BEA WebLogic Adapter for Siebel User's Guide
Version 5.5.011
For WebLogic Server 9.1

DN3501345.0706

July 27, 2006

Preface

This document is written for system integrators who develop client interfaces between Siebel and other applications.

How This Manual Is Organized

The following table lists and describes the chapters and appendixes in this manual.

Chapter/Appendix		Contents
1	Introducing the BEA WebLogic Adapter for Siebel	Explains how the BEA WebLogic Adapter for Siebel integrates with Siebel. This section also describes the Siebel architecture.
2	Creating XML Schemas and Integration Business Services	Describes how to create XML schemas and business services for Siebel objects using the Integration Business Services Engine (iBSE).
3	Listening for Siebel Events	Describes how to use the BEA WebLogic Adapter for Siebel to listen for events in a Siebel system.
4	Management and Monitoring	Describes how to configure and use monitoring tools provided by iBSE and JCA to gauge the performance of your run-time environment.
5	Troubleshooting and Error Messages	Explains the limitations and workarounds when connecting to Siebel.
A	Usage Considerations and Sample Files	Provides sample schemas for Siebel Business Components and Siebel Business Services.
B	Siebel Workflows	Describes Siebel topics relating to the processing of Siebel Integration Objects using Siebel XML.
C	Best Practices and Performance Considerations	This topic provides best practices and performance considerations for the BEA WebLogic Adapter for Siebel when deployed to the BEA WebLogic Server.

Documentation Conventions

Delete the items that do not apply to your manual and/or add special conventions that are unique to your manual.

The following table lists and describes the conventions that apply throughout this manual.

Convention	Description
`THIS TYPEFACE` or `this typeface`	Denotes syntax that you must enter exactly as shown.
`this typeface`	Represents a placeholder (or variable) in syntax for a value that you or the system must supply.
`underscore`	Indicates a default setting.
this typeface	Represents a placeholder (or variable) in a text paragraph, a cross-reference, or an important term.
this typeface	Highlights a file name or command in a text paragraph that must be lowercase.
this typeface	Indicates a button, menu item, or dialog box option you can click or select.
Key + Key	Indicates keys that you must press simultaneously.
`{ }`	Indicates two or three choices; type one of them, not the braces.
`|`	Separates mutually exclusive choices in syntax. Type one of them, not the symbol.
`...`	Indicates that you can enter a parameter multiple times. Type only the parameter, not the ellipsis points (…).
`.` `.` `.`	Indicates that there are (or could be) intervening or additional commands.

Contact Us!

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

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

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

Help Us to Serve You Better

To help our consultants answer your questions effectively, please be prepared to provide specifications and sample files and to answer questions about errors and problems.

The following tables list the specifications our consultants require.

Platform	
Operating System	
OS Version	
Product List	
Adapters	
Adapter Deployment	For example, JCA, Integration Business Services Engine
Container Version	

The following table lists components. Specify the version in the column provided.

Component	Version
Adapter	
EIS (DBMS/APP)	
HOTFIX / Service Pack	

The following table lists the types of Application Explorer. Specify the version (and platform, if different than listed previously) in the columns provided.

Application Explorer Type	Version	Platform
Swing		
Servlet		
ASP		

In the following table, specify the JVM version and vendor in the columns provided.

Version	Vendor

The following table lists additional questions to help us serve you better.

Request/Question	Error/Problem Details or Information
Provide usage scenarios or summarize the application that produces the problem.	
Did this happen previously?	
Can you reproduce this problem consistently?	
Any **change in the application environment**: software configuration, EIS/database configuration, application, and so forth?	

Request/Question	**Error/Problem Details or Information**
Under what circumstance does the problem *not* occur?	
Describe the **steps** to reproduce the problem.	
Describe the **problem**.	
Specify the **error** message(s).	

The following table lists error/problem files that might be applicable.

XML schema
XML instances
Other input documents (transformation)
Error screen shots
Error output files
Trace and log files
Log transaction

Contents

CHAPTER 1

Introducing the BEA WebLogic Adapter for Siebel

Topics:

- Features of the BEA WebLogic Adapter for Siebel
- The Siebel Application Model
- Integrating With Siebel
- Siebel EAI Architecture
- Using Application Explorer With the BEA WebLogic Adapter for Siebel
- Deployment Information for the BEA WebLogic Adapter for Siebel

This section explains how the BEA WebLogic Adapter for Siebel facilitates the exchange real-time business data between other applications and Siebel systems. It describes the key features of the adapter as well as the Siebel architecture.

Features of the BEA WebLogic Adapter for Siebel

The BEA WebLogic Adapter for Siebel provides a means to exchange real-time business data between Siebel systems and other application, database, or external business partner systems. The adapter enables external applications for inbound and outbound processing with Siebel.

The adapter uses XML messages to enable non-Siebel applications to communicate and exchange transactions with Siebel using services and events.

- **Services:** Applications use this capability to initiate a Siebel business event.
- **Events:** Applications use this capability to access Siebel data only when a Siebel business event occurs.

The BEA WebLogic Adapter for Siebel:

- Supports synchronous and asynchronous, bidirectional message interactions for Siebel Business Services, Business Components, and Integration Objects.
- Includes the Application Explorer, a GUI tool that uses the Siebel Object Manager to explore Siebel metadata and build XML schemas or Web services.
- Supports Siebel transports—MQSeries, File, and HTTP.

The Siebel Application Model

The Siebel Enterprise application defines a data abstraction layer that removes dependencies on the underlying database. After defining and connecting to a Siebel target within Application Explorer, three primary kinds of Siebel object types are visible:

- Business Objects
- Business Services
- Integration Objects.

These object types represent the Siebel data structure in the Siebel business logic layer and can be configured using Siebel Tools.

Business Objects. A business object implements a business model (logical database diagram), tying together a set of interrelated business components using links. The links provide the one-to-many relationships that govern how the business components interrelate in the context of this business object. Expanding a Business Object in Application Explorer reveals all the business components related to that particular business object. A Business Component is a logical entity that associates columns from one or more tables into a single structure. When instantiated in a Siebel application, a Business Component is comparable to a record set. Its definition in Siebel Tools provides the foundation for controlling how data is inserted, deleted, queried, and updated within the tables it references. You can view these methods when you expand any of the Business Components. When you click on any of these methods you will see the request and response schemas.

Business Services. A Business Service is a reusable module containing a set of methods. It is an object that encapsulates and simplifies the use of some set of functionality. It provides the ability to call its C++ or script methods from customer-defined scripts and object interface logic, through the invoke-method mechanism. A service has properties and methods that can be viewed by expanding a Business Service node in Application Explorer.

Integration Objects. Siebel Integration Objects represent integration metadata for Siebel Business objects, XML, SAP IDOCs, and SAP BAPIs as common structures that the EAI infrastructure can understand. You can use Siebel tools to create the XSD or XDR schemas to be used as IO nodes. (An XDR created in Siebel Tools must be converted to an XML schema in Application Explorer.) In this case, the XML is hierarchical and represents a complex data type.

Integrating With Siebel

You can use the BEA WebLogic Adapter for Siebel to invoke a Siebel business process, such as add/update account, or you can use the adapter as part of an integration effort to connect Siebel and non-Siebel systems. The BEA WebLogic Adapter for Siebel is bidirectional and can detect an event from Siebel by receiving a Siebel XML document emitted by Siebel.

When integrating with Siebel using Siebel XML documents, the adapter application developer can use existing Siebel Integration Objects or create new Siebel Integration Objects to use within a Siebel Workflow. The Workflow processes inbound or outbound Siebel XML and uses various transports such as MQSeries, File, and HTTP to exchange transactions with external systems. The Siebel Workflow is usually created by the Siebel administrator or developer using Siebel Workflow Administration screens.

When integrating with Siebel directly using the Java™ Data Bean or COM Data Interface, the BEA WebLogic Adapter for Siebel does not require a Siebel Integration Object or Siebel Workflow. Instead, it executes Siebel Business Services and Siebel Business Components directly.

The following table lists Siebel objects and describes the transport methods and processes for each object.

Siebel Objects	API or Transport	Process
Business Services	Java Data Bean (Siebel Version 6.3 - 7.8.2) Com Data Interface (Siebel Version 6.0.1 - 6.2)	Service
Business Components	Java Data Bean (Siebel Version 6.3 - 7.8.2) Com Data Interface (Siebel Version 6.0.1 - 6.2)	Service
Integration Objects	File	Event, Service
	HTTP	Event, Service
	MQSeries	Event, Service
	MQ Read	Service

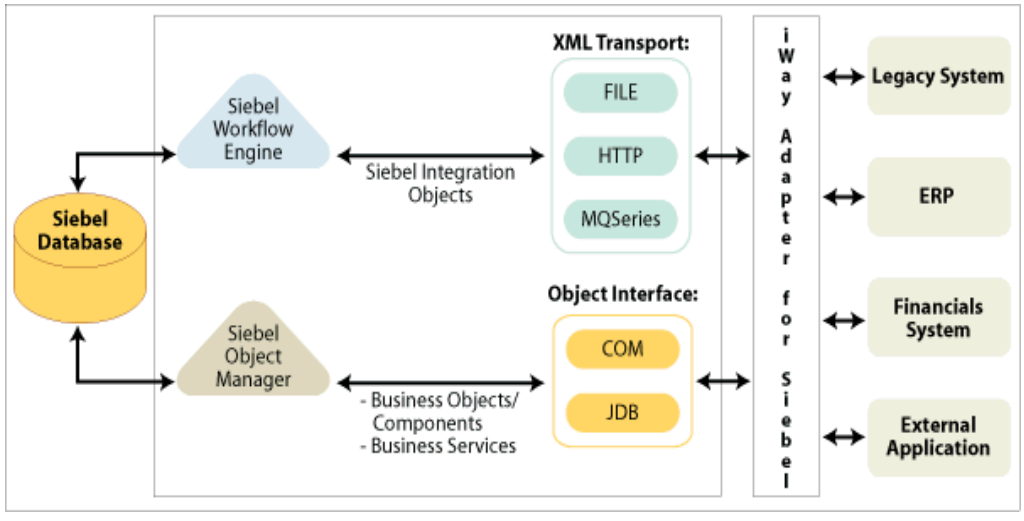
Siebel EAI Architecture

Siebel provides for integration with other applications and systems using its Siebel EAI framework and its Business Integration Manager facility. The BEA WebLogic Adapter for Siebel uses the Siebel EAI framework and leverages various integration access methods to provide the greatest amount of flexibility and functionality while working within the Siebel framework.

The BEA WebLogic Adapter for Siebel supports the following integration access methods:

- Siebel Java Data Bean for services involving Siebel Business Components or Siebel Business Services.
- Siebel COM Data Interface for services involving Siebel Business Components or Siebel Business Services.
- Siebel XML for events and services involving Siebel Integration Objects.

The following illustration shows how the BEA WebLogic Adapter for Siebel helps integrate a Siebel database through either a transport protocol such as File, HTTP, or MQSeries and the Siebel Workflow Engine for Siebel Integration Objects with legacy systems, ERPs, financial systems, and external applications. It also shows how the adapter helps to integrate a Siebel database through an object interface such as COM or JDB and the Siebel Object Manager for Business Components and Business Services with legacy systems, ERPs, financial systems, and external applications.



Using Application Explorer With the BEA WebLogic Adapter for Siebel

Application Explorer uses an explorer metaphor for browsing the Siebel system for Business Services, Business Objects, Business Components, and Integration Objects. The explorer enables you to create XML schemas and Web services for the associated object. External applications that access Siebel through the BEA WebLogic Adapter for Siebel use either XML schemas or Web services to pass data between the external application and the adapter.

Application Explorer can be deployed as a Java Web application running within a servlet container that is accessible through a Web browser. Application Explorer uses interfaces provided by Siebel and in-depth knowledge of the Siebel application systems to access and browse business object metadata. After an object is selected, Application Explorer can generate an XML schema or Web service to define the object for use in conjunction with the BEA WebLogic Adapter for Siebel.

External applications accessing Siebel via the BEA WebLogic Adapter for Siebel use either the XML document or Web service to pass data between the external application and the adapter.

The steps required to create XML schemas for Web services are illustrated in *Chapter 2, Creating XML Schemas and Integration Business Services.*

Key Features of Application Explorer

Key features of Application Explorer include:

- The ability to connect to and explore a variety of application systems.
- Access to application system object metadata.
- A point-and-click process for generating XML schemas and Web services.

Deployment Information for the BEA WebLogic Adapter for Siebel

The BEA WebLogic Adapter for Siebel works in conjunction with one of the following components:

- Integration Business Services Engine (iBSE)
- Enterprise Connector for J2EE™ Connector Architecture (JCA)

When the adapter is hosted in a third-party application server environment, Application Explorer, used to configure Siebel connections, create Web services, and configure event capabilities, can be configured to work in a Web services environment in conjunction with the iBSE. When working in a JCA environment, the connector uses the Common Client Interface (CCI) to provide fast integration services using adapters instead of using Web services.

Encoding Support on UNIX Platforms

When using the adapter on UNIX environments, you must edit the startup script for your server .

When using the adapter in a **third-party application server environment**, you must manually edit the start script for that platform to add a JVM option specifying the file encoding:

```
java $REMDBG -cp $CLASSPATH -DIWAY55=$IWAY55 -Dfile.encoding=ISO8859_1
edaqm -config $SCRIPT $2 $3 $4 $5 $6
```

Deployment Information Roadmap

The following table lists the location of deployment and user information for components of the BEA WebLogic Adapter for Siebel.

Deployed Component	For more information, see
Application Explorer	Chapters 2 and 3 of this guide *BEA WebLogic ERP Adapter Installation and Configuration*
Integration Business Services Engine (iBSE)	*BEA WebLogic ERP Adapter Installation and Configuration*
Enterprise Connector for J2EE Connector Architecture (JCA)	*BEA WebLogic ERP Adapter Installation and Configuration*

Application Explorer

Application Explorer uses an explorer metaphor to browse theSiebel system for metadata. The explorer enables you to create XML schemas and Web services for the associated object. In addition, you can create ports and channels to listen for events inSiebel. External applications that access Siebel through the BEA WebLogic Adapter for Siebel use either XML schemas or Web services to pass data between the external application and the adapter.

Integration Business Services Engine

The Integration Business Services Engine (iBSE) exposes—as Web services—enterprise assets that are accessible from adapters regardless of the programming language or the particular operating system.

iBSE simplifies the creation and execution of Web services when running:

- Custom and legacy applications

- Database queries and stored procedures
- Packaged applications
- Terminal emulation and screen-based systems
- Transactional systems

Web services is a distributed programming architecture that solves Enterprise Application Integration (EAI) hurdles that other programming models cannot. It enables programs to communicate with one another using a text-based, platform- and language-independent message format called XML.

Coupled with a platform and language independent messaging protocol called SOAP (Simple Object Access Protocol), XML enables application development and integration by assembling previously built components from multiple Web services.

Enterprise Connector for J2EE Connector Architecture (JCA)

The Enterprise Connector for J2EE Connector Architecture (JCA) enables developers of JCA-compliant applications to deploy adapters as JCA resources. The connector is supported on J2EE-compliant application servers, such as your BEA WebLogic Server.

