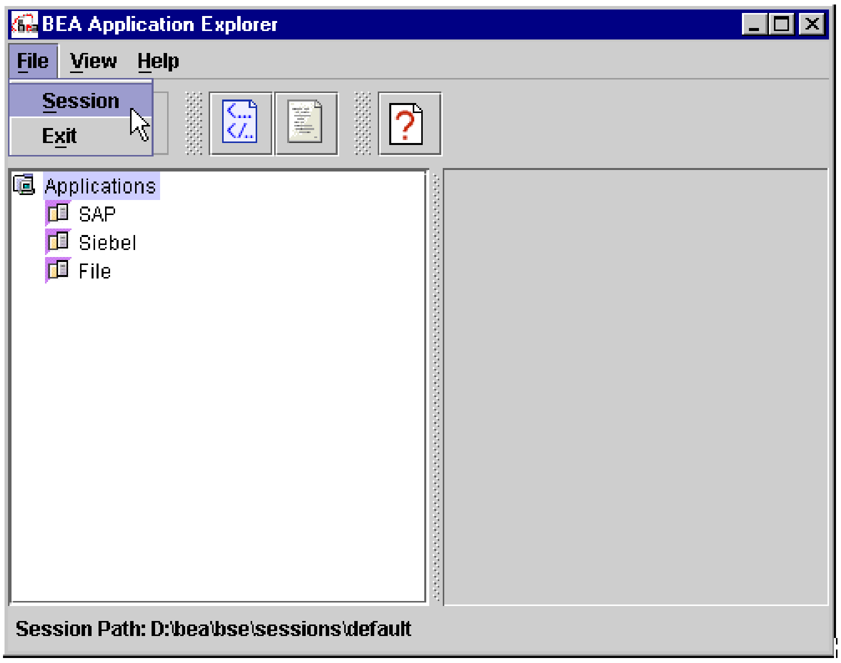
Figure 2-1 BEA Application Explorer Main Window



Session Path: D:\bea\bse\sessions\default

2. From the File menu, choose Session to change the default session path.

Figure 2-2 Choosing Session in BEA Application Explorer



Session Path: D:\bea\bse\sessions\default

The Enter Session Path window opens, displaying the default session path.

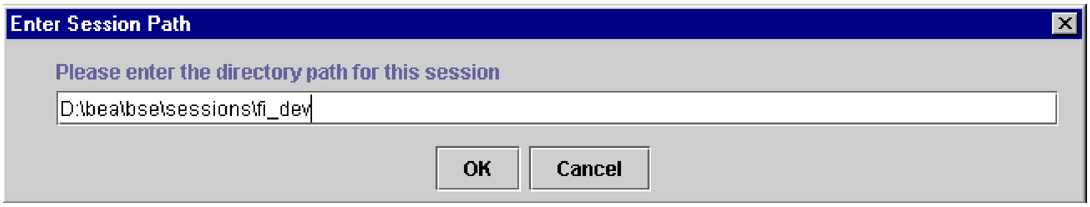
The session path holds the that schemas you that generate and your SAP connection information:

`session_path`\SAP\`connection_name`\`schemas`

3. If you want to accept the default session path, click OK. Otherwise, to specify a different path, enter the path.

   For example, you may want to specify a path for a particular project or for a logical grouping of services and events.

Figure 2-3 Enter Session Path Dialog Box

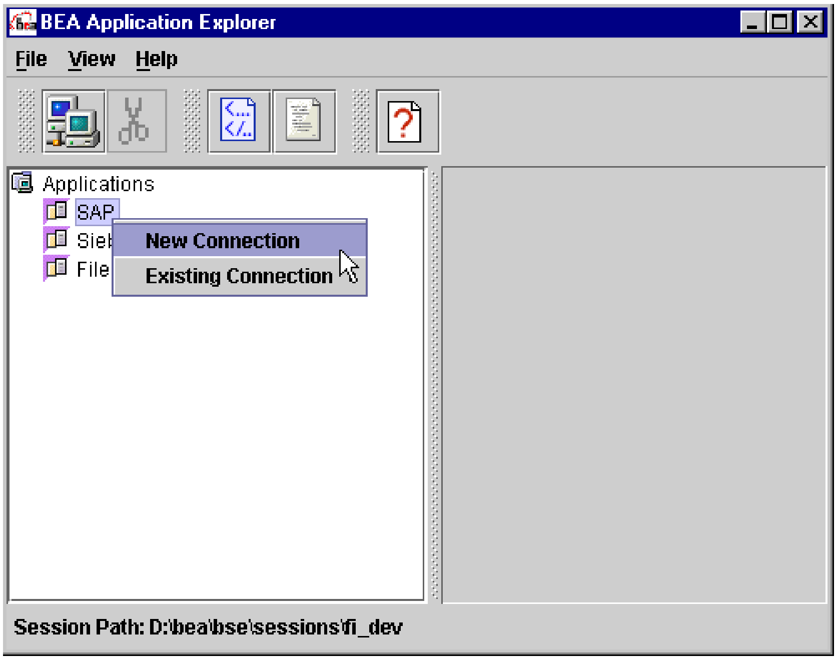


D:\bea\bse\sessions\fi_dev

4. You can define a new connection to an SAP system or use an existing connection:

   - To define a new connection to an SAP system, right-click SAP→New Connection. A dialog box opens prompting you for a connection name; continue with step 5.
   - To use an existing connection, right click SAP→Existing Connection→*your connection*. The connection is displayed below the SAP node in the left pane; skip ahead to step 8.

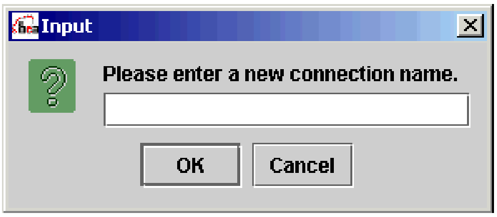
Figure 2-4 Selecting a New Connection in BEA Application Explorer



Session Path: D:\bea\bse\sessions\fi_dev

5. Enter a descriptive name for this connection and click OK.

Figure 2-5 New Connection Name Dialog Box



Please enter a new connection name.

The SAP Logon dialog box opens.

6. Enter the appropriate connection information in the System and User tabs.

Figure 2-6 SAP Logon Window - System Tab

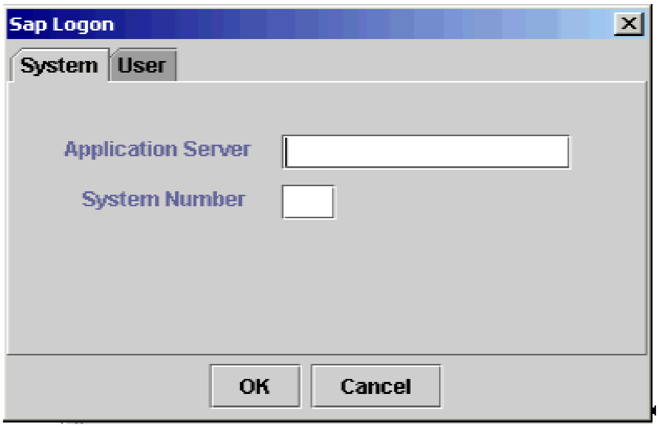
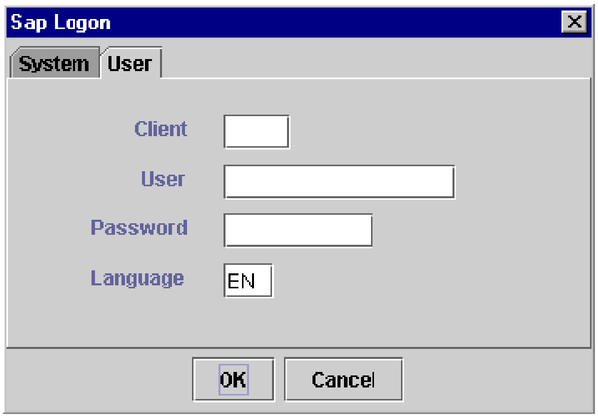


Figure 2-7 SAP Logon Window - User Tab



Language: EN

Table 2-1 SAP Logon Parameters

Property	**Description**
Application Server	Host name of the SAP machine.
System Number	SAP system number.
Client	SAP client.
User	SAP user name.
Password	SAP password.
Language	National language used by SAP. `EN` for English.

7. Click OK after you have entered the requested information.

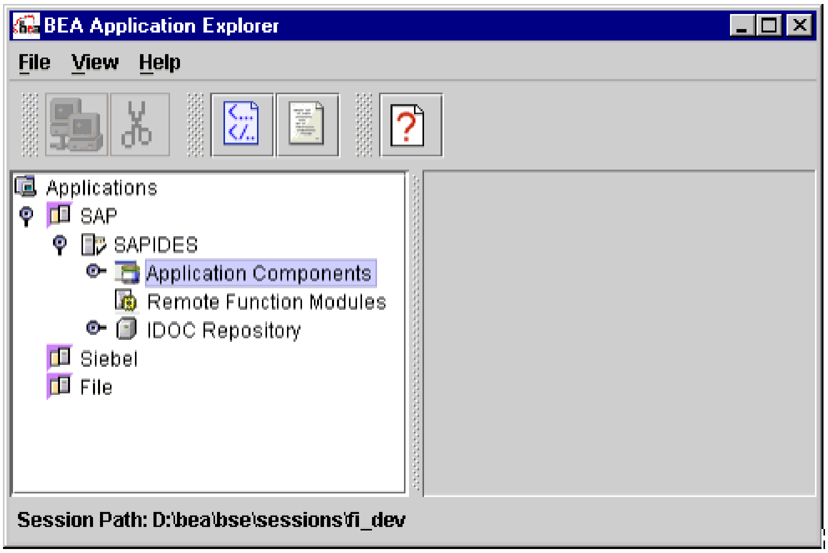
   The new connection is displayed below the SAP node in the left pane.

8. Select a category of business object, browse its objects, and select the object for which you wish to create a schema. Note that:

   - BAPIs are included in Application Components.
   - RFCs are included in Remote Function Modules.

     If you want to create a schema for a user-defined RFC module, contact your BEA customer support representative for help. You cannot use the BEA Application Explorer to generate schemas for user-defined RFC modules.
   - IDocs are included in IDOC Repository.

Figure 2-8 Selecting Application Components in BEA Application Explorer

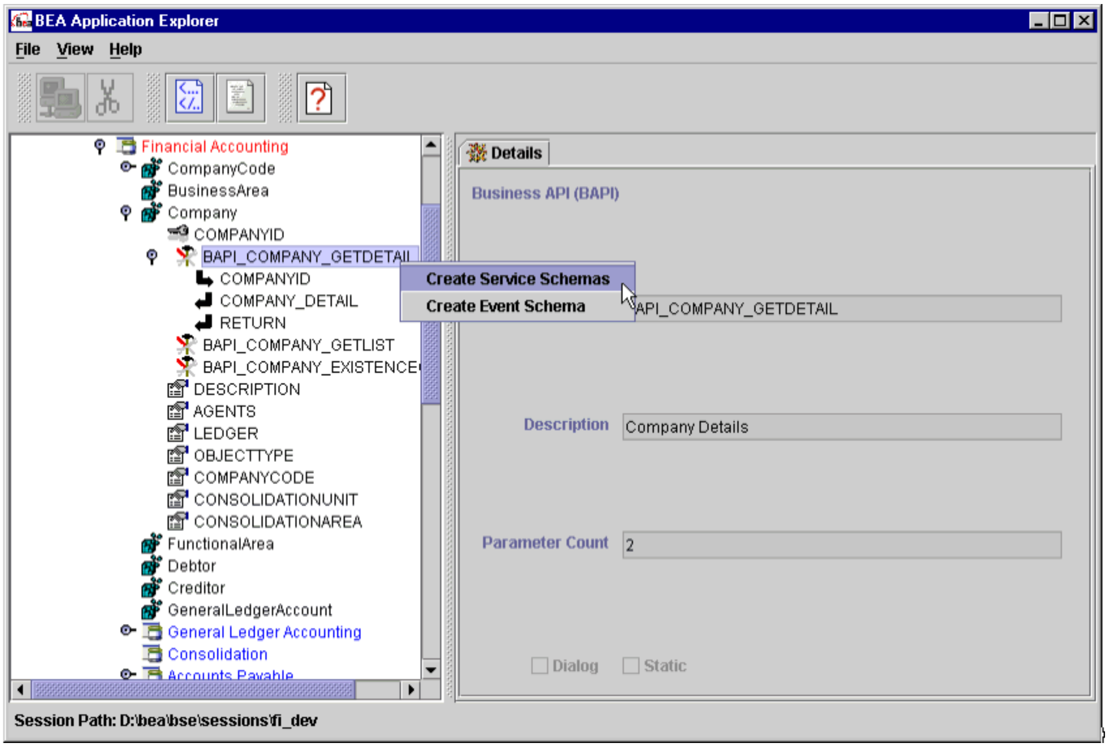


Session Path: D:\bea\bse\sessions\fi_dev

9. Right-click the desired business object to create the service schema or event schema.

   In the following figure, to illustrate this procedure, we have chosen Application Components→Financial Accounting→Company→BAPI_COMPANY_GETDETAIL,andright-clickedthat BAPI to choose Create Service Schema.

Figure 2-9 Choosing Create Service Schema in BEA Application Explorer



API_COMPANY_GETDETAIL

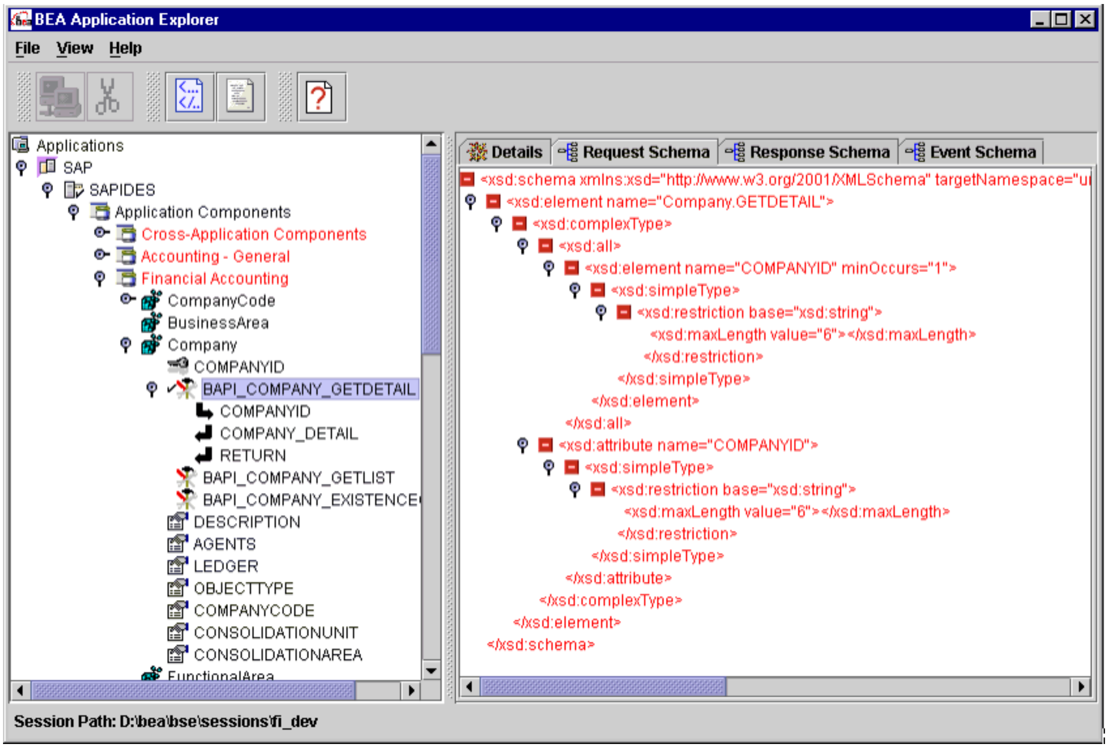
Description: Company Details

Parameter Count: 2

Session Path: D:\bea\bse\sessions\fi_dev

After BEA Application Explorer generates schemas, it displays them in the appropriate tabs of the right pane.

Figure 2-10 Displaying Schemas in BEA Application Explorer

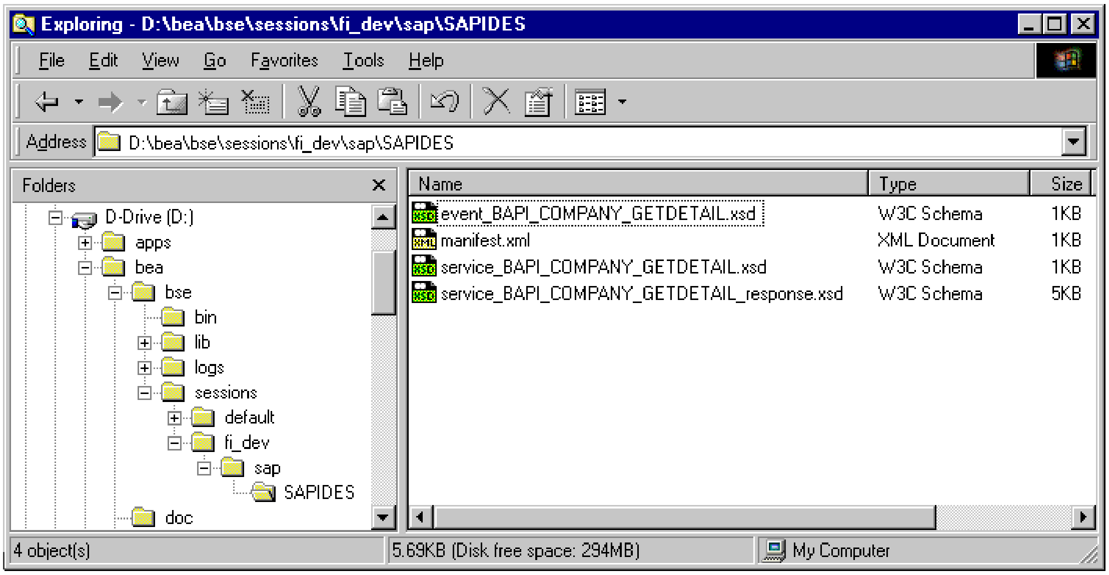


```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="u
  <xsd:element name="Company.GETDETAIL">
    <xsd:complexType>
      <xsd:all>
        <xsd:element name="COMPANYID" minOccurs="1">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:maxLength value="6"></xsd:maxLength>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
      </xsd:all>
      <xsd:attribute name="COMPANYID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="6"></xsd:maxLength>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Session Path: D:\bea\bse\sessions\fi_dev

The following figure illustrates a sample directory structure that BEA Application Explorer generated for the SAP connection named `SAPIDES` under the session named `fi_dev`.

Figure 2-11 Explorer Window - Directory Structure for an SAP connection


Exploring - D:\bea\bse\sessions\fi_dev\sap\SAPIDES
File Edit View Go Favorites Tools Help
Address D:\bea\bse\sessions\fi_dev\sap\SAPIDES
Folders
D-Drive (D:)
apps
bea
bse
bin
lib
logs
sessions
default
fi_dev
sap
SAPIDES
doc
4 object(s)
5.69KB (Disk free space: 294MB)
My Computer

Name	Type	Size
event_BAPI_COMPANY_GETDETAIL.xsd	W3C Schema	1KB
manifest.xml	XML Document	1KB
service_BAPI_COMPANY_GETDETAIL.xsd	W3C Schema	1KB
service_BAPI_COMPANY_GETDETAIL_response.xsd	W3C Schema	5KB

The generated metadata includes a manifest file (`manifest.xml`), the service request schema (`service_BAPI_COMPANY_GETDETAIL.xsd`), the response schema (`service_BAPI_COMPANY_GETDETAIL_response.xsd`), and the event schema (`event_BAPI_COMPANY_GETDETAIL.xsd`).

The following is a sample of the generated `manifest.xml` file.

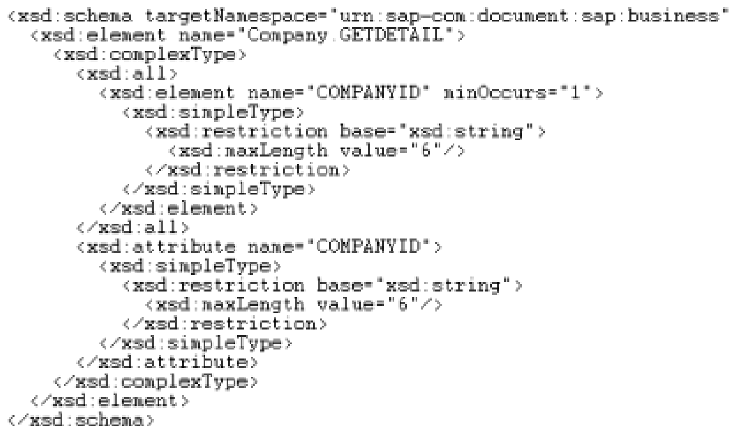
Figure 2-12 Manifest.xml File

```
- <manifest>
  - <connection>
      <user>ib4</user>
      <password>july4</password>
      <ClientNumber>800</ClientNumber>
      <language>EN</language>
      <SystemNumber>00</SystemNumber>
      <hostName>esdsun2</hostName>
    </connection>
  - <schemaref name="BAPI_COMPANY_GETDETAIL">
      <request root="Company.GETDETAIL" file="service_BAPI_COMPANY_GETDETAIL.xsd" />
      <response root="Company.GETDETAIL.Response" file="service_BAPI_COMPANY_GETDETAIL_response.xsd" />
    </schemaref>
  </manifest>
```

The BEA WebLogic Adapter for SAP uses the `manifest.xml` file and accompanying schema(s) to connect to and define the interaction with the application system from an application view. The location of this repository is pointed to in configuration of the adapter during application view creation, as described in Chapter 3, "Configuring Application View Events and Services." During creation of a service or an event, this manifest and the accompanying schemas define the interaction with the EIS.

The following is a sample request schema generated for an SAP BAPI.

Figure 2-13 Sample Request Schema



```
<xsd:schema targetNamespace="urn:sap-com:document:sap:business"
  <xsd:element name="Company.GETDETAIL">
    <xsd:complexType>
      <xsd:all>
        <xsd:element name="COMPANYID" minOccurs="1">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:maxLength value="6"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
      </xsd:all>
      <xsd:attribute name="COMPANYID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="6"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

CHAPTER

3 Configuring Application View Events and Services

This section describes how to create, configure, and test application view events and services. It includes the following topics:

- Creating and Editing Application Views
- Creating an SAP Remote Destination
- Adding an Event to an Application View
- Adding a Service to an Application View
- Deploying an Application View
- Testing an Event
- Testing a Service

Creating and Editing Application Views

This section provides the following procedures for creating and editing application views:

- Starting the Application View Console
- Creating Folders
- Creating an Application View
- Editing an Application View

Starting the Application View Console

To start the Application View Console:

1. Open the following URL in your Web browser.

   `http://host:port/wlai`

   Here, *`host`* is the TCP/IP address or DNS name where WebLogic Server is running, and *`port`* is the socket on which the server is listening.