The Connector for JCA is distributed as a standard Resource Adapter Archive (RAR) for deployment to the application server. Thus, the connector can be used in systems that are non-compliant, although services such as pooled connections are not available.

CHAPTER 2

Creating XML Schemas and Integration Business Services

Topics:

- Processing Overview
- Starting Servlet Application Explorer
- Managing a Siebel Connection
- Viewing Metadata
- Creating a Schema for a Siebel Business Component or Siebel Business Service
- Creating an XML Schema for a Siebel Integration Object
- Creating Integration Business Services

This section provides the information you require to create schemas for Siebel Business Components, Business Services, and Integration Objects. It describes how to use Servlet Application Explorer.

Although this section describes the Java™ servlet implementation of Application Explorer, other implementations provide the same functionality by means of similar graphical user interfaces.

Processing Overview

The BEA WebLogic Adapter for Siebel enables interaction with Siebel Business Services, Business Components, and Integration Objects.

When using the adapter to integrate with Siebel Business Services and Business Components, the adapter uses the Siebel-supplied Java Data Bean or COM EAI interface. You are not required to create Siebel workflows. Also, because the service is accomplished through a TCP connection, you do not require a transport layer such as MQSeries, File, or HTTP.

A request begins with the sending of an XML request document. In most cases, the response is an XML response document that indicates the execution of the Business Service or Business Component.

When using the adapter to integrate with Siebel Integration Objects, the adapter uses Siebel XML, HTTP, File, WebSphere MQ, and MSMQ transports and Siebel workflows. The workflow is defined within Siebel to either emit or receive Siebel XML through one of the supported Siebel transport services for MQSeries, File, or HTTP.

Encoding Support on UNIX Platforms

When using the adapter in a **third-party application server environment** on UNIX environments, you must edit the startup script for your server :

```
java $REMDBG -cp $CLASSPATH -DIWAY55=$IWAY55 -Dfile.encoding=ISO8859_1
edaqm -config $SCRIPT $2 $3 $4 $5 $6
```

Starting Servlet Application Explorer

Before you can use Servlet Application Explorer, you must start your application server.

Procedure: How to Start BEA WebLogic Server on Windows

To start BEA WebLogic Server on Windows:

1. Click the *Start* menu.
2. Select *Programs*, *BEA Products,* and then click *WebLogic Server.*

Procedure: How to Start BEA WebLogic Server on UNIX

To start BEA WebLogic Server on UNIX or from a command line, enter the following at the prompt:

```
BEA_HOME/user_projects/domains/DOMAIN_NAME/startWebLogic.cmd
```

where:

BEA_HOME

Is the directory where BEA WebLogic is installed.

DOMAIN_NAME

Is the domain you are using for iWay.

Procedure: How to Open Servlet Application Explorer

To open Application Explorer:

1. Ensure that your application server is running.
2. Enter the following URL in your browser:

   ```
   http://hostname:port/iwae/index.html
   ```

   where:

   hostname

   Is the name of the machine where your application server is running.

   port

   Is the port for the domain you are using for BEA. The port for the default domain is 7001.

You are ready to create new targets to the Siebel enterprise information system.

Managing a Siebel Connection

To browse the Siebel Business Services, Business Components, and Integration Objects, you must define a target to Siebel. After you define the target, the parameters are automatically saved. However, you must supply the password to Siebel every time you connect to the target. For more information on connecting to a target, see *How to Connect to a Defined Target* on page 2-8.

Note: The connection parameters can be obtained from the eapps.cfg file, which is located in the following directory:

`drive:\SiebelRoot\SWEApp\BIN`

where:

SiebelRoot

Is the Siebel installation directory.

You create a new target from the Service Adapters tab of Application Explorer. For information on creating a target, see *How to Define a Target to Siebel* on page 2-5.

Although you can maintain multiple open connections to different application systems, it is good practice to close connections when not in use. For information on disconnecting from a target, see *How to Disconnect From Siebel* on page 2-9.

After you create a target for Siebel using Application Explorer, you can edit any information that you provided during the creation process. For information on editing a target, see *How to Edit a Target* on page 2-9.

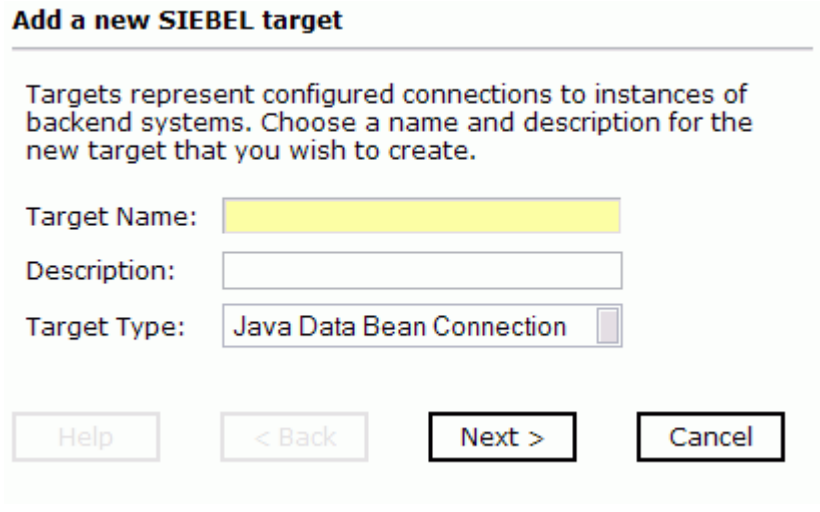
You can delete a target, rather than just disconnecting from a target and closing it. When you delete the target, the node disappears from the list of Siebel targets in the left pane of the explorer. For information on deleting a target, see *How to Delete a Target to Siebel* on page 2-9.

Procedure: How to Define a Target to Siebel

To define a target to Siebel:

1. In the left pane of Application Explorer under Service Adapters, select the *Siebel* node.
2. In the right pane, move the pointer over *Operations* and select *Define a New Target*.

   The following image shows the pane that appears on the right where you can define a new target.



   a. In the Target Name field, type a name for the new target, for example, SiebelConnection.

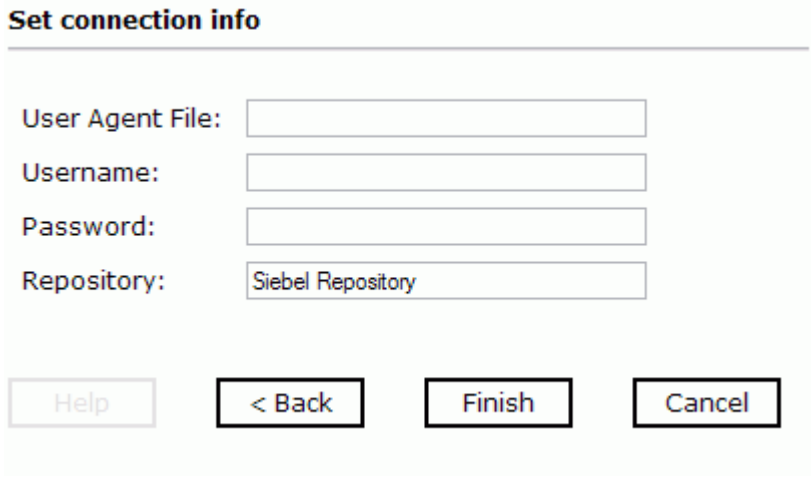
   b. In the Description field, type a description (optional).

   c. For Siebel 6.0, select *Siebel 6.2* or for Siebel 6.3, select *Java Bean Data Connection* from the Target Type drop-down list.

3. To continue, click *Next*. To go back or to cancel, click the appropriate button.

   When you select 6.2 or lower (COM), the Set connection info dialog box appears where you type connection and logon information.

   The following image shows the Set connection info dialog box.



   a. In the User Agent File field, type *UAGENT.CFG* or type the name of your configuration file, if the name is different.

   UAGENT.CFG is the default value.

   This file contains the configuration parameters for connecting to Siebel.

   b. In the Username field, type the Siebel user name.

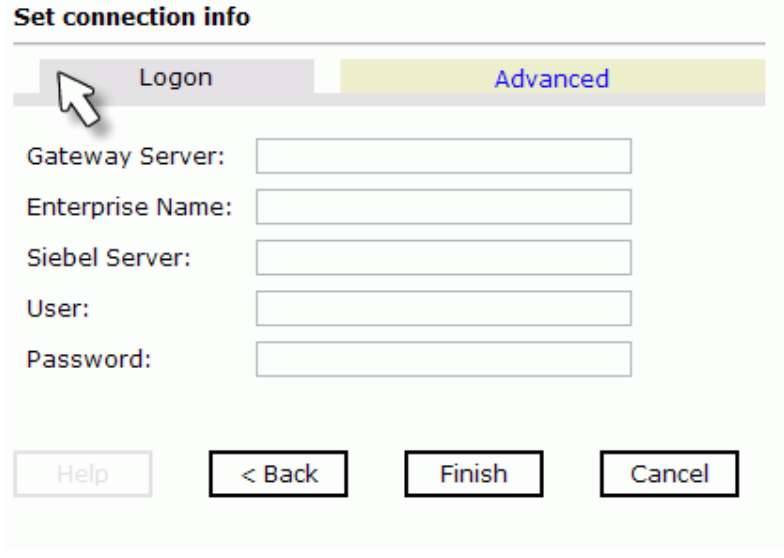
   c. In the Password field, type the password associated with the user name.

   d. In the Repository field, type the name of the repository where Application Explorer looks for metadata describing Business Services, Business Objects, and Integration Objects.

   If no repository is specified, a full list of objects from all available repositories is returned. If a specified repository is not found, an empty list of objects is returned.

When you select 6.3 (JDB), the Set connection info dialog box appears with a Logon tab and an Advanced tab.

The following image shows the active Logon tab where you type configuration information.



a. In the Gateway Server field, type the name of the server. To specify a Gateway Server that uses a port other than the default (usually, 2320), add a colon and the port number, for example, gateway_name: port_number.

b. In the Enterprise Name field, type the appropriate name.

c. In the Siebel Server field, type the name of your Siebel server. You do not have to supply a value in this field when connecting to a Siebel 7.7 system.

d. In the User field, type the user name.

e. In the Password field, type the password associated with the user name.

4. Click the *Advanced* tab and verify the following: Language, Object Manager, and Repository Name.

   Object Manager is the name of an active Siebel Object Manager. The following table shows examples of various Object Managers.

Siebel Object Manager	Associated Application
SCCObjMgr	Siebel Call Center
EAIObjMgr	Siebel 7.0.3
EAIObjMgr_enu	Siebel 7.5 and Siebel 7.7

   Note: Siebel 7.5 requires that you add the language extension (for example, _enu) to the end of the Object Manager name. Check with your Siebel Administrator for the specific names that apply to your system.

   If no repository is specified, a full list of objects from all available repositories is returned. If a specified repository is not found, an empty list of objects is returned.

   The configuration parameters supplied are used by Siebel client applications to connect to the Siebel system. For more information about these parameters, see your Siebel documentation or ask your Siebel system administrator.

5. Click *OK*.

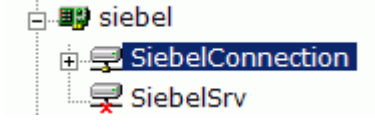
   In the left pane, the target you create appears under the Siebel node.

Procedure: How to Connect to a Defined Target

To connect to a previously defined Siebel target:

1. Click the target name under the Siebel node.
2. In the right pane, move the cursor over *Operations* and select *Connect*.
3. Type the password and click *OK*.

   The following image shows the siebel node with the SiebelConnection target icon selected beneath it. The SiebelSrv target is disconnected from the Siebel system.



You can browse available Business Objects, Business Services, and Integration Objects in the Siebel system.

Procedure: How to Disconnect From Siebel

To disconnect from Siebel:

1. In the left pane, click the target to which you are connected.
2. In the right pane, move the pointer over *Operations* and select *Disconnect*.

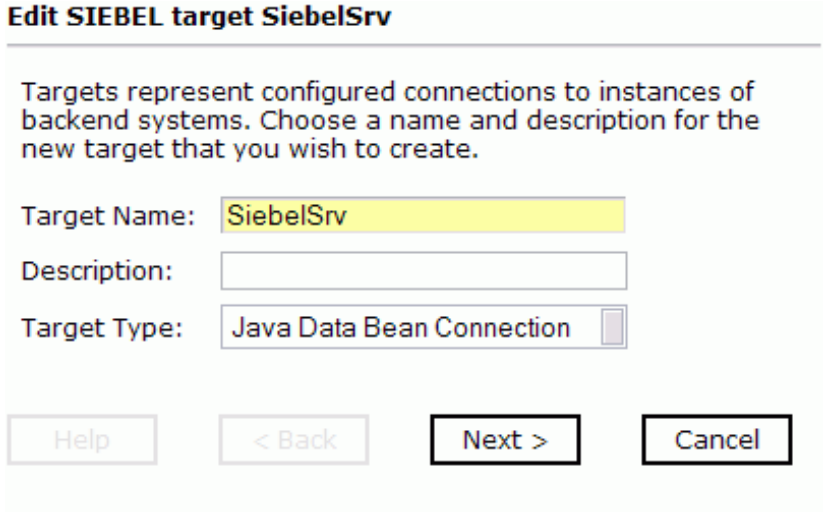
Disconnecting from the application system drops the target, but the node remains. The SiebelConnection node in the left pane changes to reflect that the target was closed.

Procedure: How to Edit a Target

To edit a target:

1. In the left pane, click the target node.
2. In the right pane, move the pointer over *Operations* and select *Edit*.

   The following image shows the Edit pane that opens on the right. There are three fields where you can edit information: Target Name, Description, and Target Type. You can click a button to continue, go back, cancel out of the Edit pane, or view help.



3. Modify the target information.
4. Click *Next*.

Procedure: How to Delete a Target to Siebel

To delete a target to Siebel:

1. In the left pane, click the target, for example, SiebelConnection.

2. In the right pane, move the cursor over *Operations* and select *Delete*.

   A confirmation dialog box opens.

3. To delete the target you selected, click *OK*.

   The SiebelConnection node disappears from the left pane.

Viewing Metadata

Viewing metadata is useful for understanding the structure of Siebel data. You can review the parameters, data types, and other attributes in the right pane.

Procedure: How to View Metadata

To view metadata:

1. If you have not started the explorer, start Application Explorer and connect to your Siebel system.
2. In the left pane, expand the Business Object or Business Service containing the component for which you want to generate schema.
3. Expand the Business Object or Business Service node.

   A Business Object contains Business Components. For each Business Component, there are insert, update, delete, and query capabilities. Integration Business Services can be created against these functions.

4. Expand the Business Object or the Business Service node in which you are interested to view the components under it.

   For a Business Object, select the node in which you are interested.

   The following image shows the Account Business Object expanded with the Account component selected.

   For a Siebel Business Service, select the object in which you are interested.

   The following image shows the Simple Add Account Business Service expanded with the addAccount component selected.

For an Integration Object, select the Integration Object in which you are interested.

The following image shows the Sample Account Integration Object selected.

5. In the right pane, click the ellipsis (…) in the Table row of the properties table.

   A metadata table appears in the right pane and displays the details of the table you selected. The following image is a sample Details for collection property Table. The table consists of a heading row with column labels that identify each of the six columns: Name, Type, MultiValued, ReadOnly, and Active. Each row represents a different property.

Details for collection property Table

Name	Type	Required	MultiValued	ReadOnly	Active
Account Competitors	string	false	false	false	true
Account Condition	string	false	false	false	true
Account Markets	string	false	false	false	true
Account Organization Integration Id	string	false	false	false	true
Account Products	string	false	false	false	true
Account Role	string	false	false	false	true
Account Status	string	false	false	false	true
Account Trend	string	false	false	false	true
Address Active Status	string	false	true	false	true
Address Id	string	false	true	false	true
Address Integration Id	string	false	true	false	true

Creating a Schema for a Siebel Business Component or Siebel Business Service

You can create service schemas for Business Services, Business Components, and Integration Objects using Application Explorer. For information on creating schemas for Integration Objects, see *Creating an XML Schema for a Siebel Integration Object* on page 2-19.

The following topic describes how to create schemas for the adapter when you deploy the BEA WebLogic Adapter for Siebel for BEA WebLogic for use in a JCA (Enterprise Connector for J2EE Connector Architecture) environment or an Integration Business Services environment.

If you plan to deploy the BEA WebLogic Adapter for Siebel in a Web services environment, see also *Creating Integration Business Services* on page 2-29.

Creating an XML Schema for a Siebel Business Component or Siebel Business Service

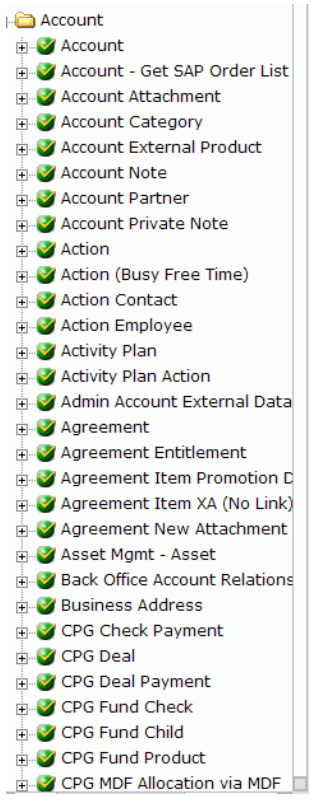
You create schemas for Siebel Business Service methods (for example, the Add method) and Business Components using Application Explorer. After you create a schema, you can use it to generate service request and response schemas for the Business Service or Business Component.

Important: The BEA WebLogic Adapter for Siebel does not support Siebel Business Services that have Business Service method arguments of data type "Hierarchy."

For information on creating schemas for Integration Objects, see *Creating an XML Schema for a Siebel Integration Object* on page 2-19.

Siebel Business Objects contain one or more Siebel Business Components. You can view Business Components by clicking the associated Business Object.

The following image shows the Account Business Object node expanded to display all Business Components beneath it.



Procedure: How to Create an XML Schema for a Siebel Business Component or Siebel Business Service

To generate service request and response schemas for a Business Component:

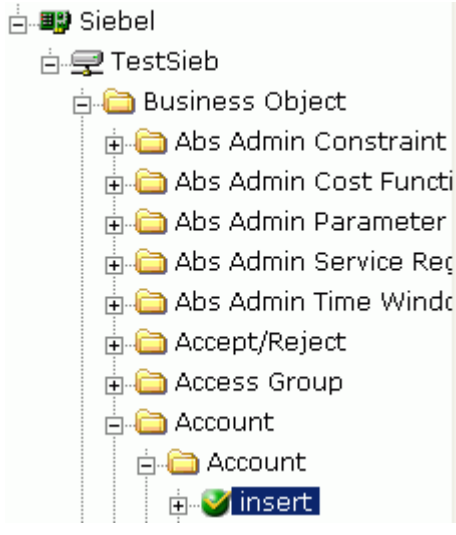
1. If you have not started the explorer, start Application Explorer and connect to your Siebel system through a target.
2. In the left pane, expand the Business Object or the Business Service node.

   You can also use the Search feature to find a particular Business Component or Business Service. For more information, see *How to Search for a Specific Siebel Object* on page 2-16.

3. Expand the Business Component or Business Service to view the objects under it.

   For a Business Component, expand the Business Object node, then expand the Business Component you want, then expand the node you want, and select the method for which you want to create a schema.

   The following image shows the Account Business Component expanded to reveal the Account node. This node is expanded with the insert method selected.



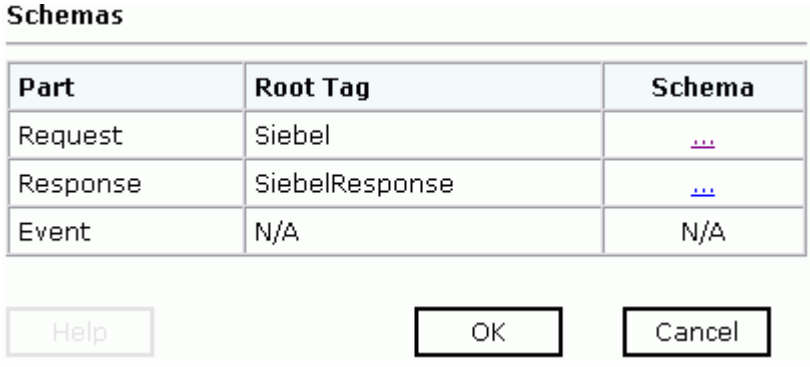
   For a Siebel Business Service, expand the Business Service node containing the object for which you want to create schema.

   The following image shows the Simple Add Account Business Service expanded with the addAccount object selected.

4. In the right pane, move the cursor over *Operations* and select *Generate Schema*.

   Application Explorer accesses the Siebel repository and builds schemas.

   The following image shows the Schemas table that appears on the right and has three columns, labeled Part, Root Tag, and Schema. The Schema column provides the locations of the schemas. There are three rows: Request, Response, and Event.