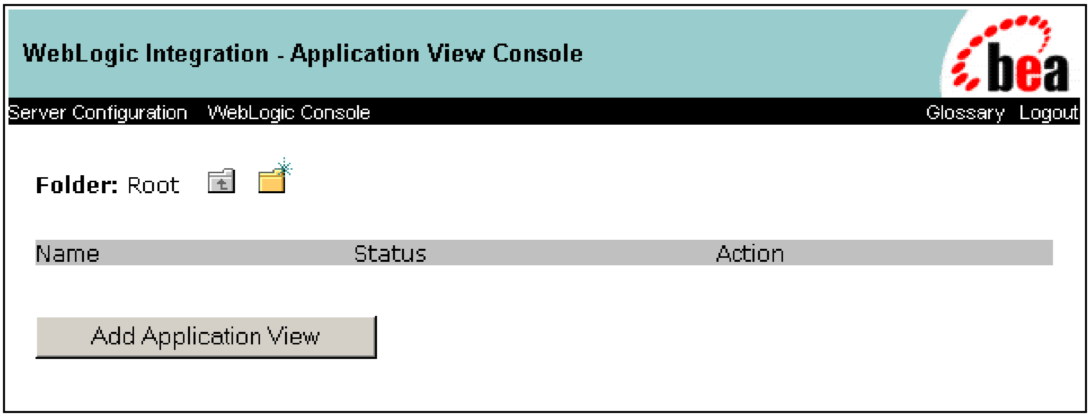
2. If prompted, enter a user name and password.

   Note: If the user name is not `system`, it must be included in the `adapter` group. For more information on adding the administrative server user name to the `adapter` group, see the *BEA WebLogic Adapter for SAP Installation and Configuration Guide*.

3. Click Login.

   The WebLogic Integration Application View Console opens.

Figure 3-1 WebLogic Integration Application View Console Window



Creating Folders

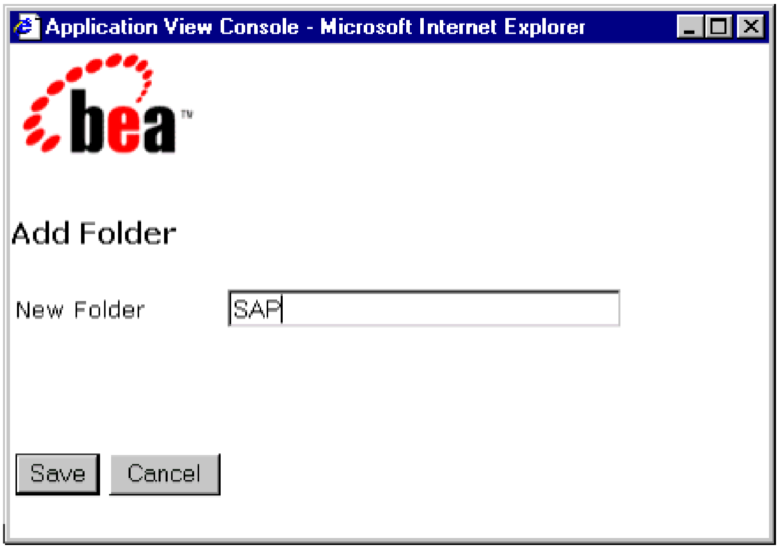
The WebLogic Integration Application View Console provides you with a root folder in which you can store all of your application views. If you wish, you can create additional folders to organize related application views into groups.

To create an application view folder:

1. Start the Application View Console as described in "Starting the Application View Console" on page 3-2.

2. Double-click the new folder icon. The Add Folder window opens.

Figure 3-2 Add Folder Window



New Folder: SAP

3. Enter a name for the folder and click Save.

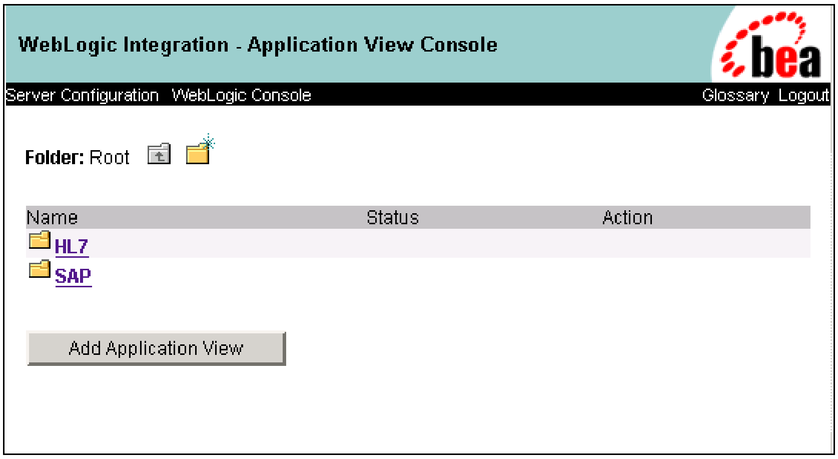
After you create a folder to contain your application views, you can create an application view as described in "Creating an Application View" on page 3-5.

Creating an Application View

To create an application view:

1. Start the Application View Console as described in "Starting the Application View Console" on page 3-2.

   Figure 3-3 Application View Console

2. Select the desired Application View folder.
3. Click Add Application View.

   The Define New Application View window opens.

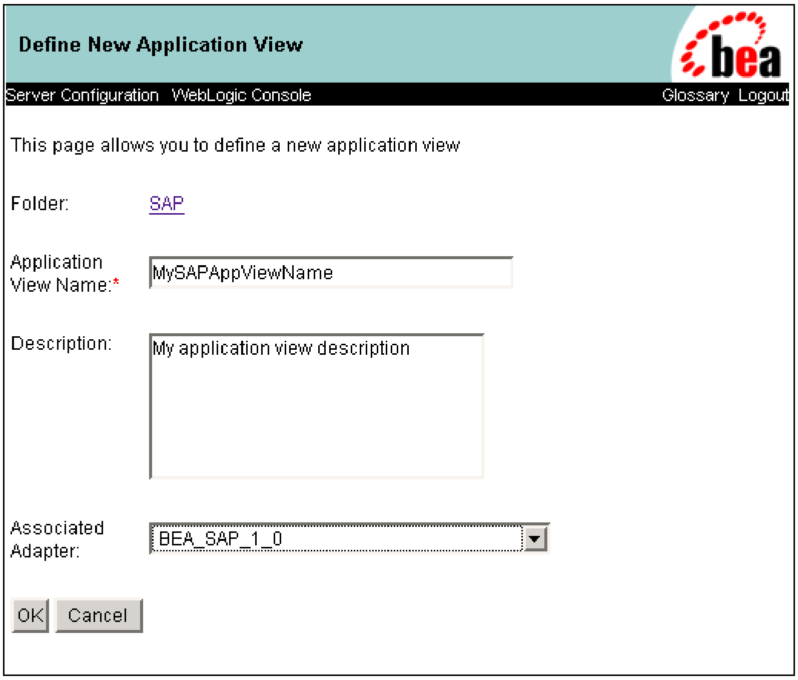
4. Enter a name and description for the application view.

   The name should describe the set of functions performed by this application view. Each application view name must be unique to its adapter. Valid characters include a-z, A-Z, 0-9, and _ (underscore).

   The description will be seen by users when they use this application view with business process management workflows.

5. Select `BEA_SAP_1_0` from the Associated Adapter drop-down list.

Figure 3-4 Define New Application View Window



This page allows you to define a new application view

Folder: SAP

Application View Name:* MySAPAppViewName

Description: My application view description

Associated Adapter: BEA_SAP_1_0

6. Click OK.

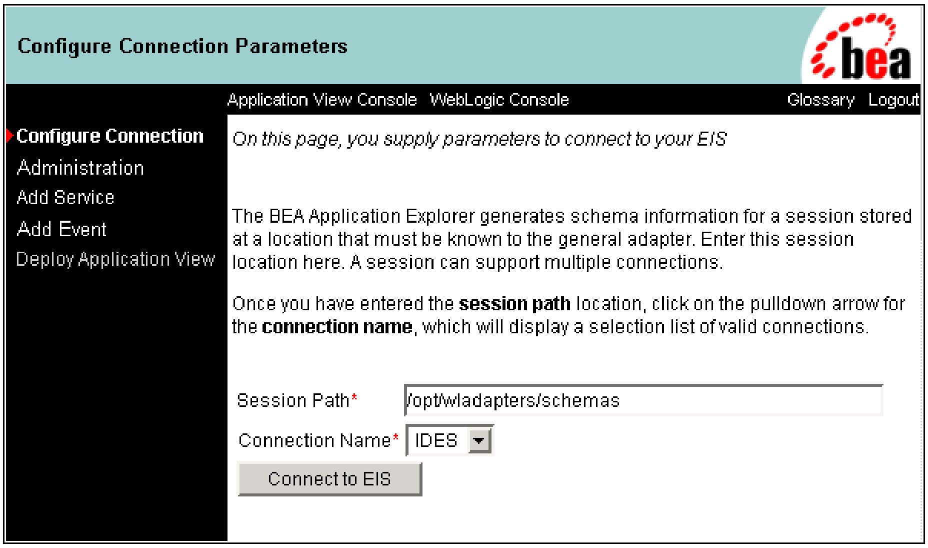
   The Configure Connection Parameters window opens.

7. Enter the name of the BEA WebLogic Adapter for SAP session path (sometimes also known as the session base directory). This path holds your SAP schema and connection information.

   This is the same path that you specify when you use the BEA Application Explorer, as described in Chapter 2, “Creating Schemas for SAP Business Objects.”

8. Select the connection name from the Connection Name drop-down list.

Figure 3-5 Configure Connection Parameters Window



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.

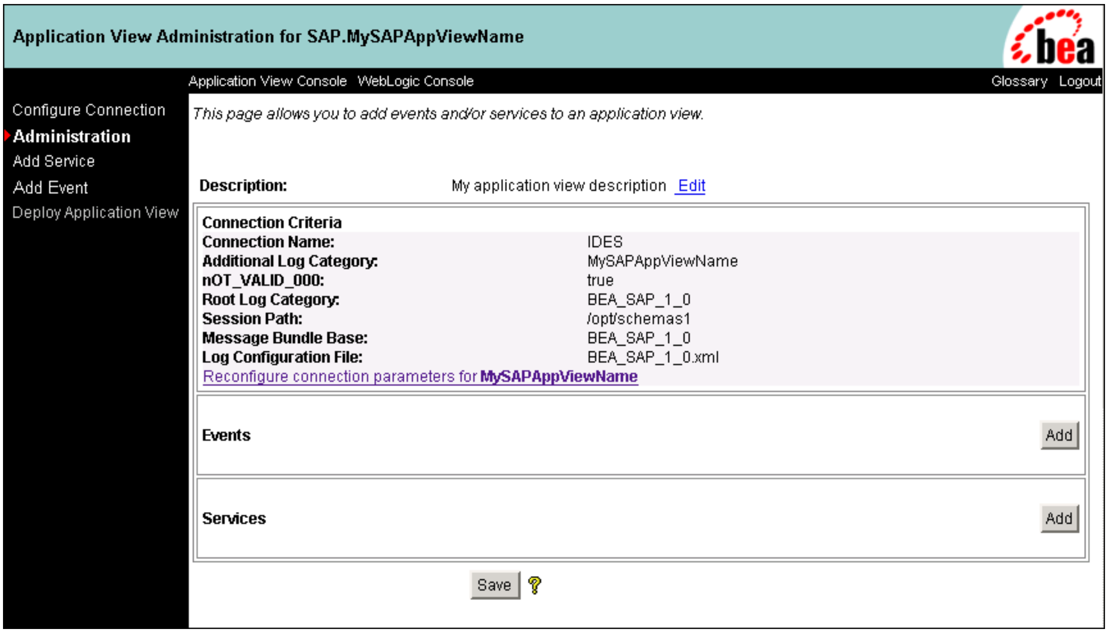
Session Path* /opt/wladapters/schemas

Connection Name* IDES

9. Click Connect to EIS. The Application View Administration window opens.

   You can access the Configure Connection Parameters window (displayed in the previous step) when the application view is not deployed by clicking the Reconfigure connection parameters link. If the application view is deployed, you must first undeploy it.

Figure 3-6 Application View Administration Window



This page allows you to add events and/or services to an application view.

Description:	My application view description Edit
Connection Criteria	
Connection Name:	IDES
Additional Log Category:	MySAPAppViewName
nOT_VALID_000:	true
Root Log Category:	BEA_SAP_1_0
Session Path:	/opt/schemas1
Message Bundle Base:	BEA_SAP_1_0
Log Configuration File:	BEA_SAP_1_0.xml

Reconfigure connection parameters for **MySAPAppViewName**

Events

Services

10. Click Save.

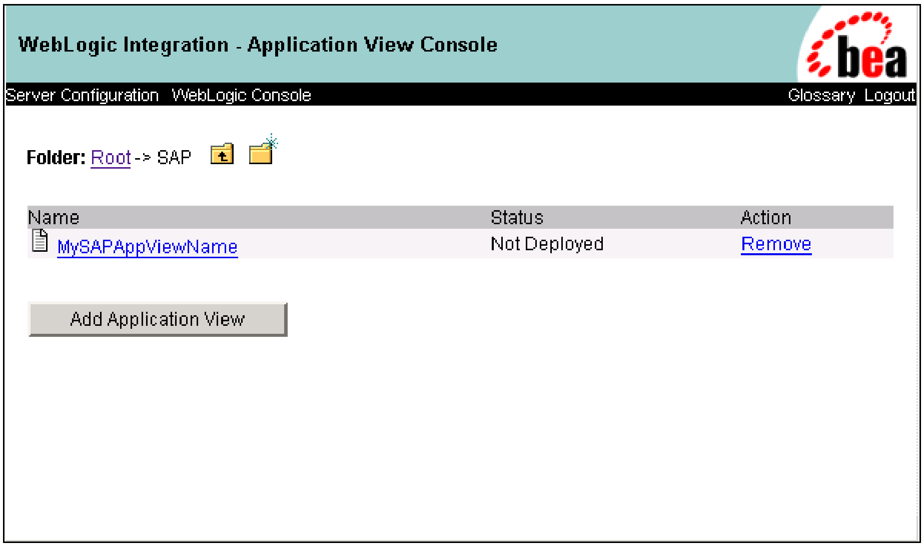
An application view is typically configured for a single business purpose, and contains all the services and events related to that business purpose. Once you have created the application view, you can add the required events and services as described in "Adding an Event to an Application View" on page 3-16 and "Adding a Service to an Application View" on page 3-20.

Editing an Application View

To edit an application view:

1. Start the Application View Console as described in "Starting the Application View Console" on page 3-2.
2. Select the desired Application View folder.

Figure 3-7 Application View Console


WebLogic Integration - Application View Console
bea
Server Configuration WebLogic Console
Glossary Logout
Add Application View

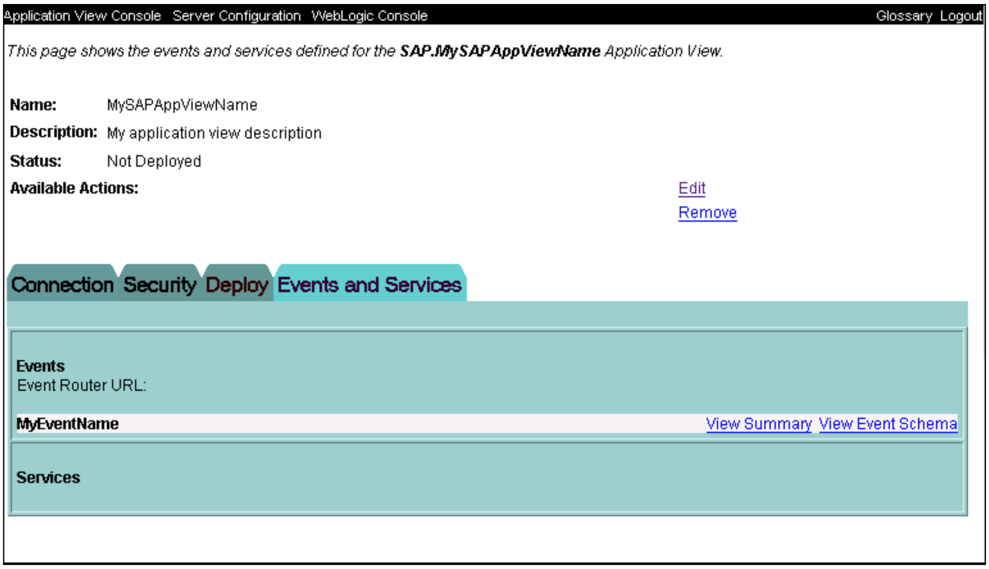
Folder: Root -> SAP

Name	Status	Action
MySAPAppViewName	Not Deployed	Remove

3. Select the Application View.

   The Summary for Application View window is displayed.

Figure 3-8 Summary for Application View Window


Application View Console Server Configuration WebLogic Console
Glossary Logout

This page shows the events and services defined for the ***SAP.MySAPAppViewName*** *Application View.*

Name: MySAPAppViewName
Description: My application view description
Status: Not Deployed
Available Actions: Edit Remove

Connection
Security
Deploy
Events and Services

Events
Event Router URL:

MyEventName View Summary View Event Schema

Services

4. If the application view is deployed, select Undeploy from the Available Actions.
5. Select Edit from the Available Actions to display the Application View Administration window.

Creating an SAP Remote Destination

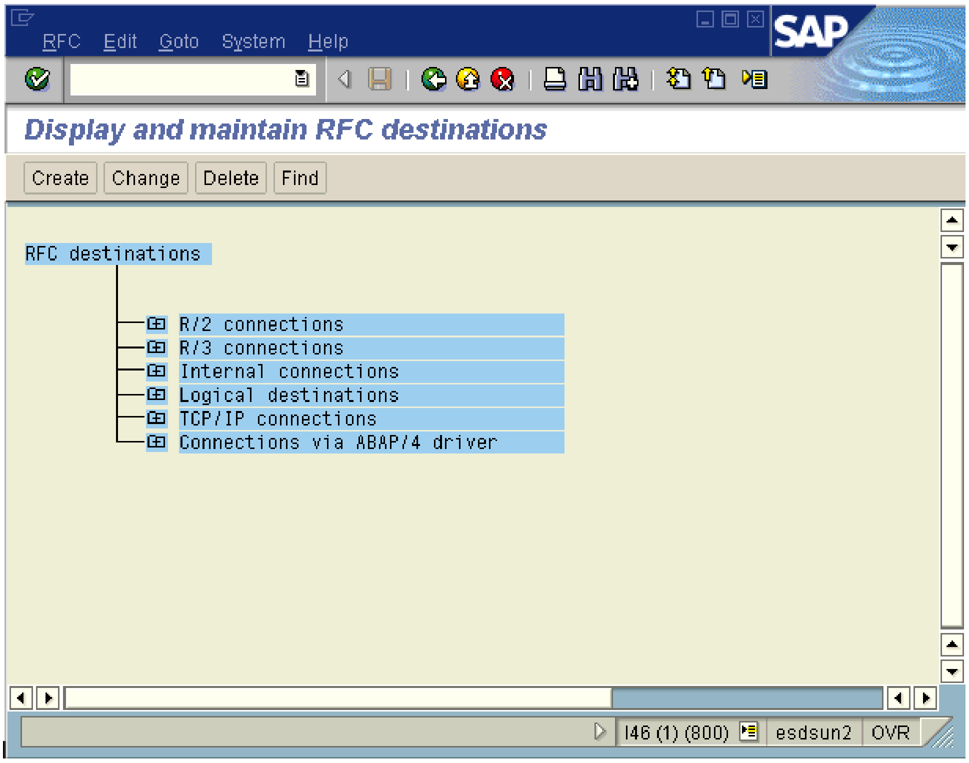
Before you add events to your application view, you must define an RFC destination on the SAP system. This enables the SAP system to issue remote function calls (RFCs) or BAPIs to the adapter, and to send IDocs to the adapter.

The RFC destination is a symbolic name specifying the target system for an RFC. The RFC destination must be configured to connect to the adapter.

To create an RFC destination called BEASAPDEST:

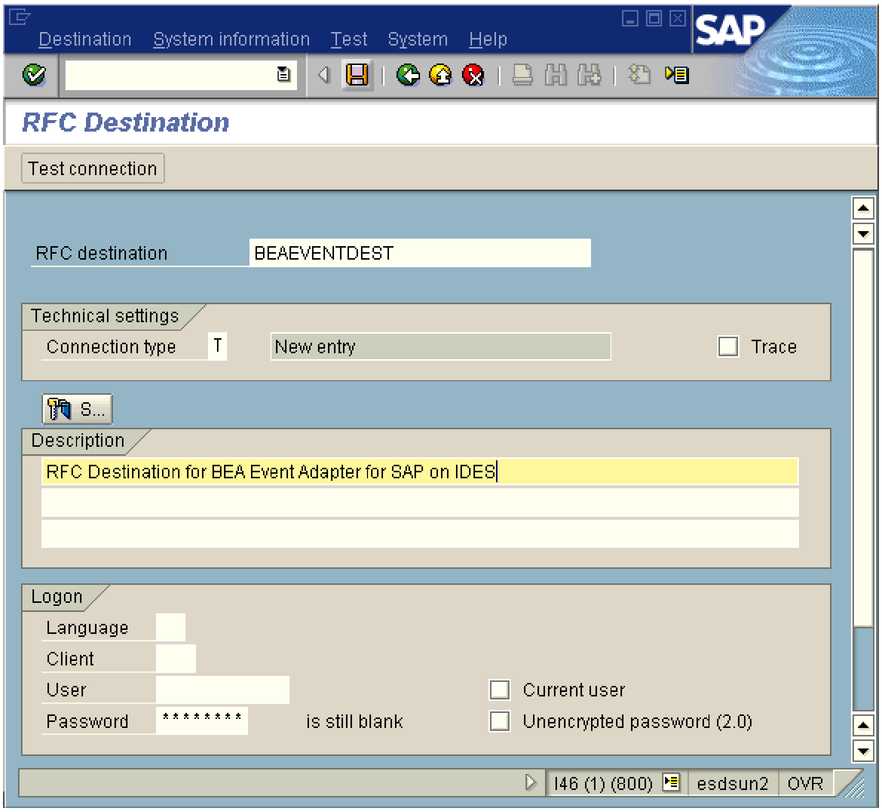
1. Start the SAPGUI and log on to the SAP system.
2. Choose Tools→Administration→Administration→Network→RFC destinations.
3. Execute transaction `sm59` into the transaction field.

Figure 3-9 Display and Maintain RFC Destinations Window



4. Right-click TCP/IP connections and choose Create.

Figure 3-10 RFC Destination Window



RFC Destination

RFC destination: BEAEVENTDEST

Connection type: T   New entry   ☐ Trace

RFC Destination for BEA Event Adapter for SAP on IDES

Language:
Client:
User: ☐ Current user
Password: ********   is still blank   ☐ Unencrypted password (2.0)

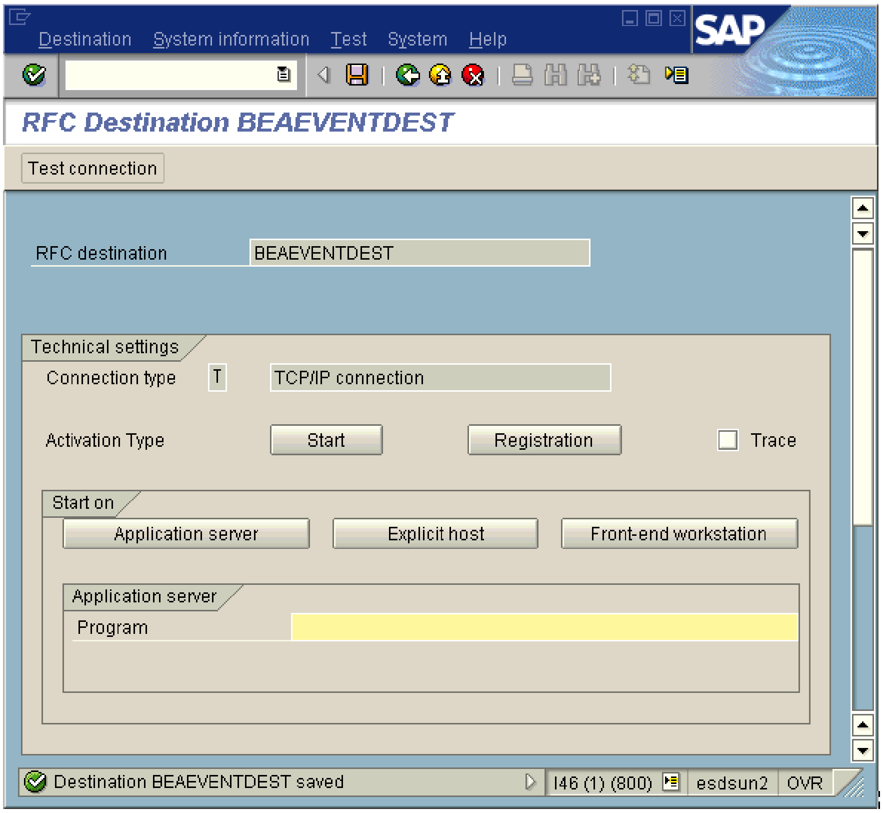
I46 (1) (800)   esdsun2   OVR

5. Enter a destination name in the RFC destination field. The name is case sensitive.
6. Enter T in the Connection type field to designate TCP/IP.
7. Enter a description of the destination, and any comments, in the Description field.

8. Click the Change button on the toolbar or choose Save from the Destination menu.

   The RFC Destination window for your destination opens.

Figure 3-11 RFC Destination BEAEVENTDEST Window



RFC Destination BEAEVENTDEST

RFC destination: BEAEVENTDEST

Technical settings
Connection type: T TCP/IP connection
Activation Type

Start on

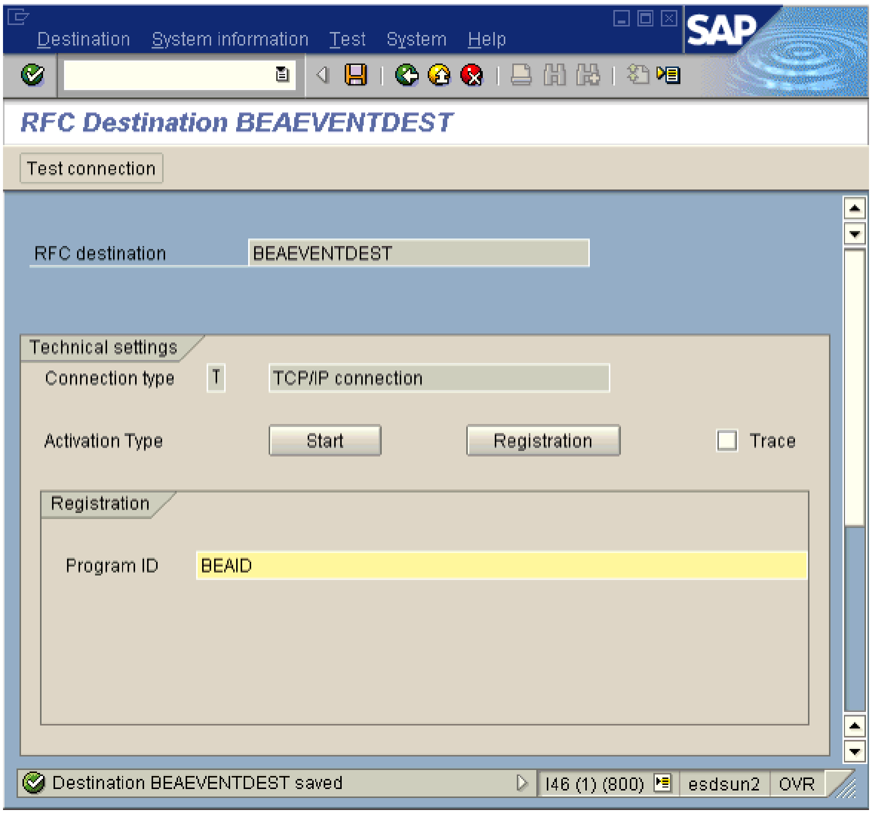
Application server
Program:

Destination BEAEVENTDEST saved | I46 (1) (800) | esdsun2 | OVR

9. Click Registration as Activation Type.
10. In field Registration Program ID field, type BEAID.

11. Click Change on the toolbar or choose Save from the Destination menu.

Figure 3-12 Destination Menu Window



RFC Destination BEAEVENTDEST

RFC destination BEAEVENTDEST

Technical settings

Connection type T TCP/IP connection

Activation Type

☐ Trace

Registration

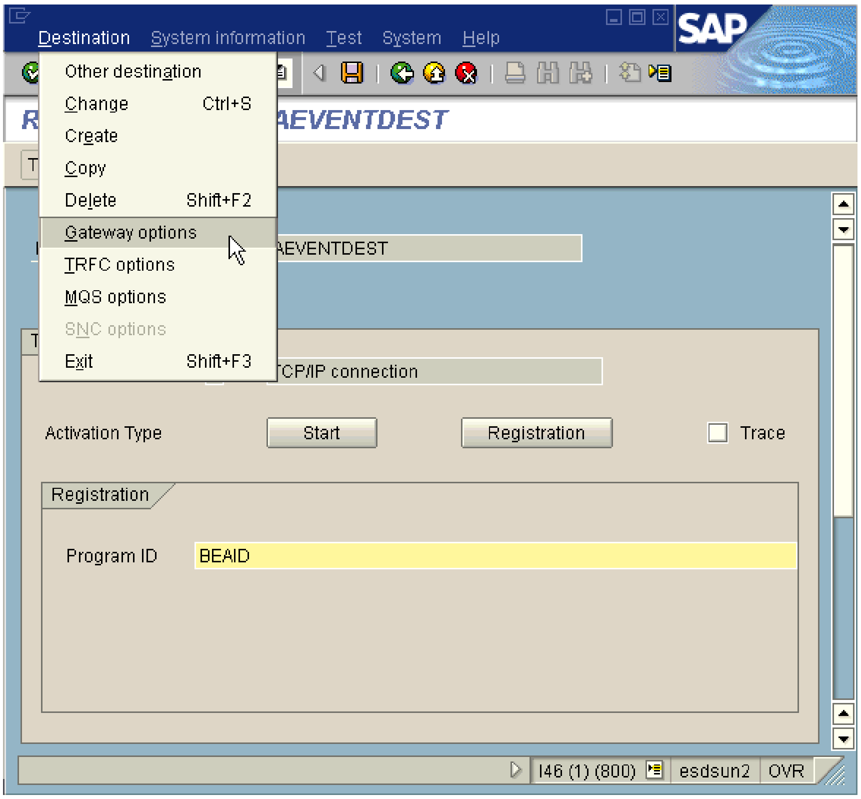
Program ID BEAID

Destination BEAEVENTDEST saved

I46 (1) (800) esdsun2 OVR

12. From the Destination Menu, choose Gateway options.

Figure 3-13 Gateway Options Window

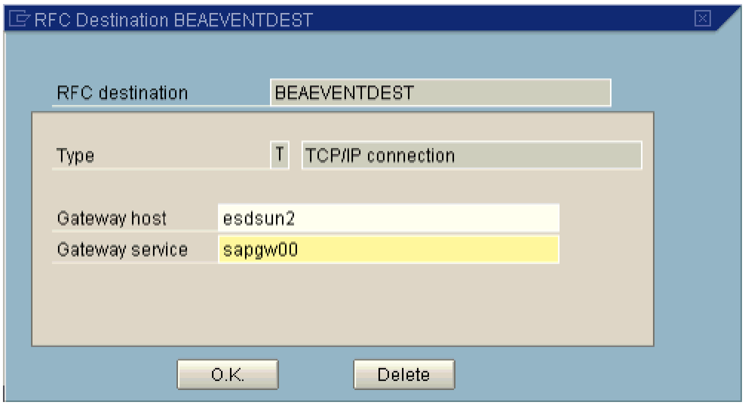


13. Enter the host name of the machine in field Gateway host.

14. Enter sapgw and the SAP system number in field Gateway service, for example, sapgw00.

15. Click OK.

Figure 3-14 Gateway Host and Service Window



RFC Destination BEAEVENTDEST

RFC destination	BEAEVENTDEST	
Type	T	TCP/IP connection
Gateway host	esdsun2	
Gateway service	sapgw00	

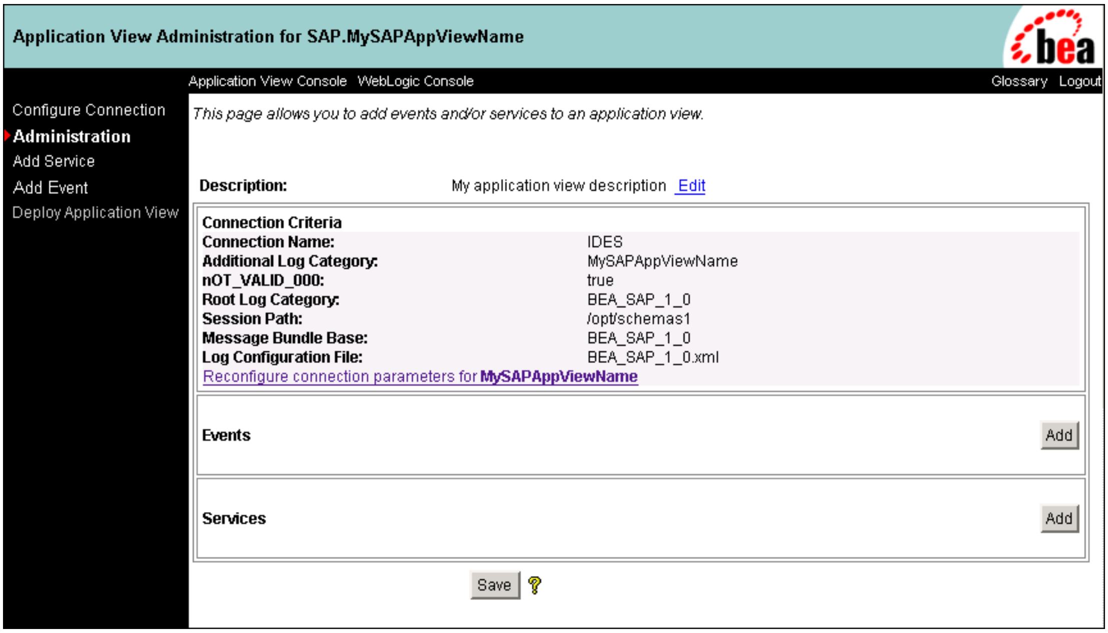
Adding an Event to an Application View

Note: To allow the SAP system to issue remote function calls (RFCs) or BAPIs to the adapter, and to send IDocs to the adapter, you must create an RFC destination on the SAP system as described in "Creating an SAP Remote Destination" on page 3-10.

To add an event to an application view:

1. If it is not already open, display the Application View Administration window as described in "Editing an Application View" on page 3-9.

Figure 3-15 Application View Console Administration Window



This page allows you to add events and/or services to an application view.

Description: My application view description

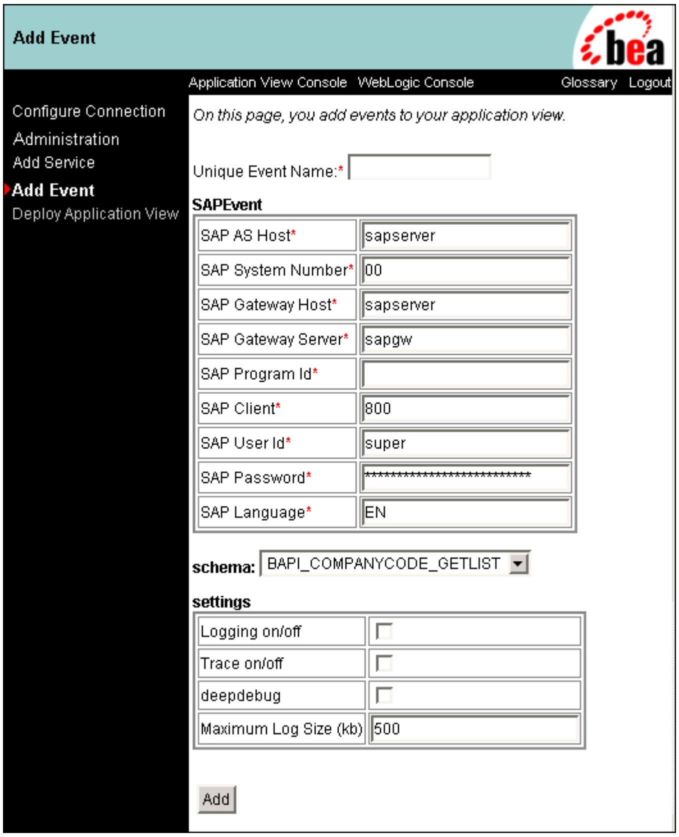
Connection Criteria	
Connection Name:	IDES
Additional Log Category:	MySAPAppViewName
nOT_VALID_000:	true
Root Log Category:	BEA_SAP_1_0
Session Path:	/opt/schemas1
Message Bundle Base:	BEA_SAP_1_0
Log Configuration File:	BEA_SAP_1_0.xml

Events

Services

2. In the Events section, click Add to display the Add Event window.

Figure 3-16 Add Event Window



On this page, you add events to your application view.

Unique Event Name:*

SAPEvent

SAP AS Host*	sapserver
SAP System Number*	00
SAP Gateway Host*	sapserver
SAP Gateway Server*	sapgw
SAP Program Id*	
SAP Client*	800
SAP User Id*	super
SAP Password*	***************************
SAP Language*	EN

schema: BAPI_COMPANYCODE_GETLIST

settings

Logging on/off	☐
Trace on/off	☐
deepdebug	☐
Maximum Log Size (kb)	500

The settings in this window correspond to the TCP connectivity that the adapter creates with SAP to receive SAP events in BAPI, RFC, or IDoc format.