5. To view a schema, click the ellipsis (…) in the row corresponding to the schema you want to view.

The following image shows the XML schema that appears in the right pane.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--  Generated by the iBSE 2004-04-09T18:18:16Z
  -->
- <xsd:schema
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   targetNamespace="urn:iwaysoftware:adapter:siebel:bo:
   xmlns:z="urn:iwaysoftware:adapter:siebel:bo:oct2003"
   elementFormDefault="qualified">
  - <xsd:element name="Siebel">
    - <xsd:complexType>
      - <xsd:sequence>
          <xsd:element name="insert"
            type="z:record" />
        </xsd:sequence>
        <xsd:attribute name="location"
          type="xsd:string" use="optional"
          default="S/BO/Account/Account/insert" />
      </xsd:complexType>
    </xsd:element>
  - <xsd:complexType name="record">
    - <xsd:sequence>
        <xsd:element
          name="Account_spcCompetitors"
          type="xsd:string" minOccurs="0" />
        <xsd:element
```

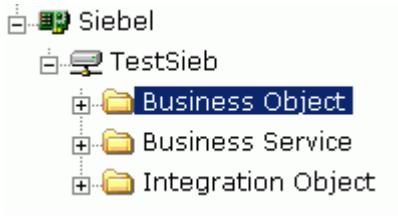
For more information on where the schemas are stored, see *Schema Location on page 2-27*.

Procedure: How to Search for a Specific Siebel Object

You can use the search function in Application Explorer to locate a Siebel object or node quickly.

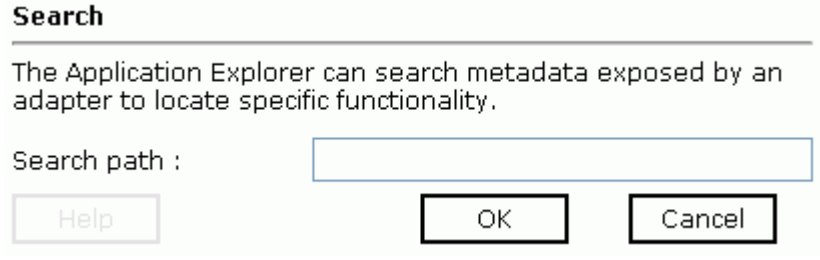
1. If you have not started the explorer, start Application Explorer and connect to your Siebel system through a target.
2. Expand the target and select *Business Object*, *Business Service*, or *Integtration Object*.

   The following image shows Business Object selected in the left pane.



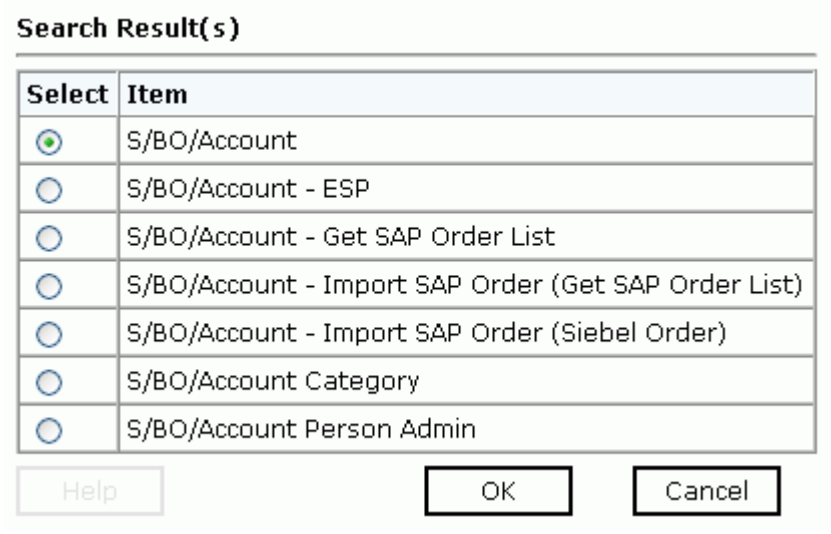
3. In the right pane, move the cursor over *Operations* and select *Search*.

   The following image shows the Search feature that appears in the right pane. It has a search path input area.



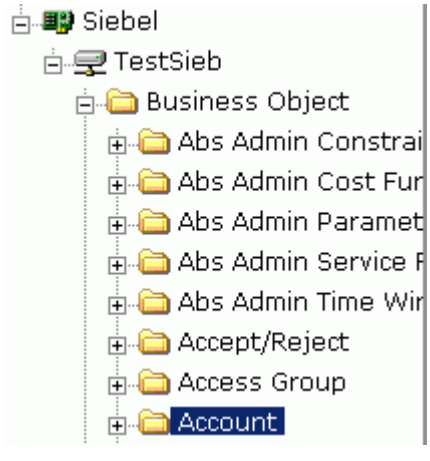
4. Enter the name of the node or object on which you want to search in the Search path text entry box, for example, Account.
5. Click *OK*.

   The following image show the search results that appear when a search for Account is conducted against the Siebel Business Objects.



6. Select the radio button next to the item in which you are interested, for example, S/BO/Account.
7. Click *OK*.

   Application Explorer locates the node you select, for example, Account.



Creating an XML Schema for a Siebel Integration Object

The BEA WebLogic Adapter for Siebel supports access to Siebel Integration Objects by using Siebel XML. Using Siebel Integration Objects through supported transports requires Siebel workflows. For more information, see Appendix B, *Siebel Workflows*.

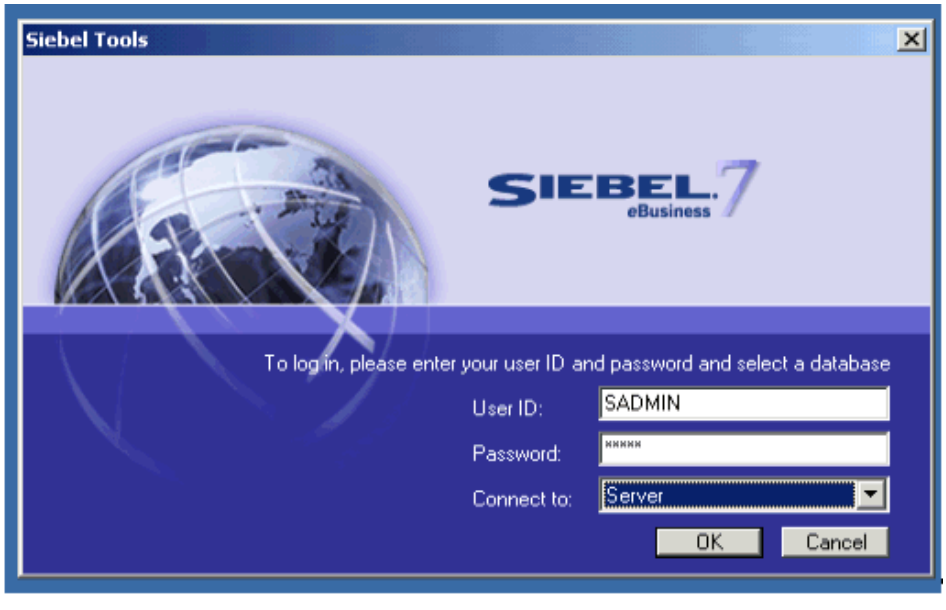
Creating a Siebel XDR or XSD Schema

For releases prior to Siebel 6.3, the Siebel Tools Schema Wizard creates only DTD schemas. You must transform these schemas manually, or by using other tools, into XDR files before Application Explorer can use them as input to create XML schemas. In addition, you must include the SiebelMessage tag reference in your XDR file. For an example, see the Sample Account Integration Object in Appendix A, *Usage Considerations and Sample Files*.

Note: Starting with version 7.5, you can generate XSD schemas using Siebel Tools. You can use these XSD files in Application Explorer.

Procedure: How to Create a Siebel XDR or XSD Schema for a Siebel Integration Object

The following image shows the Siebel Tools screen where you log on.

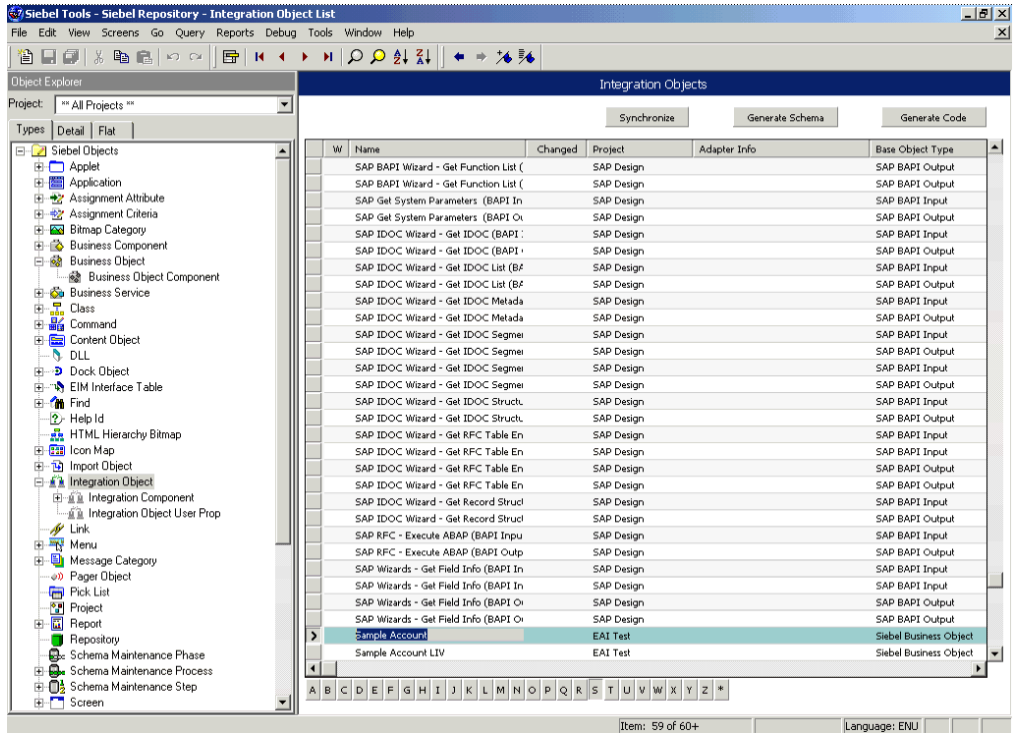


To generate a Siebel XDR or XSD schema:

1. Log on to Siebel Tools.
   a. Type a user ID and password.
   b. From the Connect to: drop-down list, select a database.

2. Click *OK*.

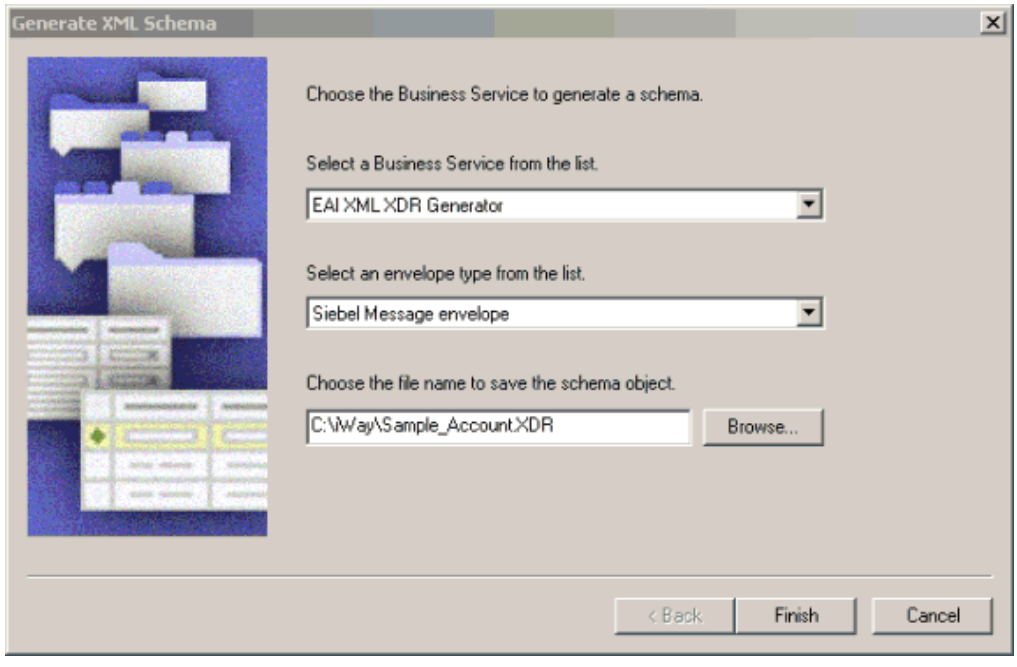
The following image shows the Siebel Tools window that opens. In the upper left is the Project drop-down list. In the lower left is the Object Explorer navigation pane with three tabs: Types, Detail, and Flat, with the Types tab selected to show a list of Siebel objects. The right pane has three buttons: Synchronize, Generate Schema, and Generate Code, followed by a table of Integration objects with their associated project and base object type. You can navigate by selecting a specific letter of the alphabet from a button beneath the table.



3. To create an XML schema, select an integration object, for example, Sample Account.

4. Click the *Generate Schema* button.

The following image shows the Generate XML Schema wizard window that opens where you select a Business Service to generate a schema.



a. From the Business Service drop-down list, select *EAI XML XDR Generator* or *EAI XML XSD Generator*.

b. From the envelope type drop-down list, select *Siebel Message envelope*.

c. In the third field, click *Browse* or type to specify a file name for the XDR or XSD schema and a directory where it can be accessed by Application Explorer, for example, C:\iWay\Sample_Account.XDR.

Note: The XDR or XSD file must be on the same computer as Application Explorer or be available through a mapped connection to another drive or machine

5. Click *Finish*.

Now you can use Application Explorer to generate XML schemas for the Siebel Integration Object.

For more information, see *How to Create a Schema For a Siebel Integration Object* on page 2-22.

Creating a Schema for a Siebel Integration Object Using Application Explorer

Application Explorer can generate schemas for Integration Objects from Siebel XDR files or it can use Siebel-generated XSD files. In Siebel version 7.5, you can create XSD schemas for Integration Objects using Siebel Tools.

If you created an XDR file using the Siebel Tools Schema Wizard, after you create the Siebel XDR schema for a selected Siebel Integration Object, you can create an XML schema using Application Explorer.

You must supply Application Explorer with the location of the previously created Siebel XDR schema for the particular integration object selected.

Note: The XDR or XSD file must be on the same computer as Application Explorer or be available through a mapped connection to another drive or machine.

Procedure: How to Create a Schema For a Siebel Integration Object

To create an XML schema from a Siebel XDR schema:

1. In Application Explorer, expand the *Integration Objects* node to browse the Integration Objects in the Siebel system.

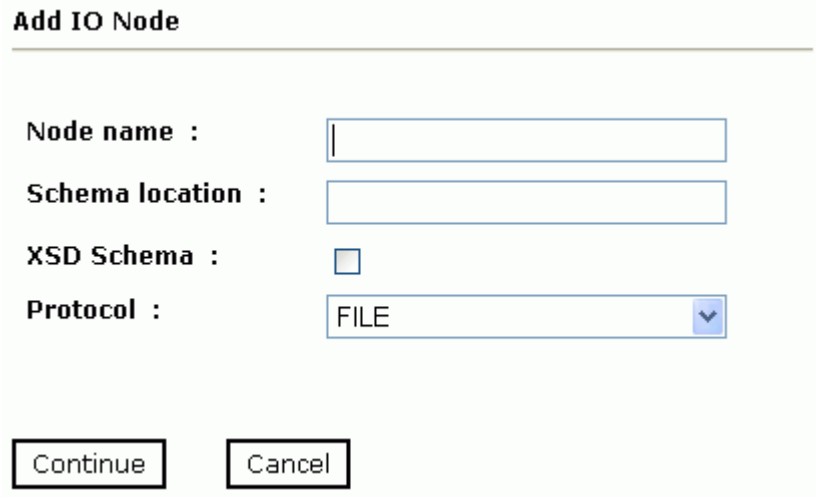
   The following image shows the Integration Object nodes in the left navigation pane and a table of property details for a selected node in the right pane. Each row of the table lists a property of the object and its value.

2. Scroll down and select an integration object, for example, Sample Account.

   You can also use the Search feature to find a particular Integration Object. For more information, see *How to Search for a Specific Siebel Object* on page 2-16.

3. To generate a schema, move the pointer over *Operations* and choose *Add IO Node*.

   The following image shows the Add IO Node pane that appears on the right.



   a. In the Node name field, type a name for the node to create under Sample Account.

   b. In the Schema location field, type the location of the XDR or XML schema that was created by Siebel Tools. Include the name of the file in the location, for example:

      ```
      C:\siebelrepo\SAMPLE_ACCOUNT.XDR
      ```

      Note: For Siebel version 7.5, you can generate XML schemas using Siebel Tools.

   c. If the XSD schema has already been generated, select *XSD Schema*. If you are using Siebel-generated XDR schemas, **do not** select the XSD schema option.

      Application Explorer uses XDR schemas as input to generate XSD schemas.

   d. From the drop-down list, choose a protocol used by the Siebel workflow for the Integration Object.

4. Click *Continue*.

   For **FILE**, provide the location used by the Siebel workflow.

   For **HTTP**, Application Explorer builds the URL that is the key to activating the SWE. The protocol definition has multiple sections:

   ```
   http://my_web_server/eai_enu/start.swe?SWEExtSource=<SourceName>&SWEExtCmd=<Execute>&UserName=<UserName>&Password=<Password>
   ```

   The following table lists and defines the parameters you supply for **HTTP**.

Parameter	**Definition**
SWE URL	Base SWE URL. For example `http://web_server/eai_enu/start.swe` where: *web_server* Is the name of the Web server hosting Siebel SWE.
SWE External Source	Section within the eai.cfg file to execute, which is the [HTTP Services] section.
SWE External Command	Use Execute.
User Name	User ID logon to execute.
Password	Logon password to execute.

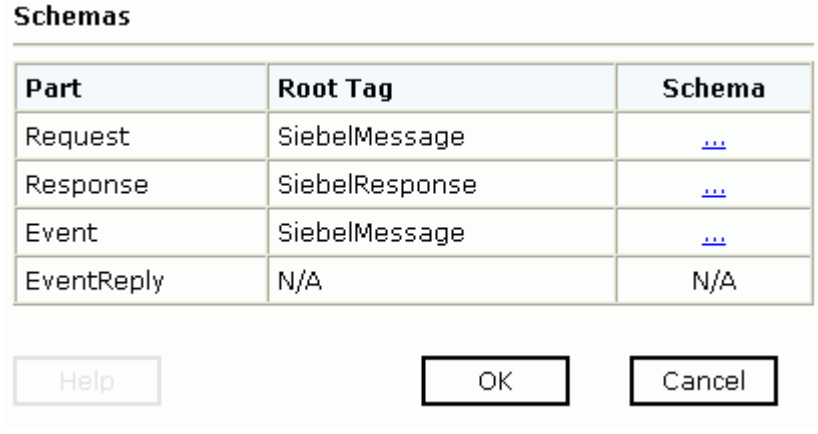
   The following table lists and defines the parameters you supply for **MQSeries**.

Parameter	**Definition**
Queue Manager Name	Name of the queue manager to which the server must connect.
MQ server host for MQClient operation	Host on which the MQ Server resides (MQ Client only).
MQ server port for MQClient operation	The number to connect to an MQ Server queue manager (MQ client only).
MQ server channel for MQClient operation	Case-sensitive name of the channel that connects with the remote MQ Server queue manager (MQ client only). The default channel name for MQSeries is `SYSTEM.DEF.SVRCONN`

Parameter	Definition
Document type XML	Keep the default selection.
respqueue	Name of the queue where messages are placed.

5. Select the node just created and then select *Generate Schema* from the Operations menu in the right pane.

   The following image shows the Schemas table that appears on the right and has three columns labeled Part, Root Tag, and Schema. The Schema column provides the locations of the schemas. There are four rows: Request, Response, Event, and EventReply.



6. To view the XML for a schema, click the ellipsis (…) in the event row.

The following image shows the XML schema that appears in the right pane.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--  Generated by the iBSE 2004-11-15T21:30:31Z
  -->
- <Schema xmlns:d="urn:schemas-microsoft-
    com:datatypes" xmlns="urn:schemas-microsoft-
    com:xml-data" name="SiebelMessage">
  - <ElementType content="eltOnly" model="closed"
      name="SiebelMessage">
      <AttributeType name="MessageId" />
      <AttributeType default="Integration Object"
        name="MessageType" />
      <AttributeType default="Sample Account"
        name="IntObjectName" />
      <AttributeType name="IntObjectFormat" />
      <attribute type="MessageId" />
      <attribute required="yes"
        type="MessageType" />
      <attribute required="yes"
        type="IntObjectName" />
      <attribute type="IntObjectFormat" />
      <element type="ListOfSampleAccount"
        minOccurs="0" maxOccurs="1" />
    </ElementType>
    <ElementType d:type="string" d:maxLength="50"
      content="textOnly" model="closed"
```

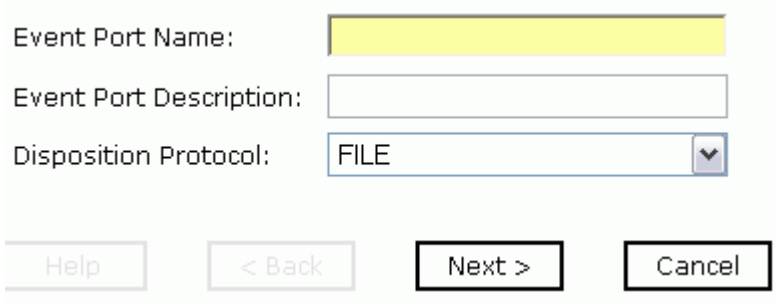
7. Click the browser *Back* button to return to the Schemas table.

A directory structure is created to store the schemas. For more information on where the schemas are stored, see *Schema Location on page 2-27*.

You are now ready to configure ports and channels or create Integration Business Services for the Siebel Integration Object node you just created.

8. To create an event port, click the IO node name you just created and select *Create Event Port* from the Operations menu in the right pane.

   The following image shows the Create Event Port pane that opens on the right. This pane includes two fields you complete to define the port: Event Port Name and Event Port Disposition. It includes the Disposition Protocol drop-down list from which you select the protocol type, a Help button, and three action buttons.



For more information on creating event ports, see *Chapter 3, Listening for Siebel Events*.

Schema Location

Application Explorer stores the schemas it creates in subdirectories under the iWay home directory of the machine where it is installed. The exact location of the schemas differs depending on whether you deploy Application Explorer with an iBSE or a JCA configuration.

When using the adapter with an **iBSE** configuration, the schemas are stored under a \schemas subdirectory of the iWay home directory, for example,

```
C:\Program Files\iway55\bea\ibse\wsdl\schemas\service\siebel\SiebSrv
```

where:

SiebSrv

Is the name of the connection to the Siebel system as defined in Application Explorer. Under this directory, Application Explorer creates subdirectories containing schemas.

When using the adapter with a **JCA** configuration, the schemas are stored under a \schemas subdirectory of the iWay home directory, for example,

```
C:\Program Files\iway55\config\base\schemas\Siebel\SiebServer
```

where:

SiebServer

Is the name of the connection to the Siebel system as defined in Application Explorer. Application Explorer stores the schemas in this directory.

Returning Fields in a Specified Order

When you create a request document from an XML schema to query the Siebel system, you can limit the expected response to specific fields that are specified in the query.

The response will contain the fields in the order in which they were specified. If you do not specify a set of fields, the response document contains the entire set.

For example, the following query will return all fields:

```
<m:Siebel location="S/BO/Account/Account/queryWithView" view="AllView">
   <m:select>
     <m:Name>Yelena*</m:Name>
   </m:select>
</m:Siebel>
```

The following query will return a response that only contains the fields Name, Location and Account Status fields:

```
<m:Siebel location="S/BO/Account/Account/queryWithView" view="AllView">
   <m:select>
      <m:Name>Yelena*</m:Name>
   </m:select>
   <m:field>Name</m:field>
   <m:field>Location</m:field>
   <m:field>Account Status</m:field>
</m:Siebel>
```

Using QueryWithView

For Business Components, the BEA WebLogic Adapter for Siebel enables Insert, Update, Delete, and Query. It also enables a method called QueryWithView. The View modes are a visibility feature provided by Siebel.