The values displayed are based on the connection information you supplied when you created the connection in the BEA Application Explorer. You are free to change these design time values for your particular run-time behavior.

3. Update the SAP event parameters as required:

   - *SAP AS Host*
     Host name of the machine running the SAP application server.
   - *SAP System Number*

- *SAP Gateway Host*
  Host name of the machine running SAP gateway server.
- *SAP Gateway Server*
  Host name of the SAP gateway server.
- *SAP Program ID*
  RFC program ID created previously (for example, BEAID).
- *SAP Client*
- *SAP User Id*
  Associated SAP user name.
- *SAP Password*
  Associated SAP user password.
- *SAP Language*
  Your desired SAP language, for example, EN for English.

4. Select the appropriate schema from the drop-down list.

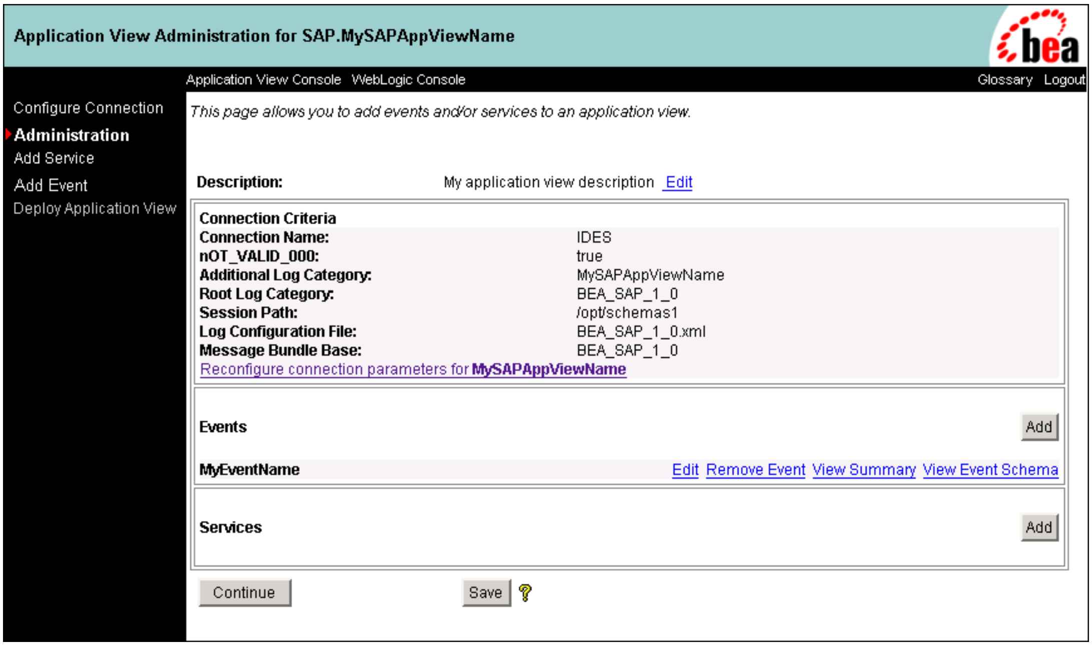
   The schema drop-down list corresponds to the manifest generated for you during your BEA Application Explorer session. All event schemas created during the session should be listed.

5. Select the appropriate check boxes in the settings section to enable logging, trace, and debug.

6. Click Add to add the event.

   The event is displayed in Events section of the Application View Administration window.

Figure 3-17 Application View Administration Window


Application View Administration for SAP.MySAPAppViewName

This page allows you to add events and/or services to an application view.

Description: My application view description Edit

Connection Criteria	
Connection Name:	IDES
nOT_VALID_000:	true
Additional Log Category:	MySAPAppViewName
Root Log Category:	BEA_SAP_1_0
Session Path:	/opt/schemas1
Log Configuration File:	BEA_SAP_1_0.xml
Message Bundle Base:	BEA_SAP_1_0

Reconfigure connection parameters for **MySAPAppViewName**

Events Add

MyEventName Edit Remove Event View Summary View Event Schema

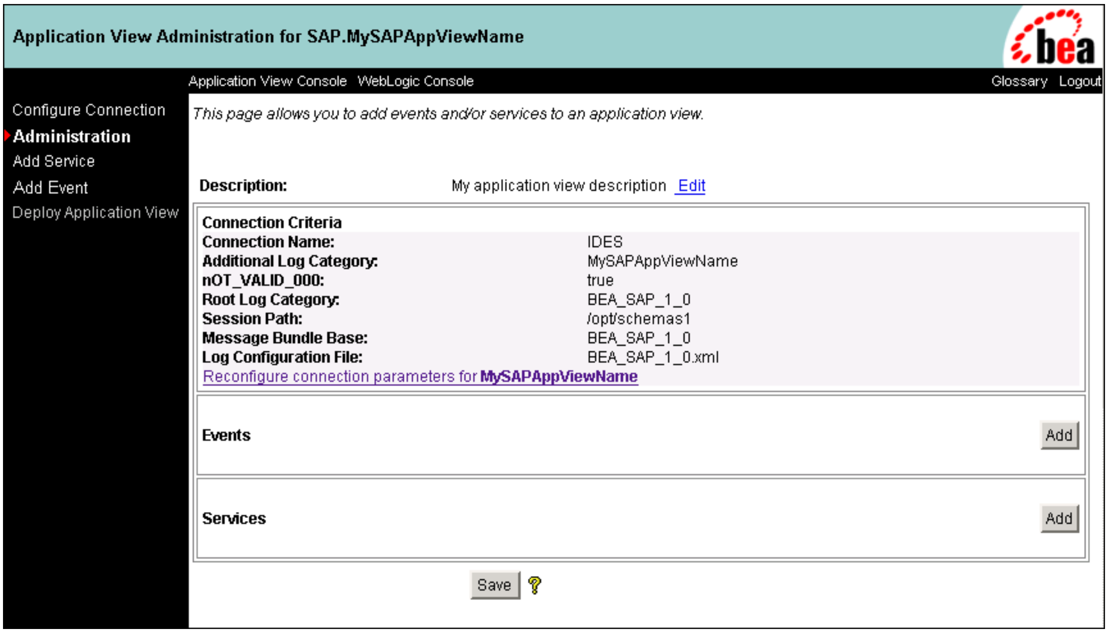
Services Add

Adding a Service to an Application View

To add a service to an application view:

1. If it is not already open, display the Application View Administration window as described in "Editing an Application View" on page 3-9.

Figure 3-18 Application View Administration Window



Application View Administration for SAP.MySAPAppViewName

This page allows you to add events and/or services to an application view.

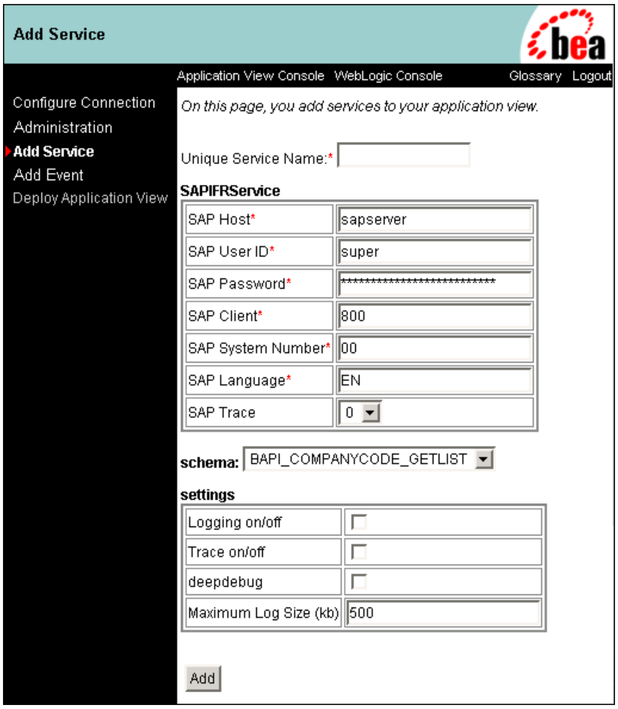
Description: My application view description Edit

Connection Criteria	
Connection Name:	IDES
Additional Log Category:	MySAPAppViewName
nOT_VALID_000:	true
Root Log Category:	BEA_SAP_1_0
Session Path:	/opt/schemas1
Message Bundle Base:	BEA_SAP_1_0
Log Configuration File:	BEA_SAP_1_0.xml

Reconfigure connection parameters for **MySAPAppViewName**

2. In the Services section, click Add to display the Add Service window.

Figure 3-19 Add Service Window



On this page, you add services to your application view.

Unique Service Name:*

SAPIFRService

SAP Host*	sapserver
SAP User ID*	super
SAP Password*	******************************
SAP Client*	800
SAP System Number*	00
SAP Language*	EN
SAP Trace	0

schema: BAPI_COMPANYCODE_GETLIST

settings

Logging on/off	☐
Trace on/off	☐
deepdebug	☐
Maximum Log Size (kb)	500

Note: The BEA WebLogic Adapter for SAP is based on a JDBC interface, so the properties of this connection are based on this framework.

3. Update the SAP service parameters as required.

- *SAP Host*
  Host name of the machine running SAP.
- *SAP User Id*
  Associated SAP user name.
- *SAP Password*
  Associated SAP user password.
- *SAP Client*
- *SAP System Number*
- *SAP Trace*
  SAP trace level. Set to 1 to enable trace, set to 0 to disable trace.

4. Select the appropriate schema from the drop-down list.

   The schema drop-down list corresponds to the manifest generated for you during your BEA Application Explorer session. All service schemas created during the session should be listed.

5. Select the appropriate check boxes in the settings section to enable logging, trace, and debug.

6. Click Add to add the service.

   The service is displayed in the Services section of the Application View Administration window.

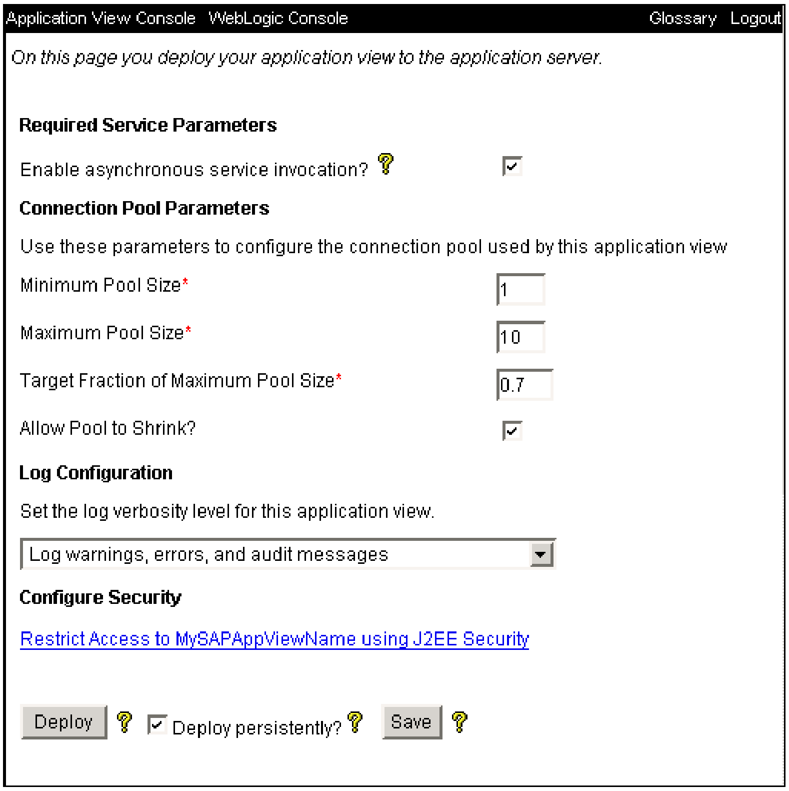
Deploying an Application View

To deploy an application view:

1. Display the Application View Administration window. See "Editing an Application View" on page 3-9.

2. Select Continue to display the Deploy Application View window.

Figure 3-20 Deploy Application View Window



On this page you deploy your application view to the application server.

Required Service Parameters

Enable asynchronous service invocation? ☑

Connection Pool Parameters

Use these parameters to configure the connection pool used by this application view

Minimum Pool Size* 1

Maximum Pool Size* 10

Target Fraction of Maximum Pool Size* 0.7

Allow Pool to Shrink? ☑

Log Configuration

Set the log verbosity level for this application view.

Log warnings, errors, and audit messages

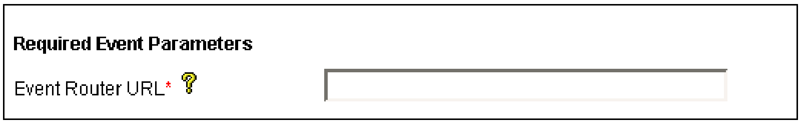
Configure Security

Restrict Access to MySAPAppViewName using J2EE Security

☑ Deploy persistently?

The settings displayed in the Deploy Application View window depend on contents of the application view. The above example is for an application view that contains only services. If the application view contains events, the Required Event Parameters section, shown in the following figure, is displayed.

Figure 3-21 Required Event Parameters



Required Event Parameters

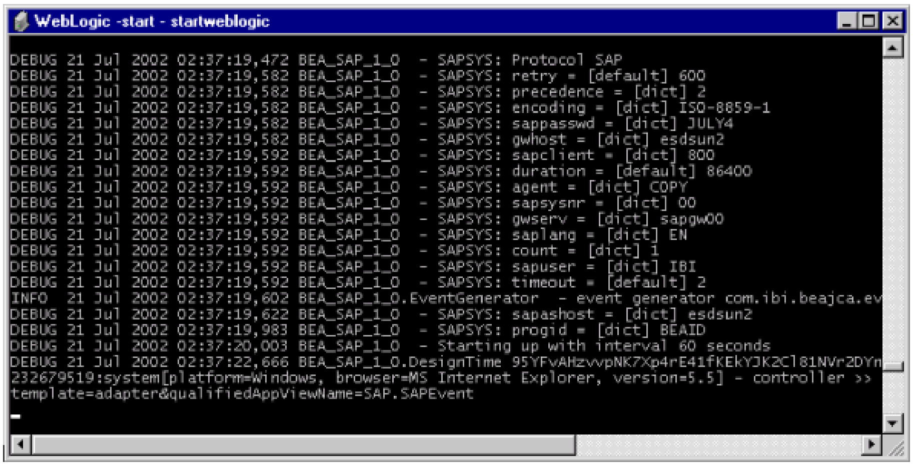
Event Router URL*

3. If required, update the settings.

4. Click Deploy to save and deploy the application view.

   In the WebLogic Server log or command console, you should see messages similar to the following as the application view is deployed.

   Figure 3-22 WebLogic Server Log Window



```
DEBUG 21 Jul 2002 02:37:19,472 BEA_SAP_1_0  - SAPSYS: Protocol SAP
DEBUG 21 Jul 2002 02:37:19,582 BEA_SAP_1_0  - SAPSYS: retry = [default] 600
DEBUG 21 Jul 2002 02:37:19,582 BEA_SAP_1_0  - SAPSYS: precedence = [dict] 2
DEBUG 21 Jul 2002 02:37:19,582 BEA_SAP_1_0  - SAPSYS: encoding = [dict] ISO-8859-1
DEBUG 21 Jul 2002 02:37:19,582 BEA_SAP_1_0  - SAPSYS: sappasswd = [dict] JULY4
DEBUG 21 Jul 2002 02:37:19,582 BEA_SAP_1_0  - SAPSYS: gwhost = [dict] esdsun2
DEBUG 21 Jul 2002 02:37:19,592 BEA_SAP_1_0  - SAPSYS: sapclient = [dict] 800
DEBUG 21 Jul 2002 02:37:19,592 BEA_SAP_1_0  - SAPSYS: duration = [default] 86400
DEBUG 21 Jul 2002 02:37:19,592 BEA_SAP_1_0  - SAPSYS: agent = [dict] COPY
DEBUG 21 Jul 2002 02:37:19,592 BEA_SAP_1_0  - SAPSYS: sapsysnr = [dict] 00
DEBUG 21 Jul 2002 02:37:19,592 BEA_SAP_1_0  - SAPSYS: gwserv = [dict] sapgw00
DEBUG 21 Jul 2002 02:37:19,592 BEA_SAP_1_0  - SAPSYS: saplang = [dict] EN
DEBUG 21 Jul 2002 02:37:19,592 BEA_SAP_1_0  - SAPSYS: count = [dict] 1
DEBUG 21 Jul 2002 02:37:19,592 BEA_SAP_1_0  - SAPSYS: sapuser = [dict] IBI
DEBUG 21 Jul 2002 02:37:19,592 BEA_SAP_1_0  - SAPSYS: timeout = [default] 2
INFO  21 Jul 2002 02:37:19,602 BEA_SAP_1_0.EventGenerator  - event generator com.ibi.beajca.ev
DEBUG 21 Jul 2002 02:37:19,622 BEA_SAP_1_0  - SAPSYS: sapashost = [dict] esdsun2
DEBUG 21 Jul 2002 02:37:19,983 BEA_SAP_1_0  - SAPSYS: progid = [dict] BEAID
DEBUG 21 Jul 2002 02:37:20,003 BEA_SAP_1_0  - Starting up with interval 60 seconds
DEBUG 21 Jul 2002 02:37:22,666 BEA_SAP_1_0.DesignTime 95YFvAHzvvpNK7Xp4rE41fKEkYJK2Cl81NVr2DYn
232679519:system[platform=Windows, browser=MS Internet Explorer, version=5.5] - controller >>
template=adapter&qualifiedAppViewName=SAP.SAPEvent
```

   When deployment is complete, the Summary for Application View window displays the application view deployment status.

At this point, you can test your the application view services and events as described in "Testing an Event" on page 3-25 and "Testing a Service" on page 3-34.

Testing an Event

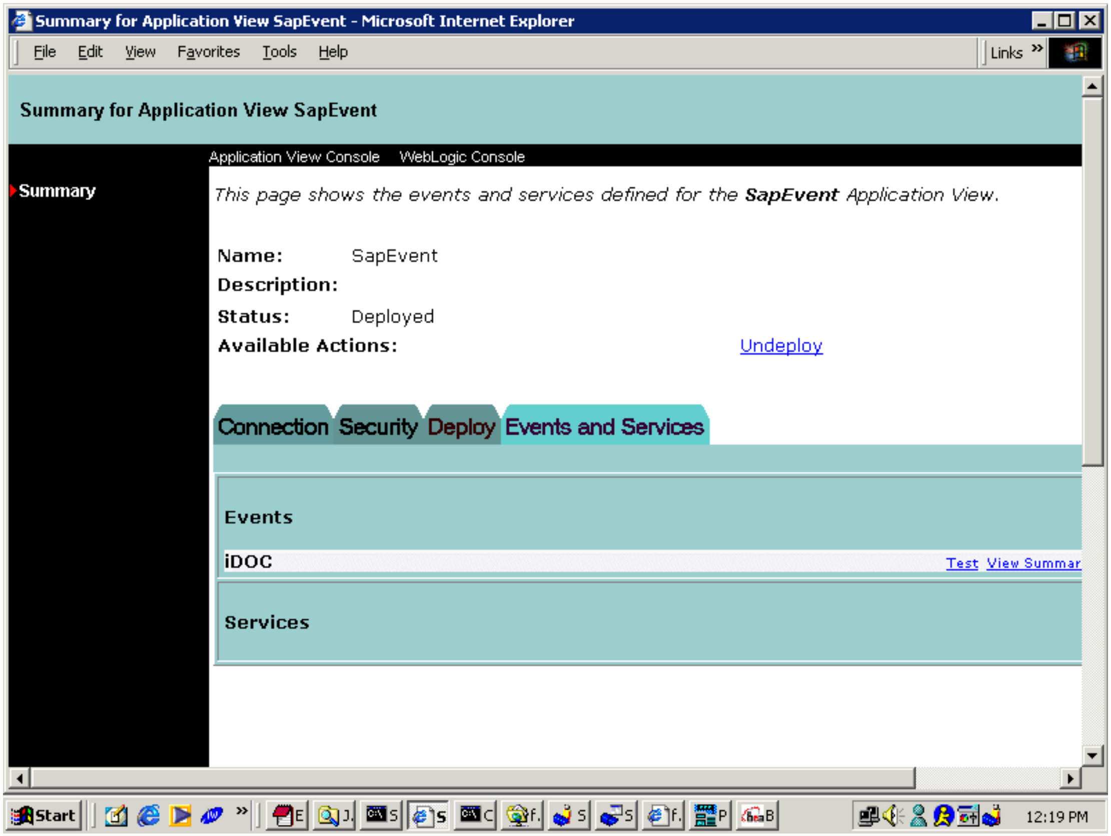
After you add an event to an application view and deploy it as described in "Adding an Event to an Application View" on page 3-16, and "Deploying an Application View" on page 3-23, you can test the event:

- Testing an Event in the Application View Console
- Testing an Event in the Studio

Testing an Event in the Application View Console

After you deploy an application view, the Summary for Application View window is displayed as shown in the following figure. From this window you can test an event as described in the following procedure.

Figure 3-23 Summary for Application View Window


Summary for Application View SapEvent - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Links
Summary for Application View SapEvent
Application View Console WebLogic Console
Summary
Connection Security Deploy Events and Services
Undeploy
Test View Summar
Start
12:19 PM

This page shows the events and services defined for the ***SapEvent*** *Application View.*

Name: SapEvent
Description:
Status: Deployed
Available Actions:

Events
iDOC

Services

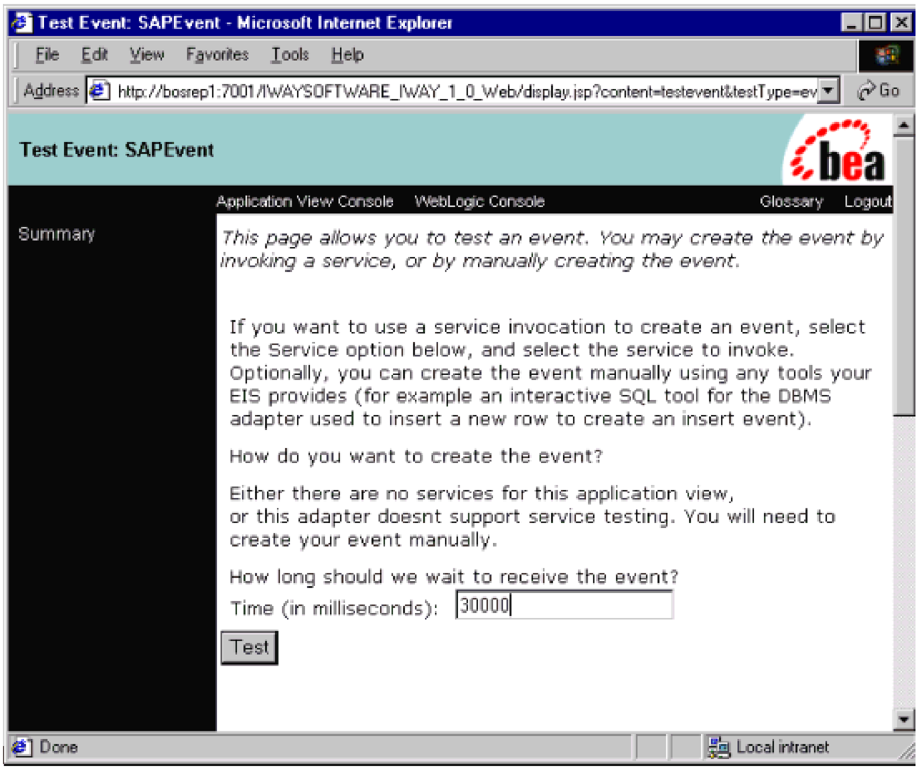
To test an event:

1. In the Events section of the Summary for Application View window, click Test.

   The Test window is displayed, indicating that you can create the event by invoking a service, or by manually creating the event. In this example, the request from SAP to the SAPEvent event is invoked manually.

2. Enter a suitable wait time. For example, 30,000 milliseconds (30 seconds) to enable you to navigate to the SAP GUI and invoke the remote function call.

Figure 3-24 Test Event Window


Test Event: SAPEvent - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://bosrep1:7001/IWAYSOFTWARE_IWAY_1_0_Web/display.jsp?content=testevent&testType=ev
Go
Test Event: SAPEvent
bea
Application View Console
WebLogic Console
Glossary
Logout
Summary
Test
Done
Local intranet

This page allows you to test an event. You may create the event by invoking a service, or by manually creating the event.

If you want to use a service invocation to create an event, select the Service option below, and select the service to invoke. Optionally, you can create the event manually using any tools your EIS provides (for example an interactive SQL tool for the DBMS adapter used to insert a new row to create an insert event).

How do you want to create the event?

Either there are no services for this application view, or this adapter doesnt support service testing. You will need to create your event manually.

How long should we wait to receive the event?

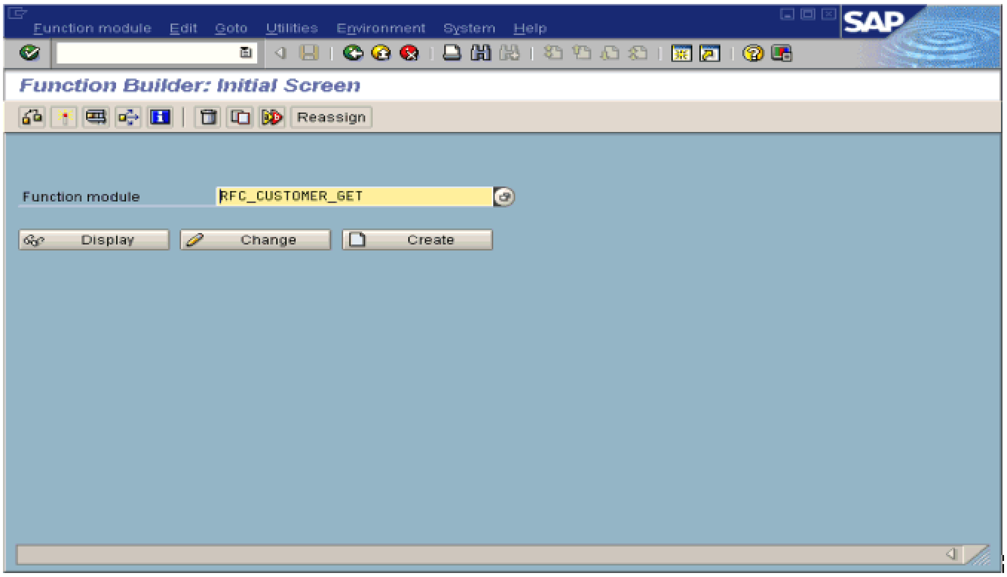
Time (in milliseconds): 30000

In the SAP Server, the transaction `/nSE37` displays the following screen where you can send RFCs to any logical system; in this case to the BEA WebLogic Adapter for SAP with an SAP event adapter configured for Program ID BEAID.

From the SAP GUI:

1. Execute transaction `/nSE37`.
2. Select a function module, for example, `RFC_CUSTOMER_GET`.

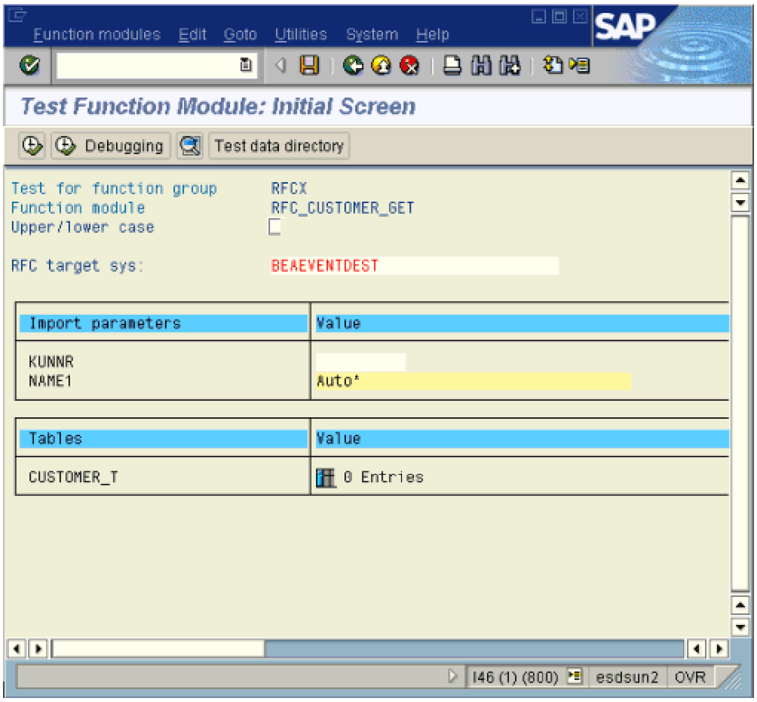
Figure 3-25 Function Builder: Initial Window



Function module RFC_CUSTOMER_GET

3. Choose single test (PF8).
4. Enter the RFC target system, for example, `BEAEVENTDEST`.
5. Enter input data for the particular RFC module; for example, `Auto* in NAME1`.
6. Execute (PF8).

Figure 3-26 Test Function Module: Initial Window



Test Function Module: Initial Screen

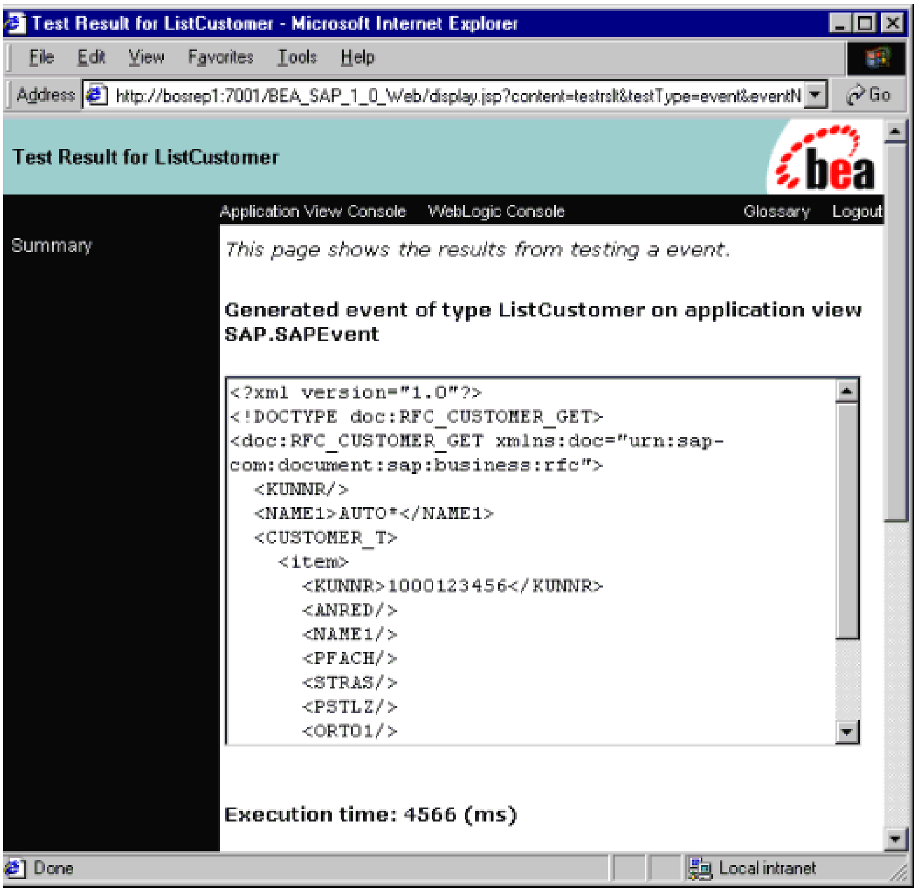
Test for function group	RFCX
Function module	RFC_CUSTOMER_GET
Upper/lower case	☐
RFC target sys:	BEAEVENTDEST

Import parameters	Value
KUNNR	
NAME1	Auto*

Tables	Value
CUSTOMER_T	0 Entries

7. A results screen appears with an RFC XML document sent to the BEA WebLogic Adapter for SAP.

Figure 3-27 Test Result for ListCustomer Window


Test Result for ListCustomer - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://bosrep1:7001/BEA_SAP_1_0_Web/display.jsp?content=testrslt&testType=event&eventN
Go
Test Result for ListCustomer
bea
Application View Console
WebLogic Console
Glossary
Logout
Summary
Done
Local intranet

This page shows the results from testing a event.

Generated event of type ListCustomer on application view SAP.SAPEvent

```
<?xml version="1.0"?>
<!DOCTYPE doc:RFC_CUSTOMER_GET>
<doc:RFC_CUSTOMER_GET xmlns:doc="urn:sap-
com:document:sap:business:rfc">
  <KUNNR/>
  <NAME1>AUTO*</NAME1>
  <CUSTOMER_T>
    <item>
      <KUNNR>1000123456</KUNNR>
      <ANRED/>
      <NAME1/>
      <PFACH/>
      <STRAS/>
      <PSTLZ/>
      <ORT01/>
```

Execution time: 4566 (ms)

You can now write custom code to exploit the adapter or create a process flow in Studio. For more information, see “Using Application Views in the Studio” in *Using Application Integration*:

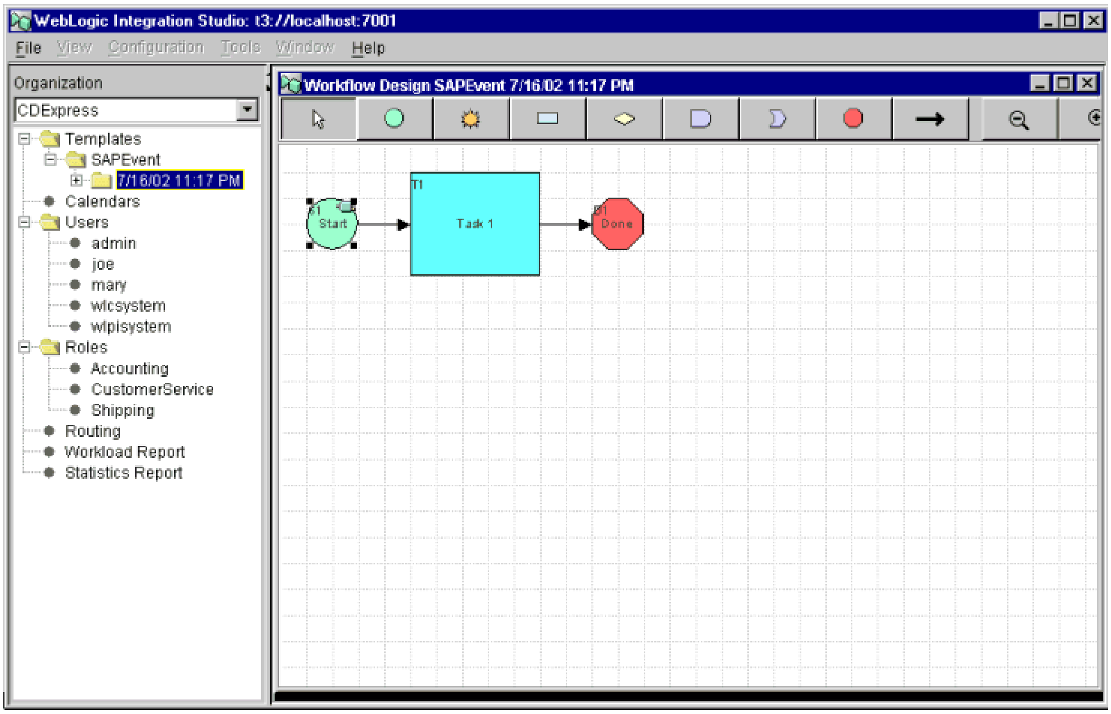
- For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/aiuser/3usruse.htm`
- For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/3usruse.htm`

Testing an Event in the Studio

To test a deployed application view event in the Studio, launch Studio and create a new template as described in “Using the Studio Interface” in *Using the WebLogic Integration Studio*:

- For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/`studio/ch2.htm
- For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/`studio/ch2.htm

Figure 3-28 New Template Window

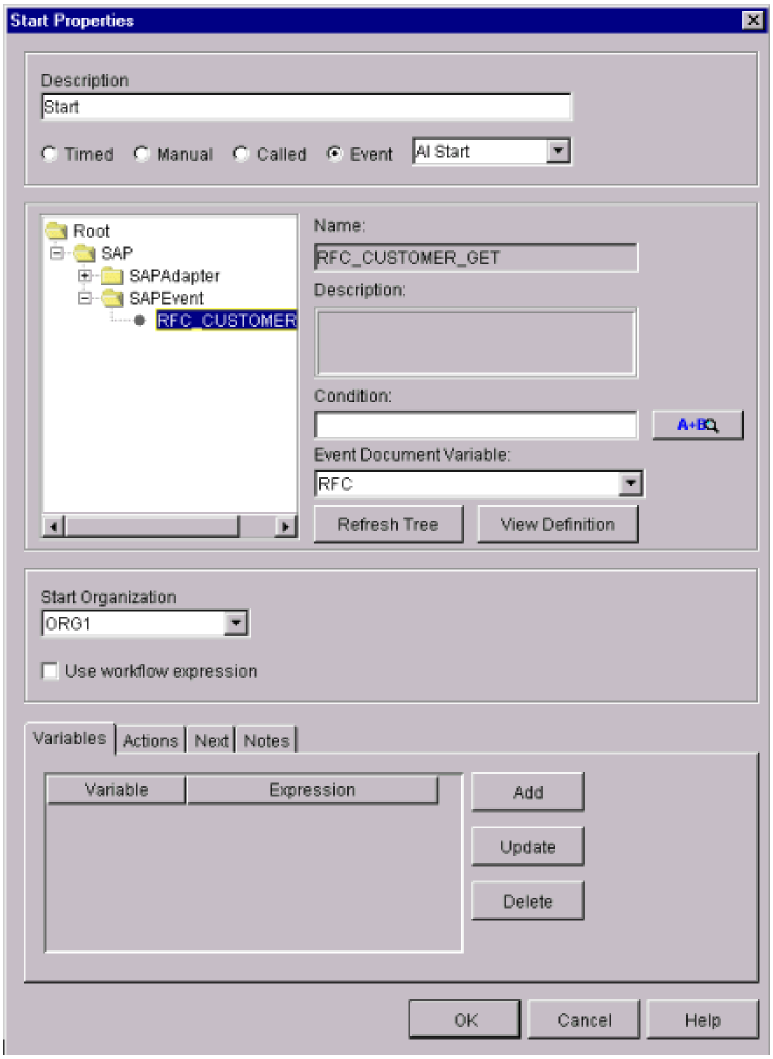


From the Start Properties form:

1. Choose Event→AI Start and select the SAP event adapter.
2. Create a <new> Event Document Variable and type a name. This variable enables you to monitor the values passed into the workflow.
3. After the workflow configuration is complete, save the template.

After you save the template, you may monitor the running instances (right-click the template and select Instances).

Figure 3-29 Start Properties Form Window



Start Properties

Description: Start

Timed · Manual · Called · [✓ Event] · AI Start

Root
- SAP
  - SAPAdapter
  - SAPEvent
    - RFC_CUSTOMER

Name: RFC_CUSTOMER_GET

Description:

Condition:

Event Document Variable: RFC

Start Organization: ORG1

Use workflow expression

Variable	Expression

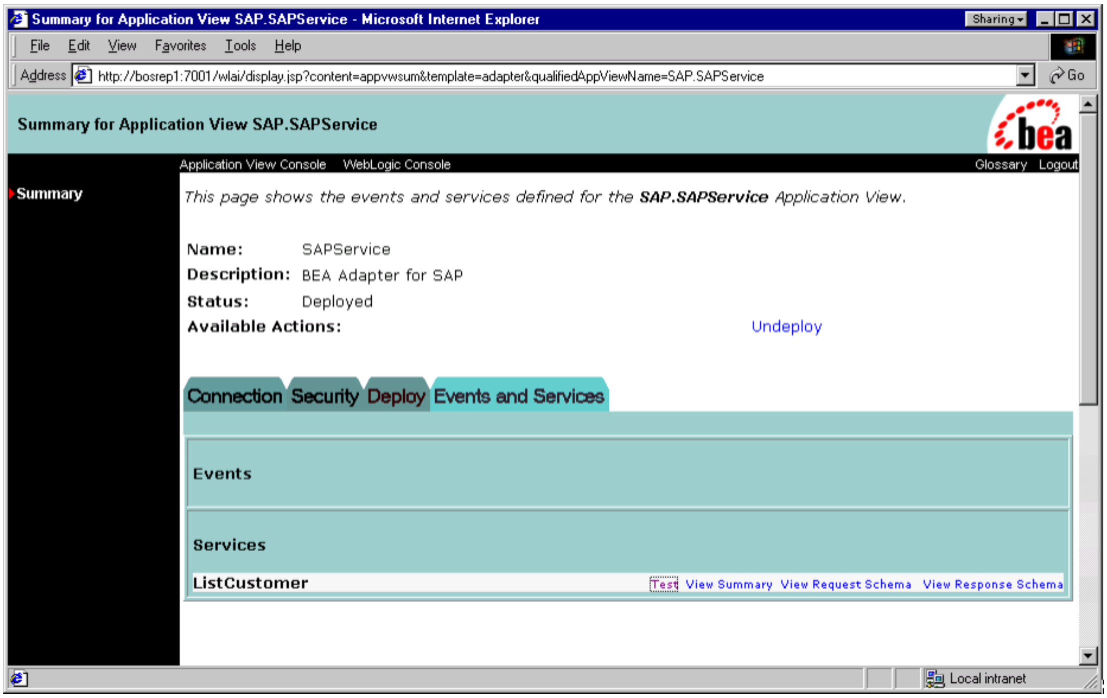
Testing a Service

After you add a service to an application view and deploy it as described in “Adding a Service to an Application View” on page 3-20 and “Deploying an Application View” on page 3-23, you can test the service.