By using QueryWithView, you can specify a Siebel View mode as a parameter. The API parameters allow different presentations of data depending on the Siebel environment that you configured.

You can use Query except when you want to enable a user to retrieve records based on different view modes. In this case, use QueryWithView. For more information on QueryWithView mode or Siebel "Visibility" concepts, see your Siebel Administrator.

The following levels are available:

- Sales Rep View
- Manager View
- Personal View
- All View
- Organization View
- Group View
- Catalog View

- SubOrganization View

Creating Integration Business Services

You can generate Integration Business Services (also known as Web services) for Siebel objects you wish to use with your adapter.

Ensure you properly configure the servlet iBSE. For more information on installing and deploying iWay components, see the *BEA WebLogic ERP Adapter Installation and Configuration* manual.

Before you create a Web service for an Integration Object, you must first create a Siebel XDR schema that Application Explorer can use to create an XSD schema, unless you are able to generate an XSD schema using Siebel Tools, which you can do with later versions of Siebel. For more information on creating schemas for Integration Objects, see *Creating an XML Schema for a Siebel Integration Object* on page 2-19.

Note: In a J2EE Connector Architecture (JCA) implementation of adapters, Web services are not available. When the adapters are deployed to use the connector for JCA, the Common Client Interface provides integration services using the adapters. For more information, see the *BEA WebLogic ERP Adapter Installation and Configuration* manual.

Important: The BEA WebLogic Adapter for Siebel does not support Siebel Business Services that have Business Service method arguments of data type "Hierarchy."

Procedure: How to Generate a Web Service

To generate a Web service:

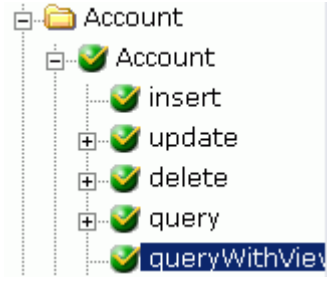
1. If you have not already connected, connect to your Siebel system.
2. Expand the Siebel node.

   For a **Siebel Business Service** or **Integration Object**, expand the node you are interested in and select the node for which you want to create a Web service.

   Note: For an Integration Object, you must first create an IO node before you create a Web service. For more information, see *Creating an XML Schema for a Siebel Integration Object* on page 2-19.

   For a **Business Component**, expand the Business Component for which you want to create a Web service and select a node.

The following image shows the Account Business Component expanded with the queryWithView method selected.



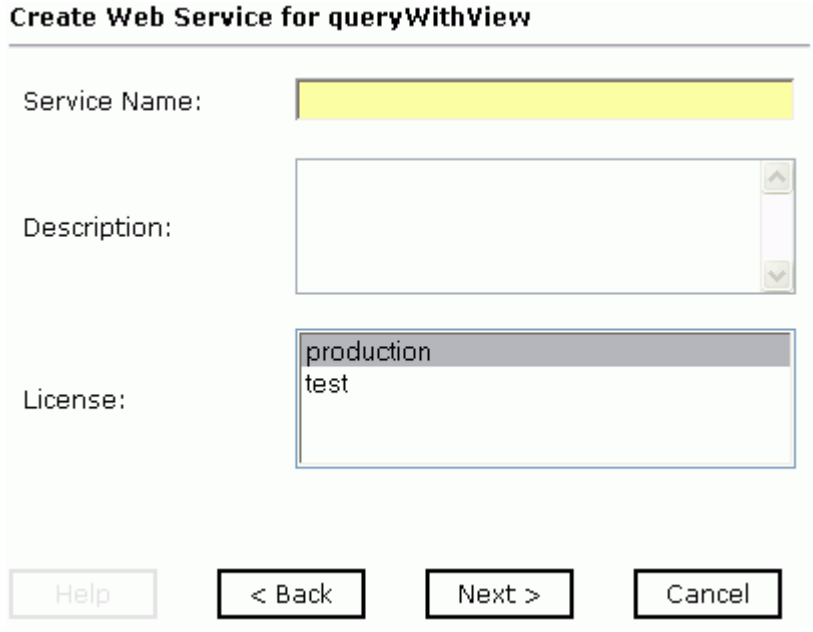
Expand the object and select a method for creating the Web service, for example, QueryWithView under Account.

3. In the right pane, move the pointer over *Operations* and select *Create Integration Business Services*.

   If this is not the first Web service you have created, you can choose whether to create a new service or use an existing service.

   To use a **previously created service**, select the option to use an existing service and when a drop-down list appears, select the Web service to which you want to add the new service.

If this is the first Web service you are creating or if you select to **create a new service**, the Create Web Service pane appears as shown in the following image.



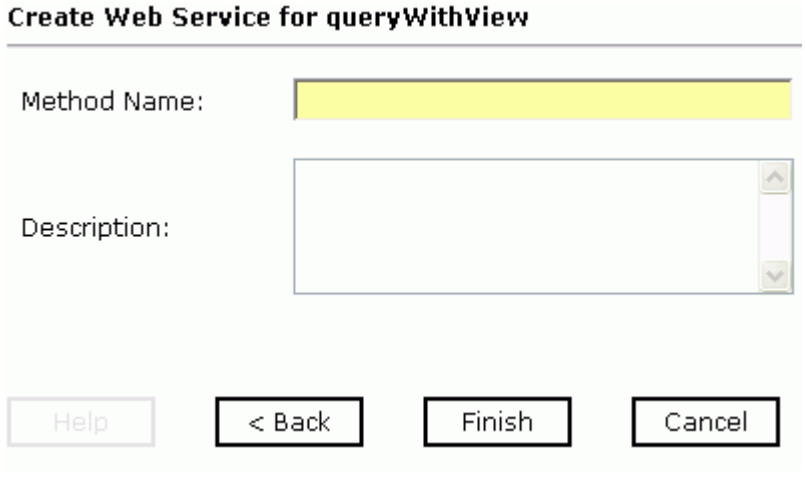
a. In the Service Name field, type a name to identify the Web service (under the Service node in the left pane of the Integration Business Services tab).

b. In the Description field, type a brief description of the Web service.

c. In the License field, select the license(s) with which you want to associate this business service. To select more than one, hold down the *Ctrl* key and click the licenses.

4. Click *Next*.

   The following image shows the Create Web Service pane that reappears and prompts you for information about the method of the service.



   a. In the Method Name field, type a name to specify the name of the SQL statement or stored procedure to add to the business service.

   b. In the Description field, type a brief description of the method.

5. Click *Finish*.

   Application Explorer switches the view to the Integration Business Services tab, and the new business service appears in the left pane.

Testing a Web Service for a Business Component or Integration Object

After you create a Web service for the Siebel Business Component, test it to ensure it functions properly. Application Explorer includes a test tool for testing a Web service.

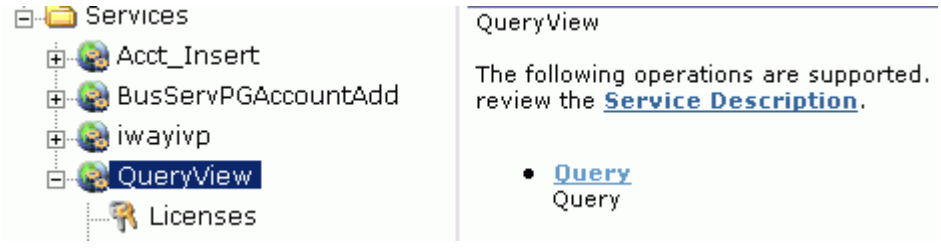
Procedure: How to Test a Web Service for a Business Component

To test a Web service for a Business Component:

1. If you are not on the Integration Business Services tab of Application Explorer, click the tab to access business services.
2. If it is not expanded, expand the *Integration Business Services* node.
3. Expand the *Services* node.

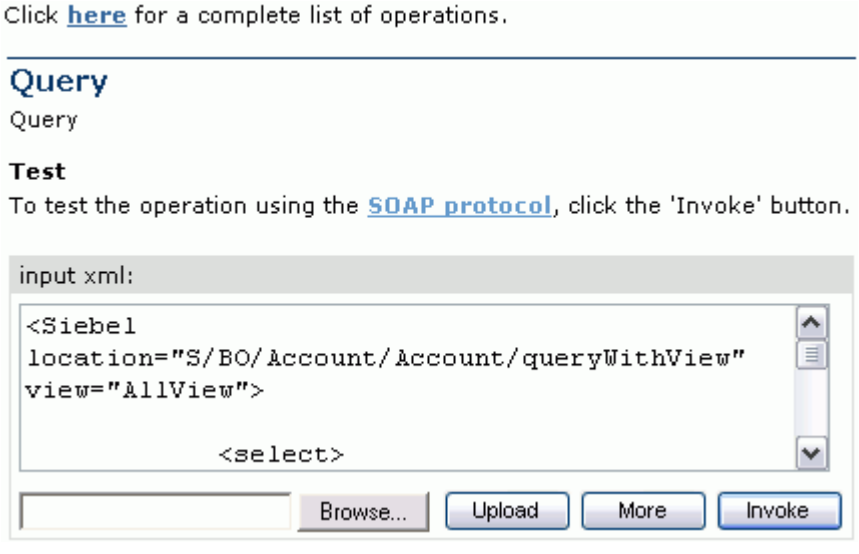
4. Select the name of the business service you want to test.

   The following image shows a window with a list of services in the left pane and information about the selected service in the right pane.



5. In the right pane, click the named business service hyperlink, for example, Query.

   The following image shows the test option that appears in the right pane. This pane provides a text field in which to paste the XML input. Beneath it is a Browse field where you can upload a file and three action buttons.



6. Provide the appropriate XML input.

7. Click *Invoke*.

   The following image shows the XML test results that appear in the right pane.

Testing a Web Service for a Business Service

After you create a Web service for the Siebel Business Service, test it to ensure it functions properly. Application Explorer includes a test tool for testing a Web service.

Procedure: How to Test a Web Service for a Business Service

To test a Web service for a Business Service:

1. If you are not on the Integration Business Services tab of Application Explorer, click the tab to access business services.
2. If it is not expanded, expand the *Integration Business Services* node.
3. Expand the *Services* node.

4. Select the name of the business service you want to test.

   The business service name appears as a link in the right pane.

5. In the right pane, click the named business services link.

   The following image shows the test option that appears in the right pane. The name of the business service appears in the upper pane. The pane has two fields for adding values for parameters.

6. Provide the appropriate input.
7. Click *Invoke*.

   Application Explorer displays the results in the right pane.

Identity Propagation

If you test or execute a Web service using a third party XML editor, for example XMLSPY, the Username and Password values that you specify in the SOAP header must be valid and are used to connect to Siebel. The user name and password values that you provided for Siebel during target creation using Application Explorer are overwritten for this Web service request. The following is a sample SOAP header that is included in the WSDL file for a Web service:

```
<SOAP-ENV:Header>
    <m:ibsinfo xmlns:m="urn:schemas-iwaysoftware-com:iwse">
        <m:service>String</m:service>
        <m:method>String</m:method>
        <m:license>String</m:license>
        <m:disposition>String</m:disposition>
        <m:Username>String</m:Username>
        <m:Password>String</m:Password>
        <m:language>String</m:language>
    </m:ibsinfo>
</SOAP-ENV:Header>
```

Note: You can remove the following tags from the SOAP header, because they are not required:

```
<m:disposition>String</m:disposition>
```

```
<m:language>String</m:language>
```

CHAPTER 3

Listening for Siebel Events

Topics:

- Understanding Event Functionality
- Creating an Event Port
- Creating a Channel
- Deploying Components in a Clustered BEA WebLogic Environment

Servlet Application Explorer deployed to BEA WebLogic Server enables you to listen for events in a Siebel system.

Although this section describes the Java™ servlet implementation of Application Explorer, other implementations provide the same functionality by means of similar graphical user interfaces.

Understanding Event Functionality

Events are generated as a result of a specific business condition being satisfied or triggered in the Siebel system. You can use events to trigger an action in your application. For example, an update to a database can reflect an update to customer information. If your application must perform an action when this happens, your application is a consumer of this event.

After you create a connection to your application system, you can add events using Servlet Application Explorer. To create an event, you must create a port and a channel.

The following is a description of how ports and channels work:

- Port

  A port associates a particular business object exposed by an adapter with a particular disposition. A disposition defines the protocol and location of the event data. The port defines the end point of the event consumption. For more information, see *Creating an Event Port* on page 3-2.

- Channel

  A channel represents configured connections to particular instances of back-end or other types of systems. A channel binds one or more event ports to a particular listener managed by an adapter. For more information, see *Creating a Channel* on page 3-16.

Creating an Event Port

You can listen for Siebel Integration Object events by configuring ports and channels. There are two methods to create an event in Application Explorer, through the Service Adapters tab or the Event Adapters tab. This section describes both methods.

Creating an Event Port From the Service Adapters Tab

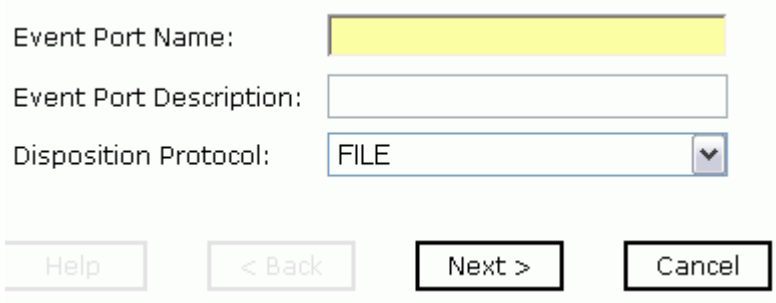
For Siebel Integration Objects, you can bypass the Event Adapters tab and create an event port directly from the Service Adapters tab.

Procedure: How to Create an Event Port From the Service Adapters Tab

To create an event port from the Service Adapters tab:

1. Select the Integration Object node you created.
2. Move the pointer over *Operations* and select *Create Event Port*.

The following image shows the Create Event Port pane that opens on the right where you can create a port.



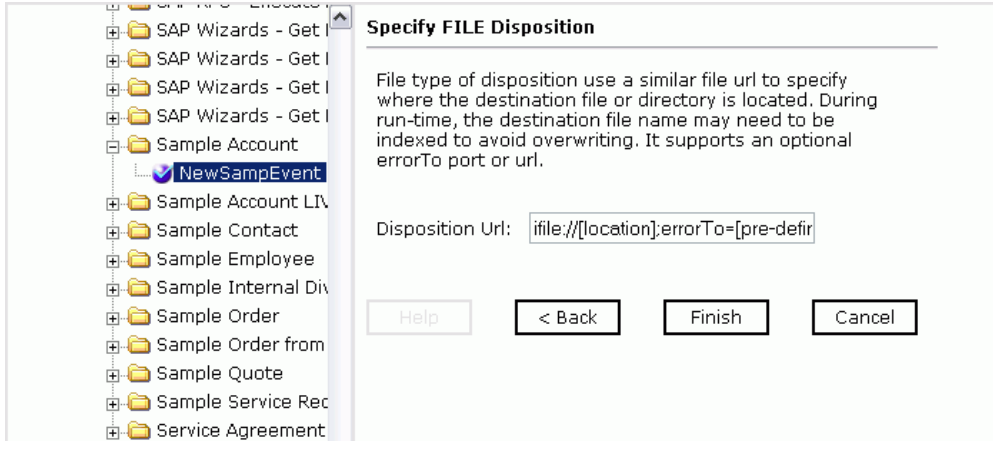
a. In the Event Port Name field, type a name for the port.

b. In the Event Port Description field, provide a brief description of the port.

c. From the Disposition Protocol drop-down list, select the required disposition, for example, FILE.