To test a service in the Application View Console:

1. In the Events section of the Summary for Application View window, click Test.

Figure 3-30 Summary for Application View Window



Summary for Application View SAP.SAPService - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://bosrep1:7001/wlai/display.jsp?content=appvwsum&template=adapter&qualifiedAppViewName=SAP.SAPService
Go
Summary for Application View SAP.SAPService
Application View Console WebLogic Console
Glossary Logout
Summary

This page shows the events and services defined for the **SAP.SAPService** Application View.

Name: SAPService
Description: BEA Adapter for SAP
Status: Deployed
Available Actions: Undeploy

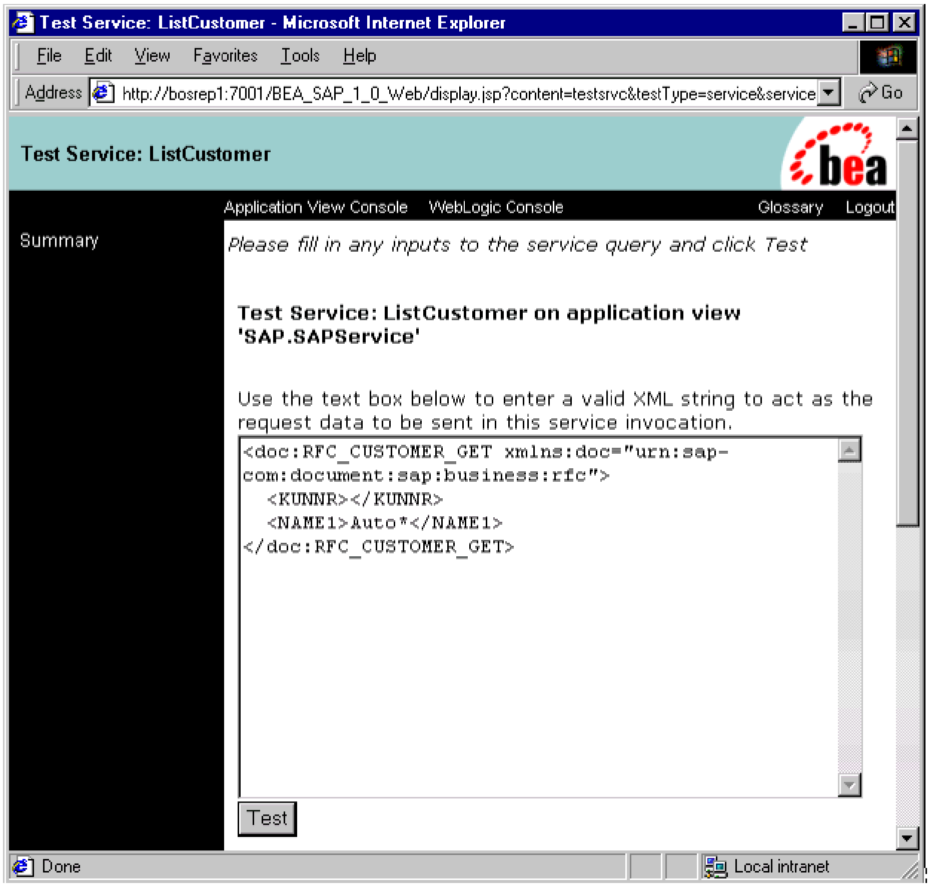
Connection Security Deploy Events and Services
Events
Services
ListCustomer Test View Summary View Request Schema View Response Schema
Local intranet

The Test Service window opens.

2. Enter a sample BEA WebLogic Adapter for SAP request, for example: `RFC_CUSTOMER_GET`:

```
<doc:RFC_CUSTOMER_GET
xmlns:doc="urn:sap-com:document:sap:business:rfc">
    <KUNNR></KUNNR>
    <NAME1>Auto*</NAME1>
</doc:RFC_CUSTOMER_GET>
```

Figure 3-31 Test Service Window


Test Service: ListCustomer - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://bosrep1:7001/BEA_SAP_1_0_Web/display.jsp?content=testsrvc&testType=service&service
Go
Test Service: ListCustomer
bea
Application View Console WebLogic Console Glossary Logout
Summary
Test
Done
Local intranet

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

Test Service: ListCustomer on application view 'SAP.SAPService'

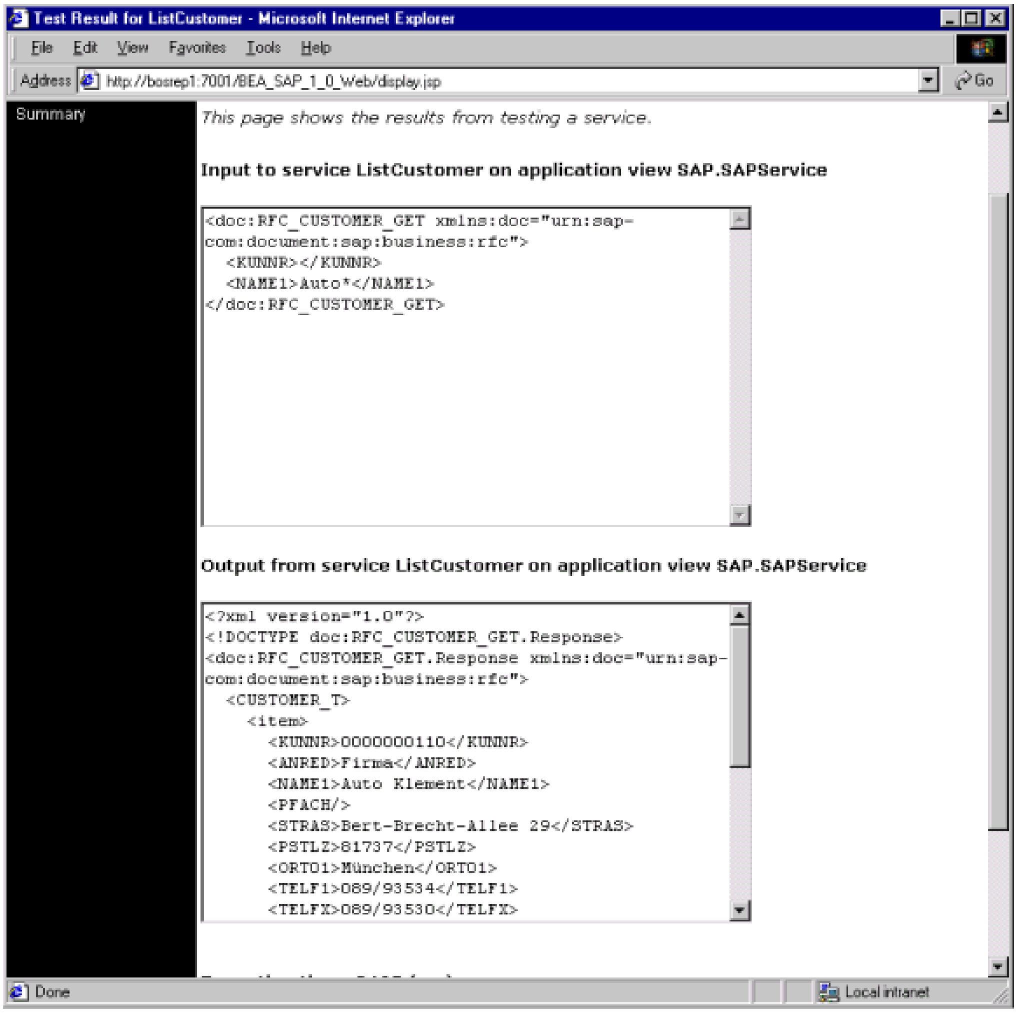
Use the text box below to enter a valid XML string to act as the request data to be sent in this service invocation.

```
<doc:RFC_CUSTOMER_GET xmlns:doc="urn:sap-
com:document:sap:business:rfc">
  <KUNNR></KUNNR>
  <NAME1>Auto*</NAME1>
</doc:RFC_CUSTOMER_GET>
```

3. Click Test to send the request through the adapter to the SAP EIS system.

   The response document should look similar to the following.

Figure 3-32 Test Results Window


Test Result for ListCustomer - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://bosrep1:7001/BEA_SAP_1_0_Web/display.jsp
Go
Summary
Done
Local intranet

This page shows the results from testing a service.

Input to service ListCustomer on application view SAP.SAPService

```
<doc:RFC_CUSTOMER_GET xmlns:doc="urn:sap-
com:document:sap:business:rfc">
  <KUNNR></KUNNR>
  <NAME1>Auto*</NAME1>
</doc:RFC_CUSTOMER_GET>
```

Output from service ListCustomer on application view SAP.SAPService

```
<?xml version="1.0"?>
<!DOCTYPE doc:RFC_CUSTOMER_GET.Response>
<doc:RFC_CUSTOMER_GET.Response xmlns:doc="urn:sap-
com:document:sap:business:rfc">
  <CUSTOMER_T>
    <item>
      <KUNNR>0000000110</KUNNR>
      <ANRED>Firma</ANRED>
      <NAME1>Auto Klement</NAME1>
      <PFACH/>
      <STRAS>Bert-Brecht-Allee 29</STRAS>
      <PSTLZ>81737</PSTLZ>
      <ORT01>München</ORT01>
      <TELF1>089/93534</TELF1>
      <TELFX>089/93530</TELFX>
```

The full response document follows.

Listing 3-1 Full Response Document from ListCustomer

```
<doc:RFC_CUSTOMER_GET.Response
xmlns:doc="urn:sap-com:document:sap:business:rfc">
   <CUSTOMER_T>
      <item>
         <KUNNR>0000000110</KUNNR>
         <ANRED>Firma</ANRED>
         <NAME1>Auto Klement</NAME1>
         <PFACH/>
         <STRAS>Bert-Brecht-Allee 29</STRAS>
         <PSTLZ>81737</PSTLZ>
         <ORT01>Mnnchen</ORT01>
         <TELF1>089/93534</TELF1>
         <TELFX>089/93530</TELFX>
      </item>
      <item>
         <KUNNR>0000001012</KUNNR>
         <ANRED>Firma</ANRED>
         <NAME1>Autohaus Franzl GmbH</NAME1>
         <PFACH/>
         <STRAS>Schwarzhauptstrasse 51</STRAS>
         <PSTLZ>80939</PSTLZ>
         <ORT01>Muenchen</ORT01>
         <TELF1>089/3546721</TELF1>
         <TELFX>089/3546722</TELFX>
      </item>
   </CUSTOMER_T>
</doc:RFC_CUSTOMER_GET.Response>
```

You can now write custom code to exploit the adapter service or create a process flow in Studio. For more information, see “Using Application Views in the Studio” in *Using Application Integration*:

- For WebLogic Integration 7.0, see
  `http://edocs.bea.com/wli/docs70/aiuser/3usruse.htm`
- For WebLogic Integration 2.1, see
  `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/3usruse.htm`

CHAPTER

4 Configuring SAP to Send IDocs to an Event

The BEA WebLogic Adapter for SAP receives IDocs from SAP using the RFC `INBOUND_IDOC_PROCESS` or `IDOC_INBOUND_ASYNCHRONOUS`. This section describes how to configure and test your SAP system to send IDocs to an event. For comprehensive information about configuring your SAP system, see your SAP documentation. This section includes the following topics:

- Defining a Logical Port
- Creating a Logical System
- Creating a Partner Profile
- Creating a Distribution Model for the Partner and Message Type
- Manually Sending an IDoc

Defining a Logical Port

The lower level networking requires that a system port number be associated with the RFC destination. The logical port identifies the port to which messages are sent. The logical port can be used only if an RFC destination was previously created.

1. In the SAP Main window, choose Tools→Business Communications→IDOCs Basis→IDOC→Port Definition, or execute transaction `WE21`.
2. Select the Transactional RFC tree item and click Create.

3. Select generate port name.

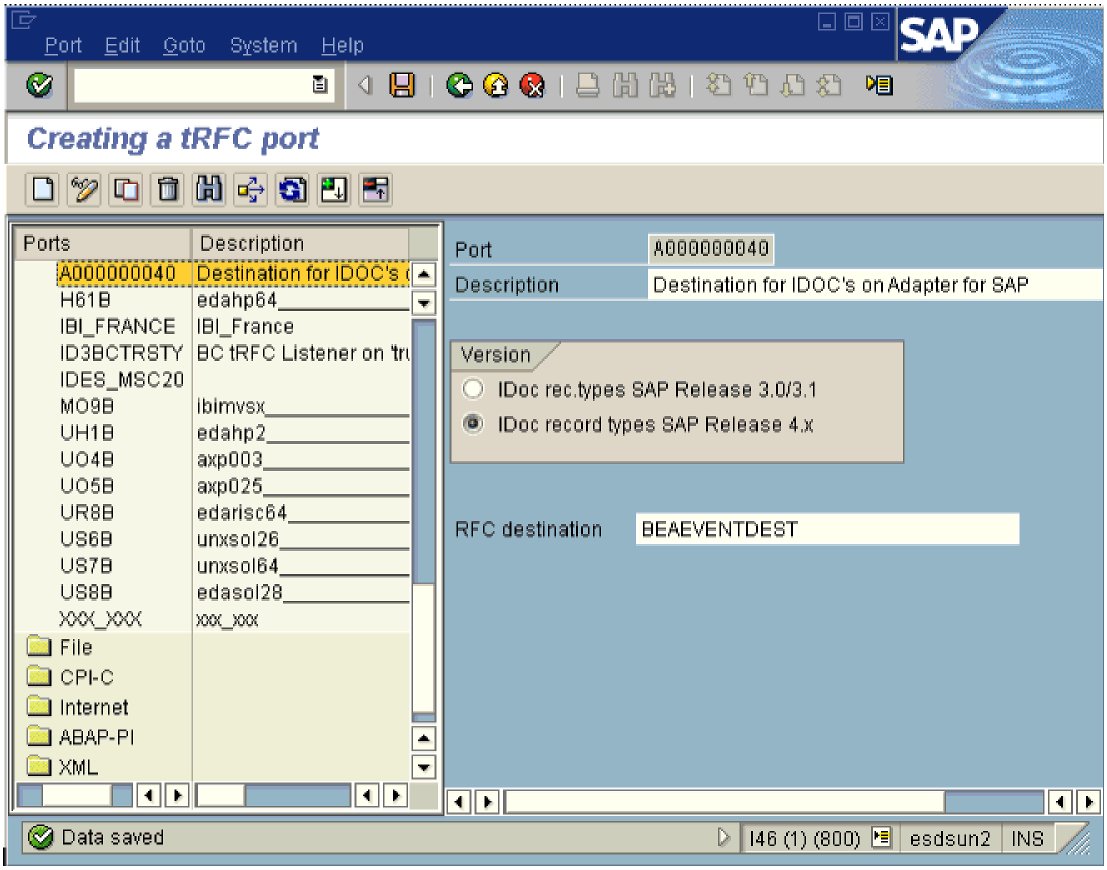
   The system generates the port name.

4. Enter the IDoc version you want to send through this port.

5. Click the destination you created, for example, `BEAEVENTDEST`.

6. Save the session, making note of the system-generated RFC port.

Figure 4-1 Creating a tRFC Port Window



Creating a tRFC port

Ports	Description
A000000040	Destination for IDOC's (
H61B	edahp64
IBI_FRANCE	IBI_France
ID3BCTRSTY	BC tRFC Listener on 'tr
IDES_MSC20	
MO9B	ibimvsx
UH1B	edahp2
UO4B	axp003
UO5B	axp025
UR8B	edarisc64
US6B	unxsol26
US7B	unxsol64
US8B	edasol28
XXX_XXX	xxx_xxx

File
CPI-C
Internet
ABAP-PI
XML

Port: A000000040
Description: Destination for IDOC's on Adapter for SAP

Version
- ( ) IDoc rec.types SAP Release 3.0/3.1
- [✓ IDoc record types SAP Release 4.x]

RFC destination: BEAEVENTDEST

Data saved   I46 (1) (800)   esdsun2   INS

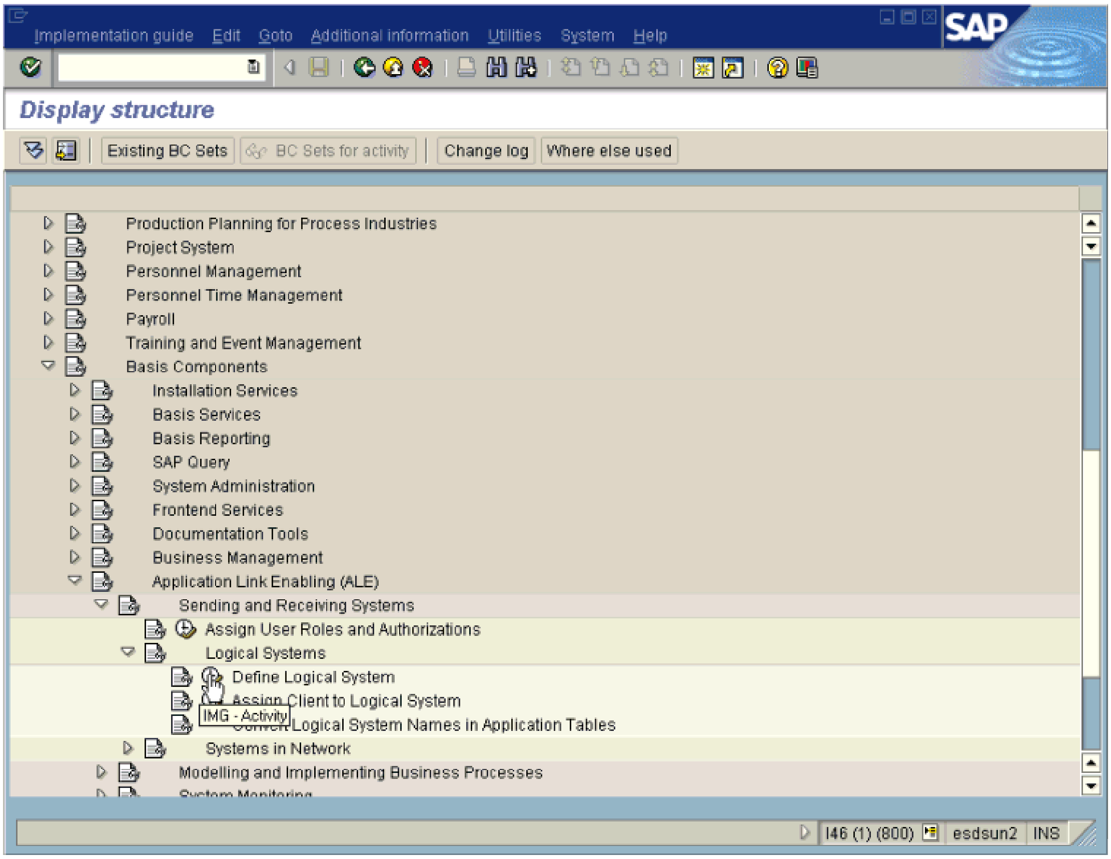
Creating a Logical System

One type of partner is a logical system. A logical system manages one or more RFC destinations. To create a logical system called `BEALOG`:

1. In the SAP Main screen, choose Tools→AcceleratedSAP→Customizing→Project Management (transaction `SPRO_ADMIN`), or else execute transaction `SPRO`.
2. Select SAP Reference IMG.

3. Expand the following nodes: Basis Components→Application Link Enabling (ALE)→Sending and Receiving Systems→Logical Systems→Define Logical System. Click the green hook beside Define Logical System.

Figure 4-2 Display Structure Window

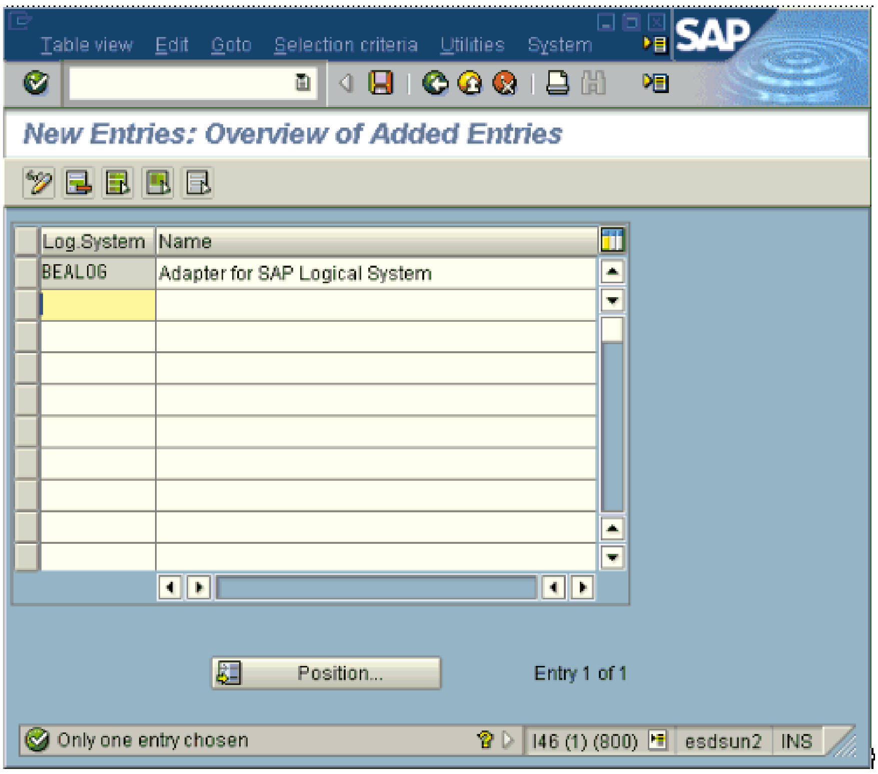


Production Planning for Process Industries
Project System
Personnel Management
Personnel Time Management
Payroll
Training and Event Management
Basis Components
Installation Services
Basis Services
Basis Reporting
SAP Query
System Administration
Frontend Services
Documentation Tools
Business Management
Application Link Enabling (ALE)
Sending and Receiving Systems
Assign User Roles and Authorizations
Logical Systems
Define Logical System
Assign Client to Logical System
Convert Logical System Names in Application Tables
Systems in Network
Modelling and Implementing Business Processes
System Monitoring

4. Select New Entries.

5. Enter a meaningful name for your partner (for example, `BEALOG`) and provide a short description.

Figure 4-3 New Entries: Overview of Added Entries Window



New Entries: Overview of Added Entries

Log.System	Name
BEALOG	Adapter for SAP Logical System

Entry 1 of 1

Only one entry chosen

I46 (1) (800) esdsun2 INS

6. Save the session.

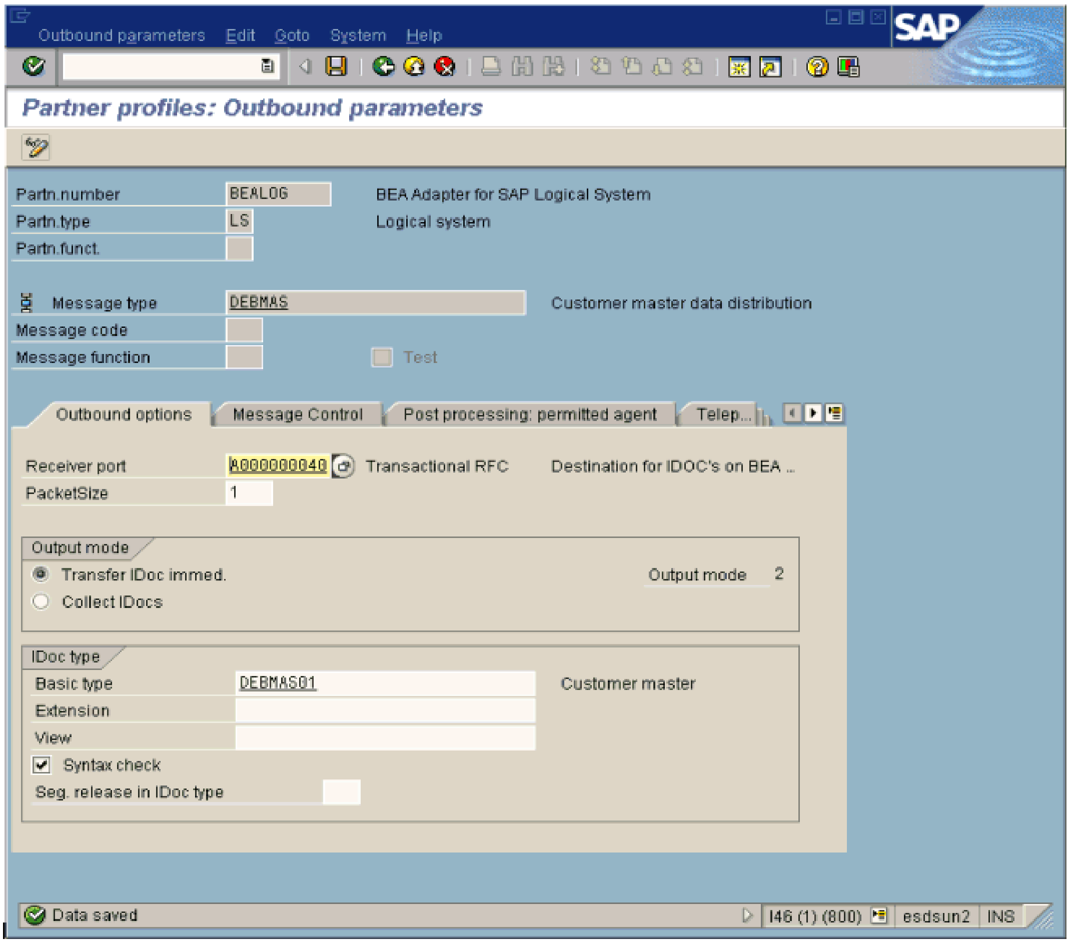
Creating a Partner Profile

To create a partner profile:

1. In the SAP Main screen, choose Tools→Business Communication→IDOC Basis→IDOC→Partner profile, or else execute transaction `WE20`.
2. Select Partner type LS (Logical system) and select Create (F5).
3. Enter Type as USER and enter Agent as OMNI (this is the user ID of the SAP system).
4. Select Create outbound parameter below the outbound parameter table control.
5. Partn.type is LS, Message Type is DEBMAS (this is the IDoc document type), and leave Partn.funct blank.
6. Select the Outbound options tab.
7. Select Transfer IDOCs Immed.
8. Enter message type of the IDoc (for example, DEBMAS).
9. Enter receiver port for example (A000000040 from the previous example screens).

10. Save the session.

Figure 4-4 Partner Profiles: Outbound Parameters Window



Partner profiles: Outbound parameters

Partn.number	BEALOG	BEA Adapter for SAP Logical System
Partn.type	LS	Logical system
Partn.funct.		
Message type	DEBMAS	Customer master data distribution
Message code		
Message function		Test

Outbound options | Message Control | Post processing: permitted agent | Telep...

Receiver port	A000000040	Transactional RFC	Destination for IDOC's on BEA ...
PacketSize	1		

Output mode
- ● Transfer IDoc immed.
- ○ Collect IDocs

Output mode 2

IDoc type

Basic type	DEBMAS01	Customer master
Extension		
View		

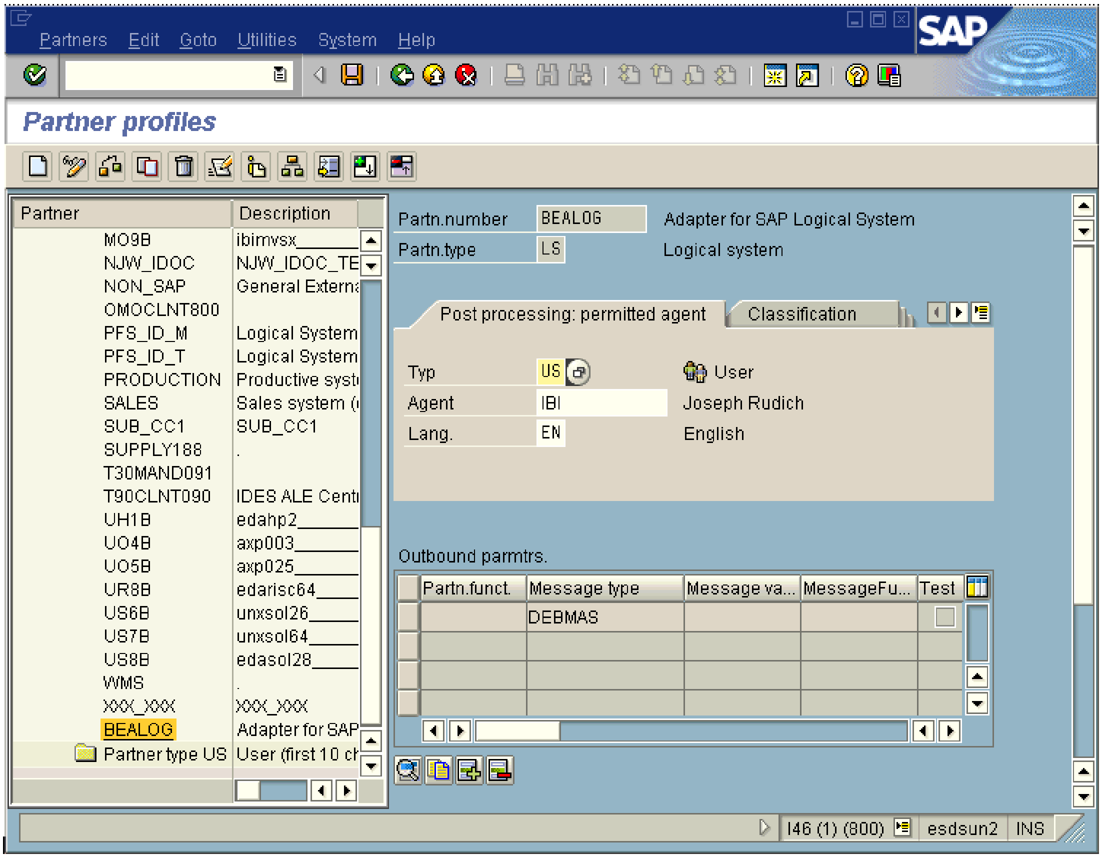
✓ Syntax check

Seg. release in IDoc type

Data saved | I46 (1) (800) | esdsun2 | INS

11. Exit the session. The SAP Partner Profiles summary window opens, displaying information for the logical system that you just created.

Figure 4-5 Partner Profiles Summary Window



Partner profiles

Partner	Description
MO9B	ibimvsx_
NJW_IDOC	NJW_IDOC_TE
NON_SAP	General Externa
OMOCLNT800	
PFS_ID_M	Logical System
PFS_ID_T	Logical System
PRODUCTION	Productive syst
SALES	Sales system (
SUB_CC1	SUB_CC1
SUPPLY188	.
T30MAND091	
T90CLNT090	IDES ALE Centr
UH1B	edahp2_
UO4B	axp003_
UO5B	axp025_
UR8B	edarisc64_
US6B	unxsol26_
US7B	unxsol64_
US8B	edasol28_
WMS	.
XXX_XXX	XXX_XXX
BEALOG	Adapter for SAP
Partner type US	User (first 10 ch

Partn.number BEALOG Adapter for SAP Logical System
Partn.type LS Logical system

Post processing: permitted agent | Classification

Typ	US	User
Agent	IBI	Joseph Rudich
Lang.	EN	English

Outbound parmtrs.

Partn.funct.	Message type	Message va...	MessageFu...	Test
	DEBMAS			☐

I46 (1) (800) esdsun2 INS

Creating a Distribution Model for the Partner and Message Type

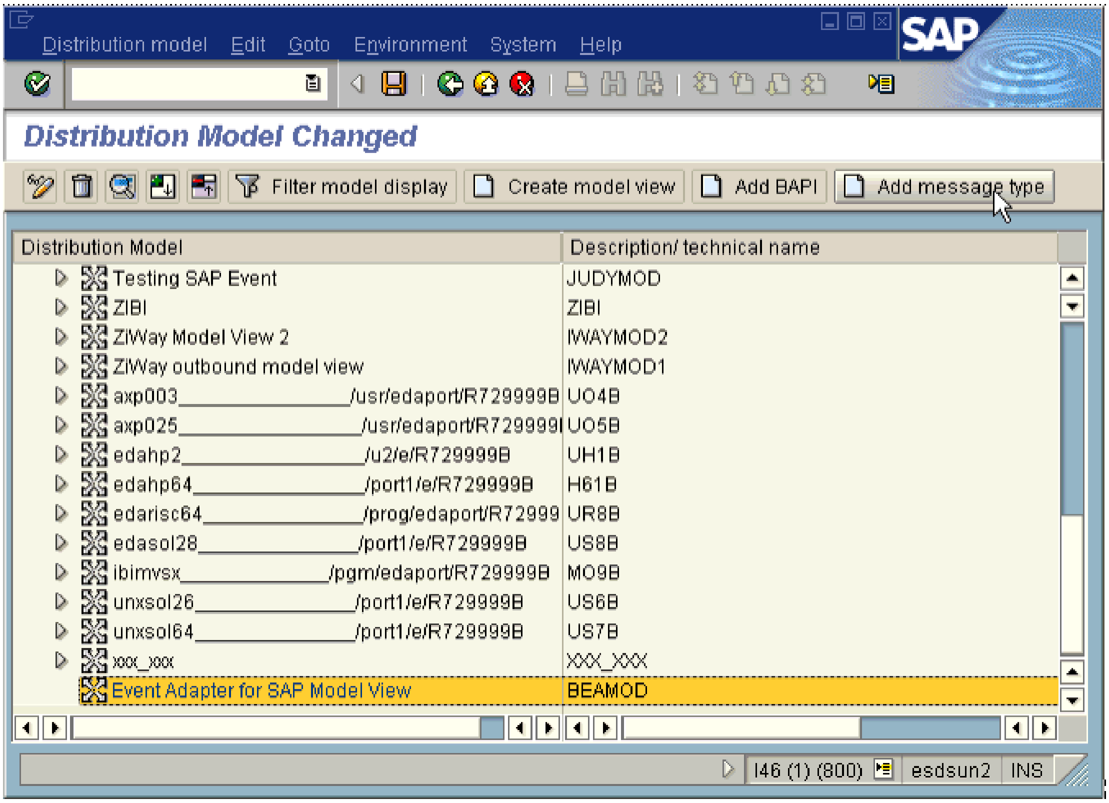
To create a distribution model called `BEAMOD`:

1. In the SAP Main screen, choose Tools→AcceleratedSAP→Customizing→Project Management, or else execute transaction `BD64`.

   The Display Distribution Model window opens.

2. Select Create model view. (You may need to switch processing mode to edit, within Distribution Model/Switch Processing).
3. Enter a short text string and a technical name for your new model view.
4. Select your new model view in the tree Distribution Model and select Add message type.

Figure 4-6 Distribution Model Changed Window



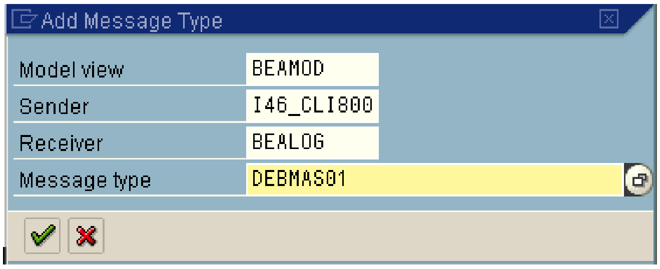
Distribution Model	Description/ technical name
Testing SAP Event	JUDYMOD
ZIBI	ZIBI
ZiWay Model View 2	IWAYMOD2
ZiWay outbound model view	IWAYMOD1
axp003_______________/usr/edaport/R729999B	UO4B
axp025_______________/usr/edaport/R729999I	UO5B
edahp2_______________/u2/e/R729999B	UH1B
edahp64_______________/port1/e/R729999B	H61B
edarisc64_______________/prog/edaport/R72999	UR8B
edasol28_______________/port1/e/R729999B	US8B
ibimvsx_______________/pgm/edaport/R729999B	MO9B
unxsol26_______________/port1/e/R729999B	US6B
unxsol64_______________/port1/e/R729999B	US7B
xxx_xxx	XXX_XXX
Event Adapter for SAP Model View	BEAMOD

I46 (1) (800) esdsun2 INS

In the dialog box, you can specify:

- Sender: for example, I46_CLI800, which points to the SAP system that will send the IDoc (in this case, an SAP 4.6B system).
- Receiver: Logical system. For example, `BEALOG`.
- Message type: Type of IDoc. For example, `DEBMAS`.

Figure 4-7 Add Message Type Dialog Box

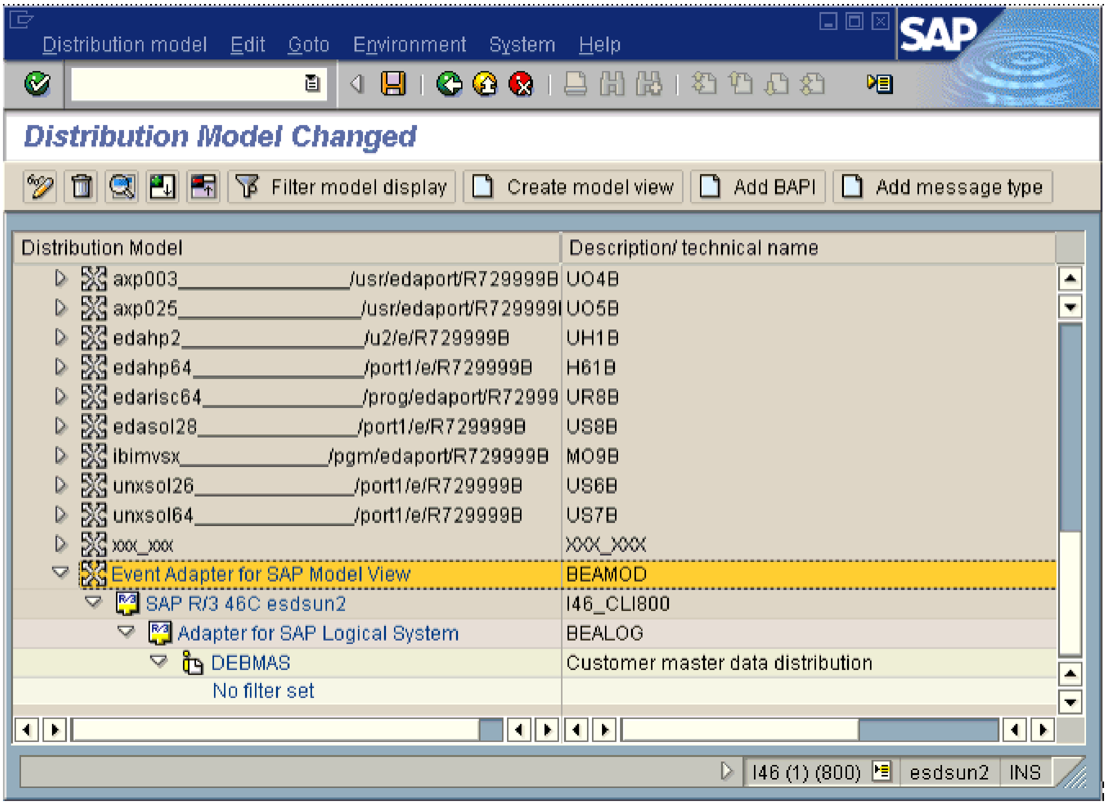


Add Message Type

Model view	BEAMOD
Sender	I46_CLI800
Receiver	BEALOG
Message type	DEBMAS01

The following screen shows the new model view that can be used to send message type DEBMAS from the I46_CLI800 SAP system to the `BEALOG` logical system.