3. Click *Next*.

The following image shows the navigation pane on the left and the Specify Disposition pane that opens on the right and has information about the disposition type, a field for the disposition URL, a Help button, and three action buttons.



4. Type the disposition url and click *Finish*.

Creating an Event Port From the Event Adapters Tab

The following procedures describe how to create an event port from the Event Adapters window for various dispositions using Application Explorer.

The following dispositions are available when using the servlet Application Explorer in conjunction with an iBSE implementation. You can switch between an iBSE and a JCA implementation by choosing one or the other from the drop-down menu in the upper right of the Application Explorer.

- File
- iBSE
- MSMQ
- JMS queue
- SOAP
- HTTP
- MQSeries

Note: The MAIL disposition option will be supported in a future release.

The following dispositions are available when using Application Explorer in conjunction with a JCA connector implementation.

- File
- HTTP
- JMS queue
- MQSeries

To create an event port for Siebel Integration Objects, you must first indicate the location of the XDR schema for that object. For more information, see *Creating an Event Port* on page 3-2.

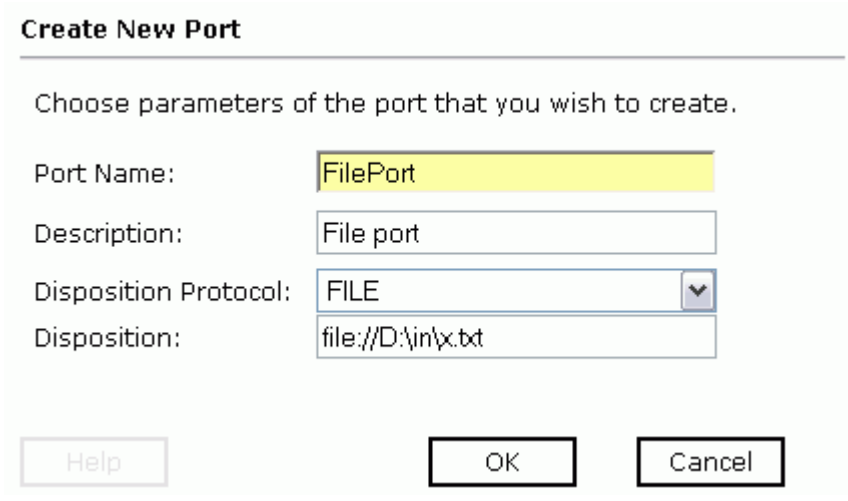
You also can create an event port directly from the Service Adapters tab. For more information, see *Creating an Event Port From the Service Adapters Tab* on page 3-2.

Procedure: How to Create an Event Port for the File Disposition

To create a specific event port for the File disposition:

1. Click the *Event Adapters* tab.
2. In the left pane, expand the *Siebel* node.
3. Select the *ports* node.
4. Move the pointer over *Operations* and select *Add a new port*.

The following image shows the Create New Port pane that opens on the right.



a. In the Port Name field, type a name for the event port.

b. In the Description field, provide a brief description of the port.

c. From the Disposition Protocol drop-down list, select *FILE*.

d. In the Disposition field, provide a destination where the event data is written.

When pointing Application Explorer to an **iBSE** deployment, use the following format:

```
ifile://[location];
errorTo=[pre-defined port name or another disposition url]
```

For example:

```
ifile://D:\in\x.txt;errorTo=ifile://D:\error
```

When pointing Application Explorer to a **JCA** deployment, provide the full path to the directory.

e. Provide values for the parameters from the following table.

The following table includes the name and description of each parameter.

Parameter	Description
location	The destination and filename of the document where event data is written, for example, D:\in\x.txt.

Parameter	Description
errorTo	Predefined port name or another disposition URL to which error logs are sent. Optional.

5. Click *OK*.

   The event port appears under the ports node in the left pane. In the right pane, a table appears that summarizes the information associated with the event port you created.

   You are ready to associate the event port with a channel. For more information, see *Creating a Channel* on page 3-16.

Procedure: How to Create an Event Port for iBSE

You can call Integration Business Services created through the Integration Business Services Engine (iBSE).

To create an event port for iBSE:

1. Click the *Event Adapters* tab.
2. In the left pane, expand the *Siebel* node.
3. Select the *ports* node.
4. Move the pointer over *Operations* and select *Add a new port*.

   The Create Event Port pane opens on the right.

   a. In the Port Name field, type a name for the connection.

      The name is used to build a repository entry as well as to identify the connection.

   b. In the Description field, type a description for the target name you just created.

   c. From the Disposition Protocol drop-down list, select *iBSE*.

   d. In the Disposition field, enter an iBSE destination in the form of:

```
ibse:svcName.mthName;
responseTo=[pre-defined port name or another disposition url];
errorTo=[pre-defined port name or another disposition url]
```

   e. Provide values for the parameters from the following table.

      The following table includes the name and description of each parameter.

Parameter	Description
svcName	Name of the service created with iBSE.

Parameter	Description
mthName	Name of the method created for the Web service.
responseTo	Location where responses to the Web service are posted. A predefined port name or another full URL. Optional.
errorTo	Location where error documents are sent. A predefined port name or another full URL. Optional.

5. Click *OK*.

   The event port appears under the ports node in the left pane. In the right pane, a table appears that summarizes the information associated with the event port you created.

   You are ready to associate the event port with a channel. For more information, see *Creating a Channel* on page 3-16.

Procedure: How to Create an Event Port for a JMS Queue

To create an event port for a JMS queue:

1. Click the *Event Adapters* tab.
2. In the left pane, expand the *Siebel* node.
3. Select the *ports* node.
4. Move the pointer over *Operations* and select *Add a new port*.

   The Create Event Port pane opens on the right.

   a. Type a name for the event port and provide a brief description.

   b. From the Disposition Protocol drop-down list, select *JMSQ*.

   c. In the Disposition field, enter a JMS destination.

      When pointing Application Explorer to an **iBSE** deployment, use the following format:

      ```
      jmsq:myQueueName@myQueueFac;jndiurl=[myurl];
      jndifactory=[myfactory];user=[user];password=[xxx];
      errorTo=[pre-defined port name or another disposition url]
      ```

      When pointing Application Explorer to a **JCA** deployment, use the following format:

      ```
      jms:jmsqueue@jmsfactory;jndiurl=;jndifactory=;
      ```

   d. Provide values for the parameters from the following table.

      The following table includes the name and description of each parameter.

Parameter	Description
queue	JNDI name of a queue to which events are emitted.
Connection Factory	A resource that contains information about the JMS Server. The WebLogic connection factory is: `javax.jms.QueueConnectionFactory`

jndiurl	The URL to use to contact the JNDI provider. The syntax of this URL depends on which JNDI provider is being used. This value corresponds to the standard JNDI property,
	`java.naming.provider.url`
	For BEA WebLogic Server this is
	`t3://host:port`
	where:
	host
	Is the machine name where WebLogic Server is installed.
	port
	Is the port on which WebLogic server is listening. The default port if not changed at installation is 7001.

jndifactory	Is JNDI context.INITIAL_CONTEXT_FACTORY and is provided by the JNDI service provider. For WebLogic Server, the WebLogic factory is `weblogic.jndi.WLInitialContextFactory`
user	A valid user name required to access a JMS server.
password	A valid password required to access a JMS server.
errorTo	Location where error documents are sent. A predefined port name or another full URL. Optional.

5. Click *OK*.

The event port appears under the ports node in the left pane. In the right pane, a table appears that summarizes the information associated with the event port you created.

You are now ready to associate the event port with a channel. For more information, see *Creating a Channel* on page 3-16.

Procedure: How to Create an Event Port for MSMQ

To create an event port for MSMQ:

1. Click the *Event Adapters* tab.
2. In the left pane, expand the *Siebel* node.
3. Select the *ports* node.
4. Move the pointer over *Operations* and select *Add a new port*.

   The Create Event Port pane opens on the right.

   a. In the Port Name field, type a name for the connection, for example, Queue1_on_NTK.

      The name is used to build a repository entry as well as to identify the connection.

   b. In the Description field, type a description for the target name you just created.

   c. From the Disposition Protocol drop-down list, select *MSMQ*.

   d. In the Disposition field, enter a MSMQ destination in the form of:

```
msmq:/host/private$/qName;
errorTo=[pre-defined port name or another disposition url]
```

   e. Provide values for the parameters from the following table.

      The following table includes the name and description of each parameter.

Parameter	Description
host	Machine name where the Microsoft Queuing system is running.
Queue Type	Private queues are queues that are not published in Active Directory. They appear only on the local computer that contains them. Private queues are accessible only by Message Queuing applications that recognize the full path name or format name of the queue. For private queues, enter *Private$*.
qName	Name of the private queue where messages are placed.
errorTo	Location where error documents are sent. A predefined port name or another full URL. Optional.

5. Click *OK*.

   The event port appears under the ports node in the left pane. In the right pane, a table appears that summarizes the information associated with the event port you created.

   You are ready to associate the event port with a channel. For more information, see *Creating a Channel* on page 3-16.

Procedure: How to Create a Port for a SOAP Disposition

To create a port for a SOAP disposition:

1. Click the *Event Adapters* tab.
2. In the left pane, expand the *Siebel* node.
3. Select the *ports* node.
4. Move the pointer over *Operations* and select *Add a new port*.

   The Create New Port pane opens on the right.

   a. In the Port Name field, type a name for the event.

   b. In the Description field, type a brief description.

   c. From the Disposition Protocol drop-down list, select *SOAP*.

   d. In the Disposition field, enter a SOAP destination, using the following format:

```
soap:[wsdl-url];soapaction=[myaction];
method=[web service method];namespace=[namespace];
responseTo=[pre-defined port name or another disposition URL];
errorTo=[pre-defined port name or another disposition url]
```

   e. Provide values for the parameters from the following table.

The following table includes the name and description of each parameter.

Parameter	Description
wsdl-url	The URL to the WSDL file that is required to create the SOAP message. For example: `http://localhost:7001/ibse/IBSEServlet/test/webservice.ibs?wsdl` where: *webservice* Is the name of the Web service you created using Application Explorer. This value can be found by navigating to the Integration Business Services tab and opening the Service Description link in a new window. The WSDL URL appears in the Address field. You also can open the WSDL file in a third party XML editor (for example, XMLSPY) and view the SOAP request settings to find this value.
soapaction	The method that will be called by the SOAP disposition. Integration Business ServicesThis value can be found in the WSDL file.
method	The Web service method you are using. This value can be found in the WSDL file.
namespace	The XML namespace you are using. This value can be found in the WSDL file.
responseTo	The location to which responses are posted, which can be a predefined port name or another URL. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.
errorTo	The location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

5. Click *OK*.

The event port appears under the ports node in the left pane. In the right pane, a table appears that summarizes the information associated with the event port you created.

Procedure: How to Create an Event Port for an HTTP Disposition

To create an event port for an HTTP disposition:

1. Click the *Event Adapters* tab.
2. In the left pane, expand the *Siebel* node.
3. Select the *ports* node.
4. Move the pointer over *Operations* and select *Add a new port*.

   The Create Event Port pane opens on the right.

   a. Type an event port name and a brief description.

   b. From the disposition protocol drop-down list, select *HTTP*.

   c. From the Disposition field, enter an HTTP destination.

      When pointing Application Explorer to an **iBSE** deployment, use the following format:

      ```
      ihttp://[myurl];
      responseTo=[pre-defined port name or another disposition url];
      ```

      The following table includes the name and description of each parameter.

Parameter	Description
url	Is the URL target for the post operation, for example `http://myhost:1234/docroot`
responseTo	Is the location where responses are posted (optional).

      When pointing Application Explorer to a **JCA** deployment, use the following format:

      ```
      http://host:port/uri
      ```

The following table includes the name and description of each parameter.

Parameter	Description
host:port	Is the combination of the name of the host on which the Web server resides and the port on which the server is listening for the post operation.
uri	Is the universal resource identifier that completes the url specification.

The event port appears under the ports node in the left pane. In the right pane, a table appears that summarizes the information associated with the event port you created.

5. Click *OK*.

   The port appears under the ports node in the left pane.

Procedure: How to Create an Event Port for an MQSeries Disposition

To create an event port for an MQSeries disposition using Application Explorer:

1. Click the *Event Adapters* tab.
2. In the left pane, expand the *Siebel* node.
3. Select the *ports* node.
4. Move the pointer over *Operations* and select *Add a new port*.

   The Create Event Port pane opens on the right.

   a. Type an event port name and a brief description.

   b. From the disposition protocol drop-down list, select *MQSeries*.

   c. In the Disposition field, enter an MQSeries destination.

      When pointing Application Explorer to an **iBSE** deployment, use the following format:

      ```
      mqseries:/qManager/qName;host=[hostname];
      port=[port];channel=[channnelname];
      errorTo=[pre-defined port name or another disposition url]
      ```

      When pointing Application Explorer to a **JCA** deployment, use the following format:

      ```
      mq:qmanager@respqueue;host=;port=;channel=
      ```

   d. Provide values for the parameters from the following table.

The following table includes the name and description of each parameter.

Parameter	Description
qManager	Is the name of the queue manager to which the server must connect.
qName or respqueue	Name of the queue where messages are placed.
host	The host on which the MQ Server is located (MQ Client only).
port	The number to connect to an MQ Server queue manager (MQ client only).
channel	The case-sensitive name of the channel that connects with the remote MQ Server queue manager (MQ client only). The default channel name for MQSeries is SYSTEM.DEF.SVRCONN.
errorTo	Location where error documents are sent. This can be a predefined port name or another full URL. Optional.

5. Click *OK*.

   The newly created event port appears under the port section of the event adapter in the left pane.

Editing or Deleting an Event Port

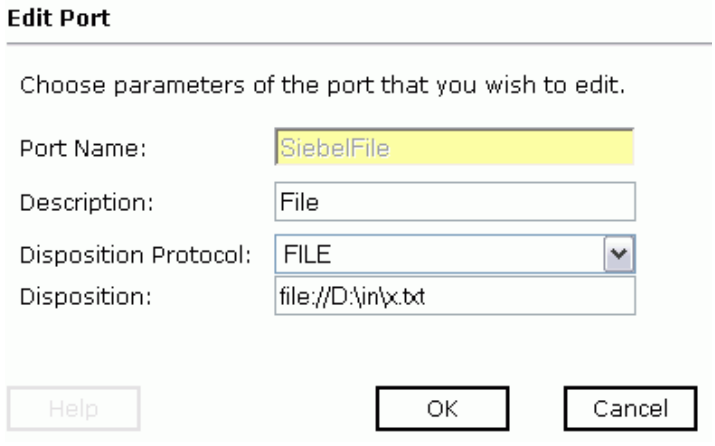
The following procedures provide information on how to modify or delete an event port.