Figure 4-8 Updated Distribution Model Changed Window



Distribution Model Changed

Distribution Model	Description/ technical name
axp003_______________/usr/edaport/R729999B	UO4B
axp025________________/usr/edaport/R729999I	UO5B
edahp2________________/u2/e/R729999B	UH1B
edahp64_______________/port1/e/R729999B	H61B
edarisc64______________/prog/edaport/R72999	UR8B
edasol28______________/port1/e/R729999B	US8B
ibimvsx______________/pgm/edaport/R729999B	MO9B
unxsol26______________/port1/e/R729999B	US6B
unxsol64______________/port1/e/R729999B	US7B
xxx_xxx	XXX_XXX
Event Adapter for SAP Model View	BEAMOD
SAP R/3 46C esdsun2	I46_CLI800
Adapter for SAP Logical System	BEALOG
DEBMAS	Customer master data distribution
No filter set	

I46 (1) (800) esdsun2 INS

You are now ready to test the connection to the WebLogic Server, as described in "Manually Sending an IDoc" on page 4-13.

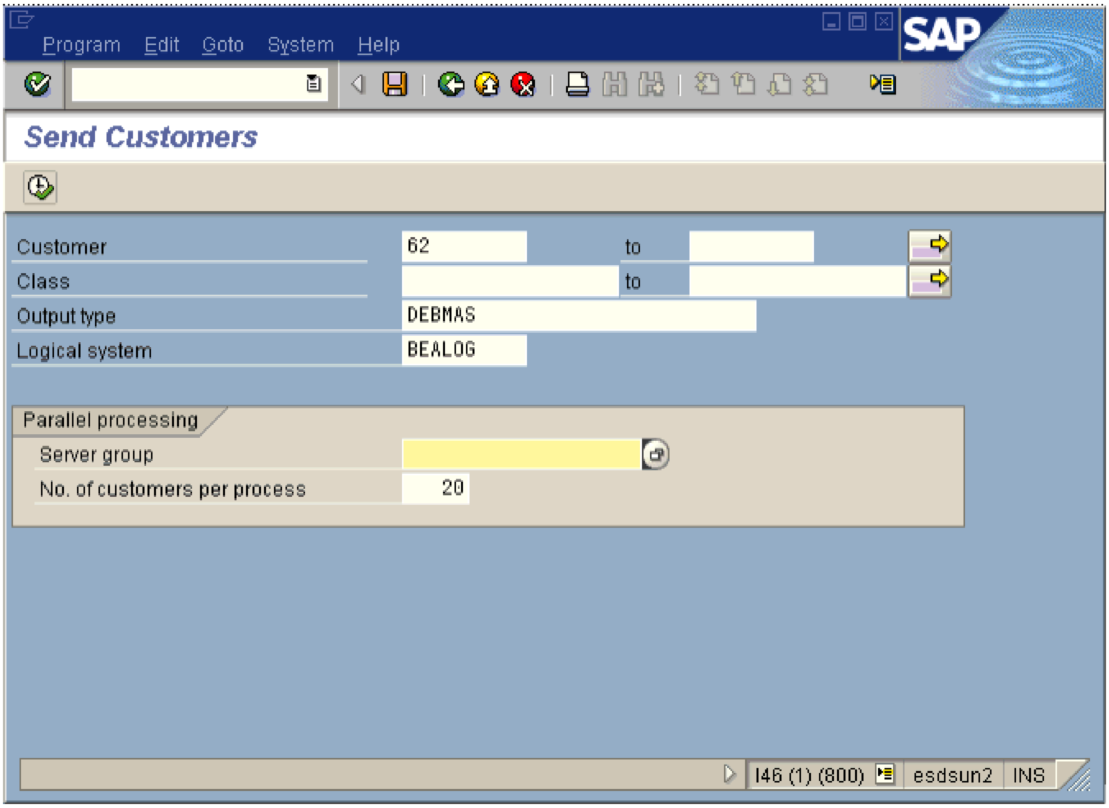
Manually Sending an IDoc

In the SAP Server, the transaction `BD12` brings you to the following screen where you can send IDocs to any logical system, in this example to WebLogic Integration with an SAP event listener (RFC listener) for program ID `BEAID`.

1. Configure the SAP listener.
2. Use the BEA Application Explorer to create appropriate schemas.
3. Enter the IDoc message type DEBMAS in the Output type field.
4. Enter the logical system (for example, `BEALOG`).
5. Click Run (transfer data).

6. The SAP listener receives the IDoc in XML format. No response is expected from the listener.

Figure 4-9 Send Customers Window



Send Customers

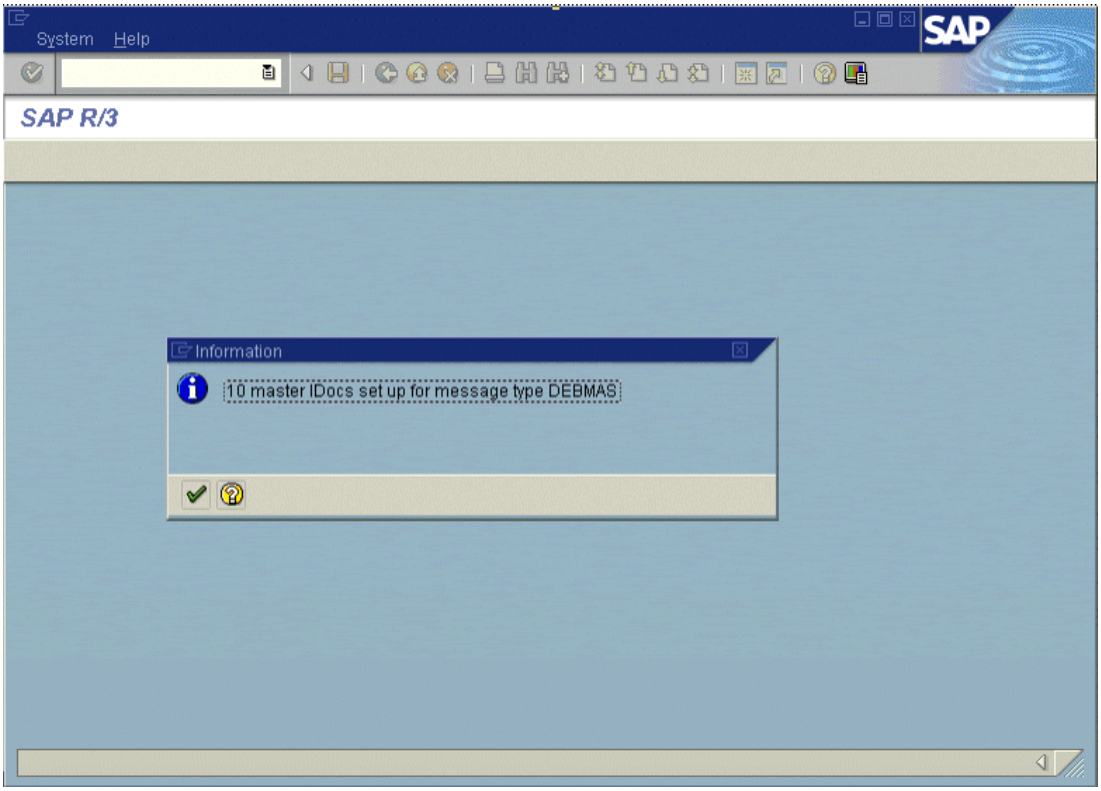
Customer	62	to	
Class		to	
Output type	DEBMAS		
Logical system	BEALOG		

Parallel processing

Server group	
No. of customers per process	20

I46 (1) (800) esdsun2 INS

Figure 4-10 Master IDocs Set Up for Message Type DEBMAS Window



10 master IDocs set up for message type DEBMAS

A Sample Files

This section provides sample request and response documents sent between SAP and the BEA WebLogic Adapter for SAP, as well as a sample RFC module. It includes:

- Sample RFC Request Document
- Sample RFC Response Document
- Sample IDoc XML for Message Type DEBMAS
- Sample RFC Module

Sample RFC Request Document

Listing A-1 Sample RFC Request Document

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

<doc:RFC_WALK_THRU_TEST
xmlns:doc="urn:sapcom:document:sap:business:rfc">

   <TEST_IN>

      <RFCFLOAT>0.0</RFCFLOAT>
      <RFCCHAR1></RFCCHAR1>
      <RFCINT2>0</RFCINT2>
      <RFCINT1>0</RFCINT1>
      <RFCCHAR4></RFCCHAR4>
      <RFCINT4>10</RFCINT4>
      <RFCHEX3>000000</RFCHEX3>
      <RFCCHAR2></RFCCHAR2>
      <RFCTIME>10:09:32</RFCTIME>
      <RFCDATE>2001-09-05</RFCDATE>
      <RFCDATA1>Hello World</RFCDATA1>
      <RFCDATA2></RFCDATA2>
   </TEST_IN>
   <DESTINATIONS>
   </DESTINATIONS>
   <LOG>
   </LOG>
</doc:RFC_WALK_THRU_TEST>
```

Sample RFC Response Document

Listing A-2 Sample RFC Response Document

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

<doc:RFC_WALK_THRU_TEST.Response
xmlns:doc="urn:sapcom:document:sap:business:rfc">

   <TEST_OUT>

      <RFCFLOAT>0.0</RFCFLOAT>
      <RFCCHAR1></RFCCHAR1>
      <RFCINT2>0</RFCINT2>
      <RFCINT1>0</RFCINT1>
      <RFCCHAR4></RFCCHAR4>
      <RFCINT4>10</RFCINT4>
      <RFCHEX3>000000</RFCHEX3>
      <RFCCHAR2></RFCCHAR2>
      <RFCTIME>10:09:32</RFCTIME>
      <RFCDATE>2001-09-05</RFCDATE>
      <RFCDATA1>Hello World</RFCDATA1>
      <RFCDATA2></RFCDATA2>
   </TEST_OUT>
   <DESTINATIONS>
   </DESTINATIONS>
   <LOG>
   </LOG>
</doc:RFC_WALK_THRU_TEST.Response>
```

Sample IDoc XML for Message Type DEBMAS

Listing A-3 Sample IDoc XML for Message Type DEBMAS

```
<?xml version="1.0" ?>
<DEBMAS01>
   <IDOC BEGIN="1">
      <EDI_DC40 SEGMENT="1">
         <TABNAM>EDI_DC40</TABNAM>
         <MANDT>800</MANDT>
<DOCNUM>0000000000236015</DOCNUM>
         <DOCREL>46C</DOCREL>
         <STATUS>30</STATUS>
         <DIRECT>1</DIRECT>
         <OUTMOD>2</OUTMOD>
         <EXPRSS></EXPRSS>
         <TEST></TEST>
         <IDOCTYP>DEBMAS01</IDOCTYP>
         <CIMTYP></CIMTYP>
         <MESTYP>DEBMAS</MESTYP>
         <MESCOD></MESCOD>
         <MESFCT></MESFCT>
         <STD></STD>
         <STDVRS></STDVRS>
         <STDMES></STDMES>
         <SNDPOR>SAPI46</SNDPOR>
         <SNDPRT>LS</SNDPRT>
         <SNDPFC></SNDPFC>
         <SNDPRN>I46_CLI800</SNDPRN>
         <SNDSAD></SNDSAD>
         <SNDLAD></SNDLAD>
         <RCVPOR>A000000018</RCVPOR>
         <RCVPRT>LS</RCVPRT>
         <RCVPFC></RCVPFC>
         <RCVPRN>SAMP</RCVPRN>
         <RCVSAD></RCVSAD>
         <RCVLAD></RCVLAD>
         <CREDAT>2001-09-04</CREDAT>
         <CRETIM>16:44:52</CRETIM>
         <REFINT></REFINT>
         <REFGRP></REFGRP>
         <REFMES></REFMES>
```

```
    <ARCKEY></ARCKEY>
    <SERIAL>20010904164452</SERIAL>
</EDI_DC40>
<E1KNA1M SEGMENT="1">
   <MSGFN>005</MSGFN>
   <KUNNR>0000000001</KUNNR>
   <ANRED></ANRED>
   <AUFSD></AUFSD>
   <BAHNE></BAHNE>
   <BAHNS></BAHNS>
   <BBBNR>0000000</BBBNR>
   <BBSNR>00000</BBSNR>
   <BEGRU></BEGRU>
   <BRSCH></BRSCH>
   <BUBKZ>0</BUBKZ>
   <DATLT></DATLT>
   <FAKSD></FAKSD>
   <FISKN></FISKN>
   <KNRZA></KNRZA>
   <KONZS></KONZS>
   <KTOKD>0001</KTOKD>
   <KUKLA></KUKLA>
   <LAND1>US</LAND1>
   <LIFNR></LIFNR>
   <LIFSD></LIFSD>
   <LOCCO></LOCCO>
   <LOEVM></LOEVM>
   <NAME1>Apple Corp</NAME1>
   <NAME2></NAME2>
   <NAME3></NAME3>
   <NAME4></NAME4>
   <NIELS></NIELS>
   <ORT01>Floral Park</ORT01>
   <ORT02></ORT02>
   <PFACH></PFACH>
   <PSTL2></PSTL2>
   <PSTLZ>10010</PSTLZ>
   <REGIO>NY</REGIO>
   <COUNC></COUNC>
   <CITYC></CITYC>
   <RPMKR></RPMKR>
   <SORTL>APPLE</SORTL>
   <SPERR></SPERR>
   <SPRAS>E</SPRAS>
   <STCD1></STCD1>
   <STCD2></STCD2>
  <STKZA></STKZA>
  <STKZU></STKZU>
  <STRAS>123 Main street</STRAS>
```

```
<TELBX></TELBX>
<TELF1></TELF1>
<TELF2></TELF2>
<TELFX></TELFX>
<TELTX></TELTX>
<TELX1></TELX1>
<LZONE>0000000001</LZONE>
<XZEMP></XZEMP>
<VBUND></VBUND>
<STCEG></STCEG>
<GFORM></GFORM>
<BRAN1></BRAN1>
<BRAN2></BRAN2>
<BRAN3></BRAN3>
<BRAN4></BRAN4>
<BRAN5></BRAN5>
<UMJAH>0000</UMJAH>
<UWAER></UWAER>
<JMZAH>000000</JMZAH>
<JMJAH>0000</JMJAH>
<KATR1></KATR1>
<KATR2></KATR2>
<KATR3></KATR3>
<KATR4></KATR4>
<KATR5></KATR5>
<KATR6></KATR6>
<KATR7></KATR7>
<KATR8></KATR8>
<KATR9></KATR9>
<KATR10></KATR10>
<STKZN></STKZN>
<UMSA1>0</UMSA1>
<TXJCD></TXJCD>
<PERIV></PERIV>
<KTOCD></KTOCD>
<PFORT></PFORT>
<DTAMS></DTAMS>
<DTAWS></DTAWS>
<HZUOR>00</HZUOR>
<CIVVE>X</CIVVE>
<MILVE></MILVE>
<SPRAS_ISO>EN</SPRAS_ISO>
<FITYP></FITYP>
<STCDT></STCDT>
<STCD3></STCD3>
<STCD4></STCD4>
<XICMS></XICMS>
<CFOPC></CFOPC>
<TXLW1></TXLW1>
```

```
  <TXLW2></TXLW2>
  <CCC01></CCC01>
  <CCC02></CCC02>
  <CCC03></CCC03>
  <CCC04></CCC04>
  <CASSD></CASSD>
  <KDKG1></KDKG1>
  <KDKG2></KDKG2>
  <KDKG3></KDKG3>
  <KDKG4></KDKG4>
  <KDKG5></KDKG5>
  <NODEL></NODEL>
  <XSUB2></XSUB2>
 <WERKS></WERKS>
<E1KNVVM SEGMENT="1">
   <MSGFN>005</MSGFN>
   <VKORG>0001</VKORG>
   <VTWEG>01</VTWEG>
   <SPART>01</SPART>
  <BEGRU></BEGRU>
  <LOEVM></LOEVM>
  <VERSG></VERSG>
  <AUFSD></AUFSD>
  <KALKS>1</KALKS>
  <KDGRP></KDGRP>
  <BZIRK></BZIRK>
  <KONDA></KONDA>
  <PLTYP></PLTYP>
  <AWAHR>100</AWAHR>
  <INCO1></INCO1>
  <INCO2></INCO2>
  <LIFSD></LIFSD>
  <AUTLF></AUTLF>
  <ANTLF>9</ANTLF>
  <KZTLF></KZTLF>
  <KZAZU>X</KZAZU>
  <CHSPL></CHSPL>
  <LPRIO>00</LPRIO>
  <EIKTO></EIKTO>
  <VSBED>01</VSBED>
  <FAKSD></FAKSD>
  <MRNKZ></MRNKZ>
  <PERFK></PERFK>
  <PERRL></PERRL>
  <WAERS>EUR</WAERS>
  <KTGRD></KTGRD>
   <ZTERM></ZTERM>
   <VWERK></VWERK>
   <VKGRP></VKGRP>
```

```
<VKBUR></VKBUR>
<VSORT></VSORT>
<KVGR1></KVGR1>
<KVGR2></KVGR2>
<KVGR3></KVGR3>
<KVGR4></KVGR4>
<KVGR5></KVGR5>
<BOKRE></BOKRE>
<KURST></KURST>
<PRFRE></PRFRE>
<KLABC></KLABC>
<KABSS></KABSS>
<KKBER></KKBER>
<CASSD></CASSD>
<RDOFF></RDOFF>
<AGREL></AGREL>
<MEGRU></MEGRU>
<UEBTO>0.0</UEBTO>
<UNTTO>0.0</UNTTO>
<UEBTK></UEBTK>
<PVKSM></PVKSM>
<PODKZ></PODKZ>
<PODTG>         0</PODTG>
<E1KNVPM SEGMENT="1">
    <MSGFN>005</MSGFN>
    <PARVW>AG</PARVW>
    <KUNN2>0000000001</KUNN2>
    <DEFPA></DEFPA>
    <KNREF></KNREF>
    <PARZA>000</PARZA>
</E1KNVPM>
<E1KNVPM SEGMENT="1">
    <MSGFN>005</MSGFN>
    <PARVW>RE</PARVW>
    <KUNN2>0000000001</KUNN2>
    <DEFPA></DEFPA>
    <KNREF></KNREF>
    <PARZA>000</PARZA>
</E1KNVPM>
<E1KNVPM SEGMENT="1">
    <MSGFN>005</MSGFN>
    <PARVW>RG</PARVW>
    <KUNN2>0000000001</KUNN2>
    <DEFPA></DEFPA>
    <KNREF></KNREF>
    <PARZA>000</PARZA>
</E1KNVPM>
<E1KNVPM SEGMENT="1">
    <MSGFN>005</MSGFN>
```

```
            <PARVW>WE</PARVW>
            <KUNN2>0000000001</KUNN2>
            <DEFPA></DEFPA>
            <KNREF></KNREF>
            <PARZA>000</PARZA>
        </E1KNVPM>
        <E1KNVIM SEGMENT="1">
             <MSGFN>005</MSGFN>
             <ALAND>DE</ALAND>
             <TATYP>MWST</TATYP>
             <TAXKD>0</TAXKD>
        </E1KNVIM>
     </E1KNVVM>
   </E1KNA1M>
  </IDOC>
</DEBMAS01
>
```

Sample RFC Module

Once the you have configured an event and RFC destination, you can write ABAP code to execute calls at your new destination (that is, the application view event).

The following is sample code that makes use of a user-defined RFC module named `Z_EVENT_DISPATCH`.

Listing 1-4 Sample Code With User-Defined RFC

```
FUNCTION Z_01_EVENT_DISPATCH.
CALL FUNCTION 'Z_EVENT_DISPATCH'
  DESTINATION 'BEADEST'
  EXPORTING
    EVENT = EVENT
    RECTYPE = RECTYPE
    OBJTYPE = OBJTYPE
    OBJKEY = OBJKEY
  TABLES
    EVENT_CONTAINER = EVENT_CONTAINER.
ENDFUNCTION.
```