Procedure: How to Edit an Event Port

To edit an existing event port:

1. In the left pane, select the event port you want to edit.
2. In the right pane, move the pointer over *Operations* and select *Edit*.

The following image shows the Edit Port pane that opens on the right. It includes a field for typing the port name, a field for typing a description, a drop-down list for selecting the disposition protocol, and a field for specifying the disposition. It also includes a Help button and two action buttons.



3. Make the required changes to the Description, Disposition Protocol, or Disposition fields, and click *OK*.

   Note: The Edit Port pane does not allow you to change the name of the port, only the parameters.

Procedure: How to Delete an Event Port

To delete an existing event port:

1. Select the event port you want to delete.
2. In the right pane, move the pointer over *Operations* and select *Delete*.

   A confirmation dialog box opens.
3. To delete the event port you selected, click *OK*.

   The event port disappears from the list in the left pane.

Using the Default Event Port

When using Application Explorer to connect to Siebel and listen for events, a default event port is available at all times as shown in the following image.

The default event port can be used for testing purposes or when you do not want to route event data to a specific port you configured. The default port is enabled when you start a channel that does not have a specific event port assigned.

The default event data is actually a file disposition that writes to an out.xml file in the following output directory:

```
ifile://./eventOut/out.xml
```

Creating a Channel

The following procedures describe how to create a channel for your event. All defined event ports must be associated with a channel. You can create three types of channels:

- HTTP
- File
- MQSeries

Procedure: How to Create an HTTP Channel

To create an HTTP channel using Application Explorer:

1. Click the *Event Adapters* tab.

   The adapters in the left pane support events.

2. Expand the *Siebel* node.

   The ports and channels nodes appear in the left pane.

3. Click the *channels* node.

4. In the right pane, move the pointer over *Operations* and select *Add a new channel*.

   a. When the Add a new channel window opens, type a name for the channel, for example, NewChannel.

   b. Type a brief description.

   c. From the drop-down list, select *HTTP Listener*.

5. Click *Next*.

   The Edit Channels window opens in the right pane.

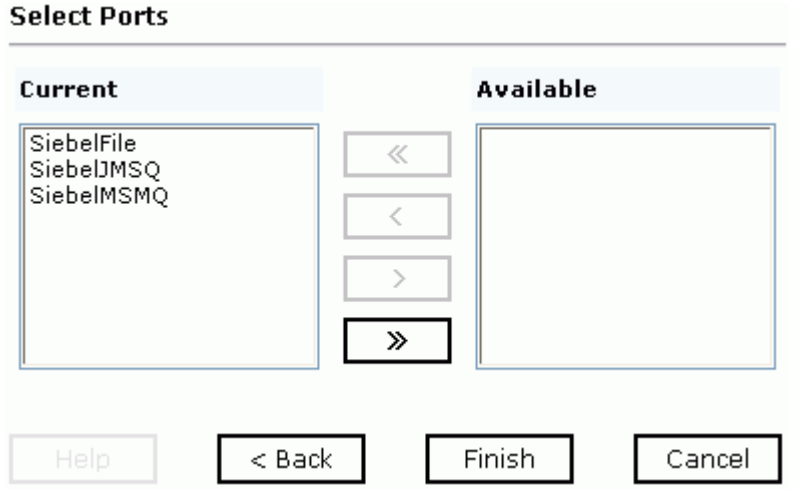
6. Provide values for the parameters from the following table.

The following table includes the name and description of each parameter.

Parameter	Description
Listener port	Port on which to listen for Siebel event data.
Synchronization Type	Sync hronization types are not applicable to Siebel events.

7. Click *Next*.

The following image shows the Select Ports pane that opens with a list of the ports that are currently associated in the Current field on the left. On the right is a list of available ports in the Available field. The pane also includes arrow buttons, a Help button, and three action buttons.



a. Select an event port from the list of current ports.

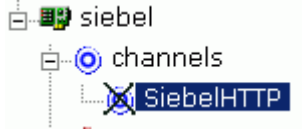
b. Click the single right arrow button to transfer the port to the list of available ports. To associate all the event ports, click the double right arrow button.

8. Click *Finish*.

   The following image is an example of the summary window that opens, showing a description of the channel, its status, and available ports.

   All the information in the summary is associated with the channel you created.

   The following image shows a channel that appears beneath the channels node in the left pane. An X over the icon indicates that the channel is currently disconnected. You must start the channel to activate your event configuration.



9. In the right pane, move the pointer over *Operations* and select *Start the channel*.

   The channel you created becomes active.

   The X that was over the icon disappears.

10. To stop the channel, move the pointer over *Operations* and select *Stop the channel*.

Procedure: How to Create a File Channel

To create a channel using Application Explorer:

1. Click the *Event Adapters* tab.

   The adapters in the left pane support events.

2. Expand the *Siebel* node.

   The ports and channels nodes appear in the left pane.

3. Click the *channels* node.

4. In the right pane, move the pointer over *Operations* and select *Add a new channel*.

   The Add a new channel window opens.

   a. Type a name for the channel, for example, NewChannel.

   b. Type a brief description.

   c. From the drop-down list, select *File Listener*.

5. Click *Next*.

The Edit Channels window opens with three tabs in the right pane.

a. In the Request tab, provide values for the parameters from t.

The following table includes the name and description of each parameter.

Parameter	Description
Polling Location	The target file system location for the Siebel XML file.
File Mask	The file name to be used for the output file generated as a result of this operation.

b. In the Response tab, provide values for the parameters from the following table.

The following table includes the name and description of each parameter.

Parameter	Description
Synchronization Type	Sync hronization types are not applicable to Siebel events.
Response/Ack Directory	Choose from three options: • REQUEST • REQUEST_RESPONSE • REQUEST_ACK

c. In the Advanced tab, provide values for the parameters from the following table.

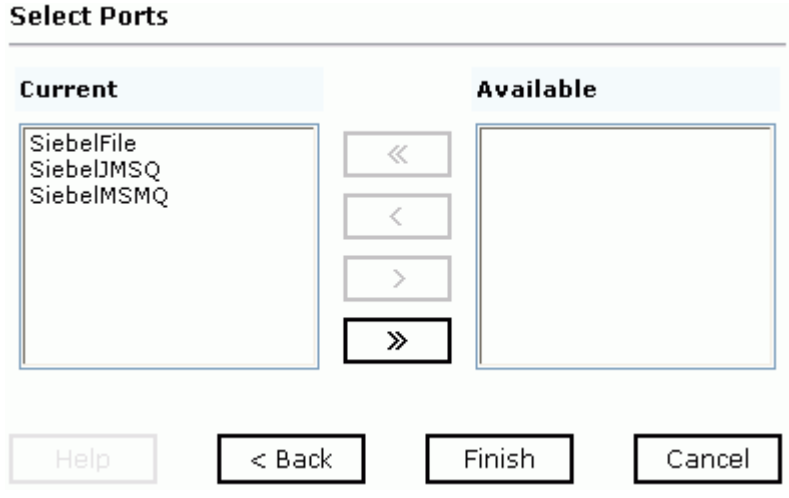
The following table includes the name and description of each parameter.

Parameter	Description
Error Directory	Directory to which documents with errors are written.
Poll interval (msec):	The interval (in milliseconds) when to check for new input. Optional. The default is 3 seconds.
Processing Mode	Choose Sequential or Threaded. • Sequential indicates single processing of requests. • Threaded indicates processing of multiple requests simultaneously.

Parameter	Description
Thread limit	If you selected threaded processing, indicate the maximum number of requests that can be processed simultaneously.

6. Click *Next*.

The following image shows the Select Ports pane that opens with a list of the ports that are currently associated in the Current field on the left. On the right is a list of available ports in the Available field. The pane also includes arrow buttons, a Help button, and three action buttons.



a. Select an event port from the list of current ports.

b. Click the single right arrow button to transfer the port to the list of available ports. To associate all the event ports, click the double right arrow button.

7. Click *Finish*.

The summary window opens. A summary provides the channel description, channel status, and available ports. All the information is associated with the channel you created.

The channel appears under the channels node in the left pane. An X over the icon indicates that the channel is currently disconnected. You must start the channel to activate your event configuration.

8. In the right pane, move the pointer over *Operations* and select *Start the channel*.

   The channel you created becomes active.

   The X that was over the icon disappears.

9. To stop the channel, move the pointer over *Operations* and select *Stop the channel*.

Procedure: How to Create an MQSeries Channel

To create an MQSeries channel using Application Explorer:

1. Click the *Event Adapters* tab.

   The adapters in the left pane support events.

2. Expand the *Siebel* node.

   The ports and channels nodes appear in the left pane.

3. Click the *channels* node.

4. In the right pane, move the pointer over *Operations* and select *Add a new channel*.

   The Add a new channel window opens.

   a. Type a name for the channel, for example, NewChannel.

   b. Type a brief description.

   c. From the drop-down list, select *MQ Series Listener.*

5. Click *Next*.

   The Edit Channels panes opens on the right and has three tabs.

   a. In the Request tab, provide values for the parameters from the following table.

      The following table includes the name and description of each parameter.

Parameter	Description
Queue manager name	Name of the queue manager to which the server must connect.
MQ server host for MQClient operation	Host on which the MQ Server is located (MQ Client only).
MQ server port for MQClient operation	The number to connect to an MQ Server queue manager (MQ client only).

Parameter	Description
MQ server channel for MQClient operation	The case-sensitive name of the channel that connects with the remote MQ Server queue manager (MQ client only). The default channel name for MQSeries is SYSTEM.DEF.SVRCONN.
Document type XML	Keep the default selection.
Request queue name	Queue where the message is routed and where request documents are received. The name of the queue is case-sensitive.

b. In the Response tab, provide values for the parameters from the following table.

The following table includes the name and description of each parameter.

Parameter	Description
Synchronization Type	Sync hronization types are not applicable to Siebel events.

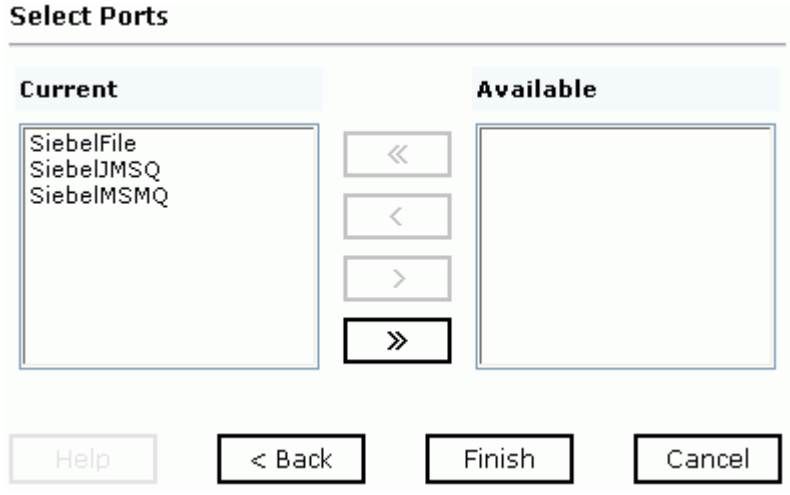
c. In the Advanced tab, provide values for the parameters from the following table.

The following table includes the name and description of each parameter.

Parameter	Description
Error Directory	Directory to which documents with errors are written.
Message wait interval (msec):	The interval (in milliseconds) when to check for new input Optional. The default is 3 seconds.
Mode of operation	Choose Sequential or Threaded. • Sequential indicates single processing of requests. • Threaded indicates processing of multiple requests simultaneously.
Thread limit	If you selected threaded processing, indicate the maximum number of requests that can be processed simultaneously.

6. Click *Next*.

The following image shows the Select Ports pane that opens with a list of the ports that are currently associated in the Current field on the left. On the right is a list of available ports in the Available field. The pane also includes arrow buttons, a Help button, and three action buttons.



a. Select an event port from the list of current ports.

b. Click the single right arrow button to transfer the port to the list of available ports. To associate all the event ports, click the double right arrow button.

7. Click *Finish*.

The summary window opens. A summary provides the channel description, channel status, and available ports. All the information is associated with the channel you created.

The channel appears under the channels node in the left pane. An X over the icon indicates that the channel is currently disconnected. You must start the channel to activate your event configuration.

8. In the right pane, move the pointer over *Operations* and select *Start the channel*.

The channel you created becomes active.

The X that was over the icon disappears.

9. To stop the channel, move the pointer over *Operations* and select *Stop the channel*.

Procedure: How to Edit a Channel

To edit an existing channel:

1. In the left pane, select the channel you want to edit.
2. In the right pane, move the pointer over *Operations* and select *Edit*.

   The Edit channels window opens.
3. Make the required changes to the channel configuration and click *Finish*.

Procedure: How to Delete a Channel

To delete an existing channel:

1. In the left pane, select the channel you want to delete.
2. In the right pane, move the pointer over *Operations* and select *Delete*.

   A confirmation dialog box opens.
3. To delete the channel you selected, click *OK*.

   The channel disappears from the list in the left pane.

Deploying Components in a Clustered BEA WebLogic Environment

Events can be configured in a clustered BEA WebLogic environment. You can deploy iBSE or JCA to this environment. This topic uses iBSE as an example, but you can follow the same procedures when deploying JCA. The only difference is that you need to deploy the JCA connector .RAR file to the clustered environment.

A cluster consists of multiple server instances running simultaneously, yet appears to clients to be a single server instance. The server instances that contain a cluster can be run on one machine, but are usually run on multiple machines.

Clustering provides the following benefits:

- Load balancing. For more information, see the BEA clustering documentation.
- High availability

Service requests are processed through the HTTP router and routed to an available managed server.

Events are server-specific and are not processed through the HTTP router. You must configure each server separately.

Procedure: How to Deploy Components in a Clustered Environment

To deploy components in a clustered environment:

1. Using the BEA Configuration Wizard, configure an administrative server to manage the managed servers.
2. Add and configure as many managed servers as required.
3. Add and configure an HTTP router. This does not have to be a part of WebLogic and can be an outside component.
4. If you configure the HTTP router within WebLogic, start it by entering the following command:

   `StartManagedWebLogic` *`HTTPROUTER`* `http://`*`localhost`*`:7001`

   where:

   `HTTPROUTER`

   Is the name of the server on which the HTTP router is running.

   `http://`*`localhost`*`:7001`

   Is the location of the admin console.
5. Add the managed servers to your cluster/clusters.

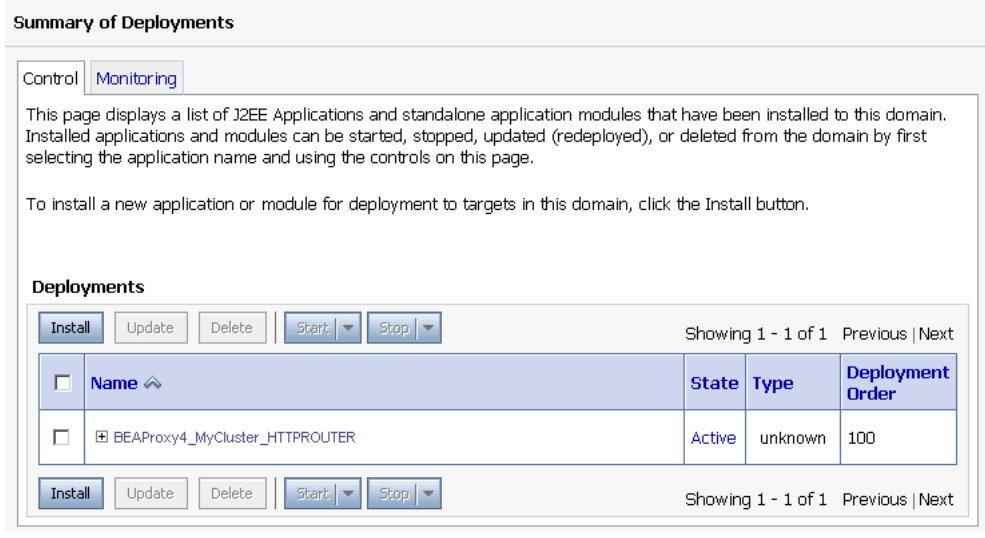
   For more information on configuring the BEA WebLogic Server in a clustered environment, see *Deploying WebLogic Integration Solutions*.

Procedure: How to Deploy iBSE in a Clustered Environment

To deploy iBSE in a clustered environment:

1. Start the BEA WebLogic Server and open the WebLogic Server Console.
2. In the left pane, click *Lock & Edit* and click *Deployments*.

The Summary of Deployments page opens. Notice that the cluster you configured using the BEA Configuration Wizard is added to the list of deployments in the WebLogic Server Console.



3. Click *Install*.

   A page appears where you can specify the location of the file or directory you wish to deploy.

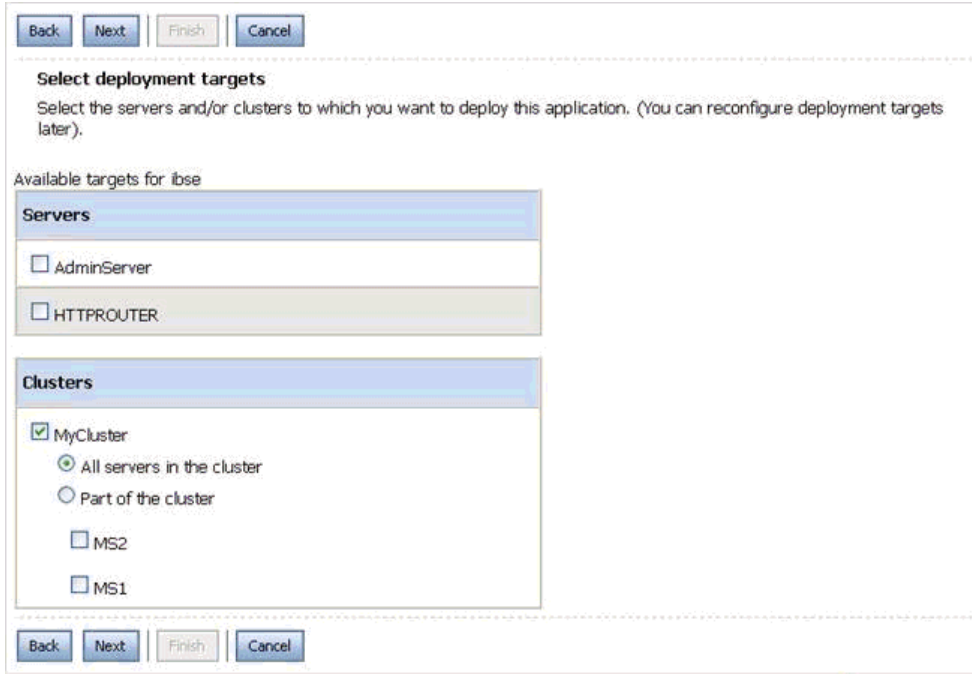
4. Deploy iBSE to the cluster by clicking the links below *Location* and then selecting the radio button next to the ibse directory, for example:

   ```
   C:\Program Files\iWay55\bea\ibse
   ```

5. Click the radio button next to the ibse directory and click *Next*.

6. Click *Next* again, leaving the default *Install this deployment as an application* option selected.

The Select deployment targets page opens.



7. In the Clusters section, click the checkbox next to the name of the cluster you configured, for example *MyCluster*.
8. Click the *All servers in the cluster* radio button and click *Next*.
9. Click *Next* again, leaving the default values.
10. Click *Finish* to complete the deployment.
11. On the left, click *Activate Changes*.
12. On the right, click *Control*.
13. Click the *ibse* checkbox.
14. Click *Start* and select *Servicing All Requests* from the drop-down menu.
15. Click *Yes*.

    The State of the iBSE application is now *Active*.

Note: iBSE must use a database repository (SQL Server, Oracle, DB2, or Sybase). Do not use a file repository. You can select this in the Repository Type drop-down list in the iBSE monitoring page. For more information, see the *BEA WebLogic ERP Adapter Installation and Configuration* documentation. After configuring a database repository, you must restart all of the managed servers.

`http://hostname:port/ibse/IBSEConfig/`

where:

hostname

Is where your application server is running. Use the IP address or machine name in the URL; do not use localhost.

port

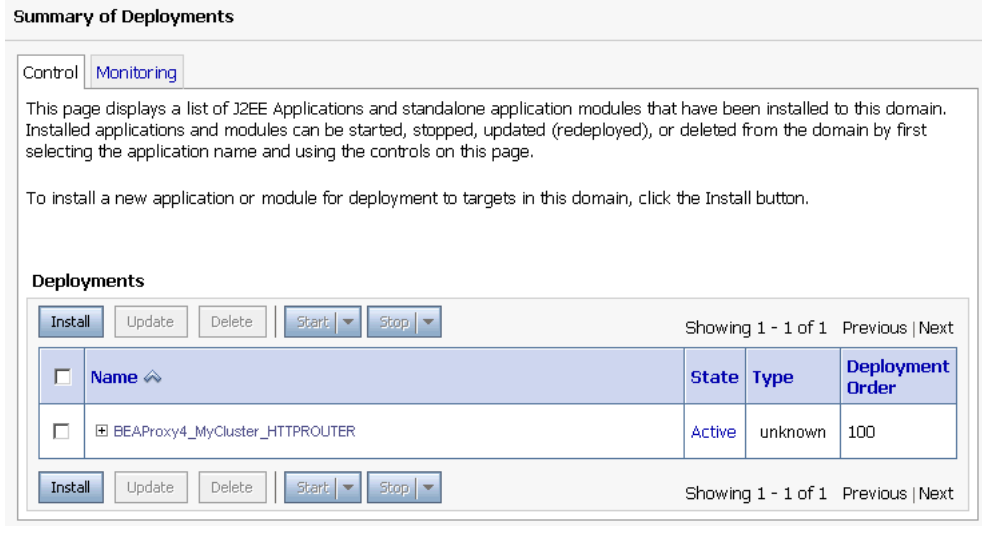
Is the port specific to each server, since you deploy iBSE to an entire cluster. For example, 8001, 8002, or any other port that is specified for each managed node.

Procedure: How to Deploy Servlet Application Explorer to an Admin Server

To deploy Servlet Application Explorer to an Admin server:

1. Start the BEA WebLogic Server and open the WebLogic Server Console.
2. In the left pane, click *Lock & Edit* and click *Deployments*.

   The Summary of Deployments page opens. Notice that the cluster you configured using the BEA Configuration Wizard is added to the list of deployments in the WebLogic Server Console.



3. Click *Install*.

   A page appears where you can specify the location of the file or directory you wish to deploy.

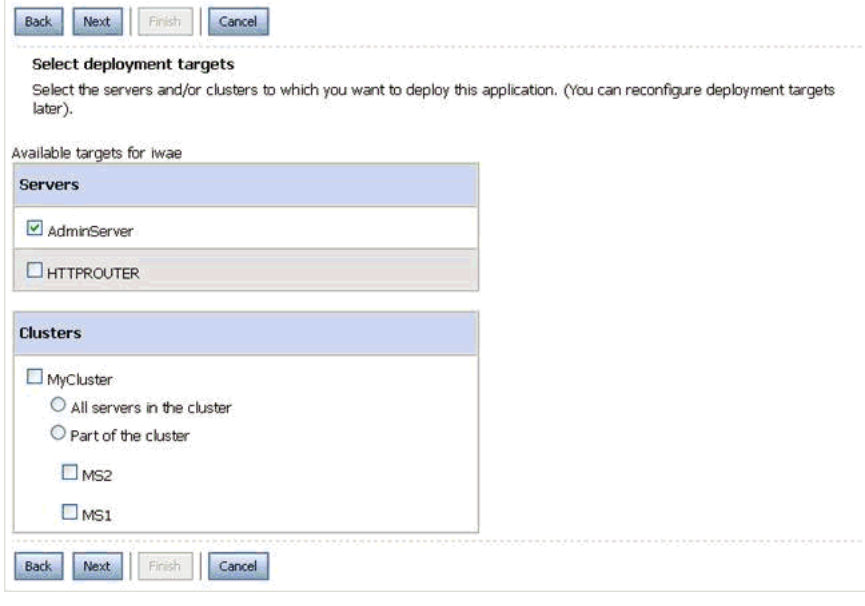
4. Deploy Servlet Application Explorer to the cluster by clicking the links below *Location* and then selecting the radio button next to the ibse directory, for example:

   ```
   C:\Program Files\iWay55\bea\iwae
   ```

5. Click *Next* after selecting the radio button next to the iwae directory.

6. Click *Next* again, leaving the default *Install this deployment as an application* option selected.

   The Select deployment targets page opens.



7. In the Servers section, click the AdminServer checkbox and click *Next*.

8. Click *Next* again, leaving the default values.

9. Click *Finish* to complete the deployment.

10. On the left, click *Activate Changes*.

11. On the right, click *Control*.

12. Click the *iwae* checkbox.

13. Click *Start* and select *Servicing All Requests* from the drop-down menu.

14. Click *Yes*.

The State of the iwae application is now *Active*.

Procedure: How to Configure Ports and Channels in a Clustered Environment

You can use Servlet Application Explorer to configure ports and channels in a clustered environment.

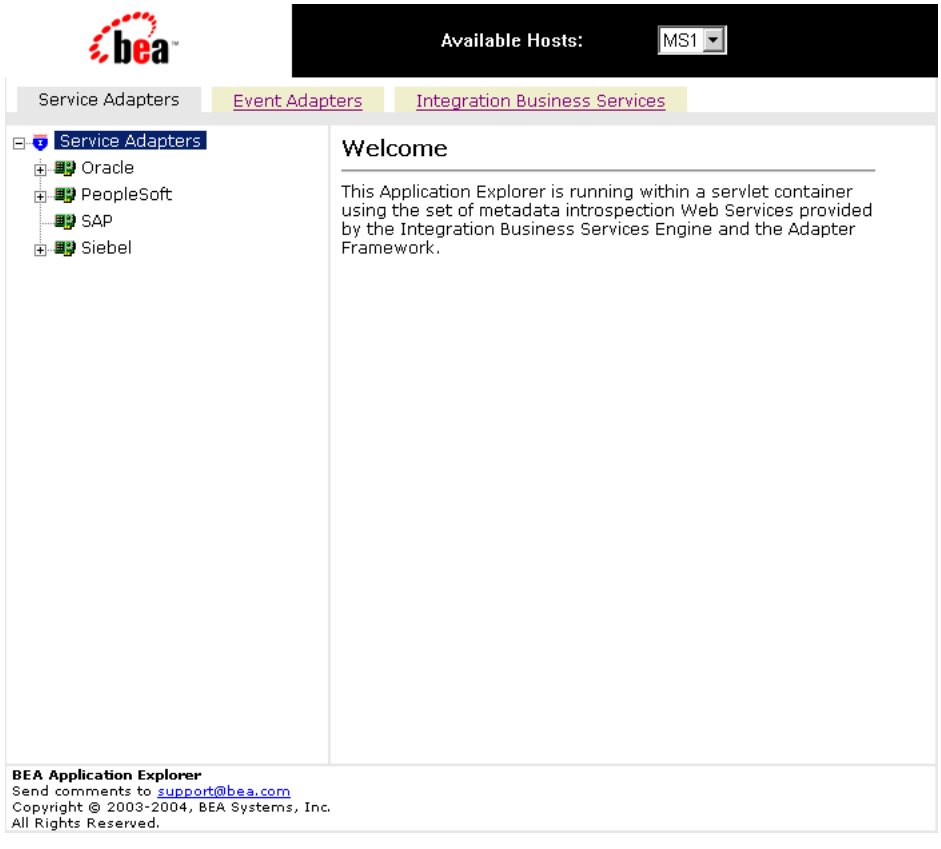
Note: Before using Servlet Application Explorer in a clustered environment, you must edit the web.xml file and specify the correct URL to your iBSE deployment. The default location on Windows is:

```
C:\Program Files\iWay55\bea\iwae\WEB-INF\web.xml
```

For more information on configuring the web.xml file for the Servlet Application Explorer, see the *BEA WebLogic ERP Adapter Installation and Configuration* documentation.

To configure ports and channels in a clustered environment:

1. Start Servlet Application Explorer.



The Available Hosts drop-down menu in the upper right lists the JCA or iBSE instances in your cluster, for example, MS1 and MS2.

2. Click the *Event Adapters* tab.
3. Select an adapter from the adapter list (in this example, Siebel) and add a new port. For more information, see *Creating an Event Port* on page 3-2.
4. Create a channel and add the port you created. For more information, see *Creating a Channel* on page 3-16.
5. Enter the application server parameters for your channel.
6. Start the channel.
7. Select the second iBSE instance, for example MS2, from the Available Hosts drop-down menu.

   The connection to iBSE must be configured to each instance of the managed server.

   The following operations performed on one managed server will be replicated on all other managed servers:

   - Create port and channel. Creates the channel and port under all available servers.
   - Delete port and channel. Deletes the port and channel under all available servers.

   The following operations must be performed on each server:

   - Start channel. Starts the channel for the specific server.
   - Stop channel. Stops the channel for the specific server.

CHAPTER 4

Management and Monitoring

Topics:

- Managing and Monitoring Services and Events Using iBSE
- Managing and Monitoring Services and Events Using the JCA Test Tool
- Setting Engine Log Levels
- Configuring Connection Pool Sizes
- Migrating Repositories
- Exporting or Importing Targets
- Retrieving or Updating Web Service Method Connection Information
- Starting or Stopping a Channel Programmatically

After you create services and events using Application Explorer, you can use management and monitoring tools to measure the performance in your run-time environment. This section describes how to configure and use these features.

Managing and Monitoring Services and Events Using iBSE

Integration Business Services Engine (iBSE) provides a console that enables you to manage and monitor services and events currently in use. The console also displays resource usage and invocation statistics. These indicators can help you adjust your environment for optimum efficiency.

The following monitoring levels are available for services:

- System
- Service
- Method

The following monitoring levels are available for events:

- System
- Channel
- Port

Procedure: How to Configure Monitoring Settings

To configure monitoring settings:

1. Ensure that your BEA WebLogic Server is started.
2. To access the monitoring console, enter the following URL in your Web browser:

   `http://hostname:port/ibse/IBSEConfig`

   where:

   hostname

   Is the machine where the application server is running.

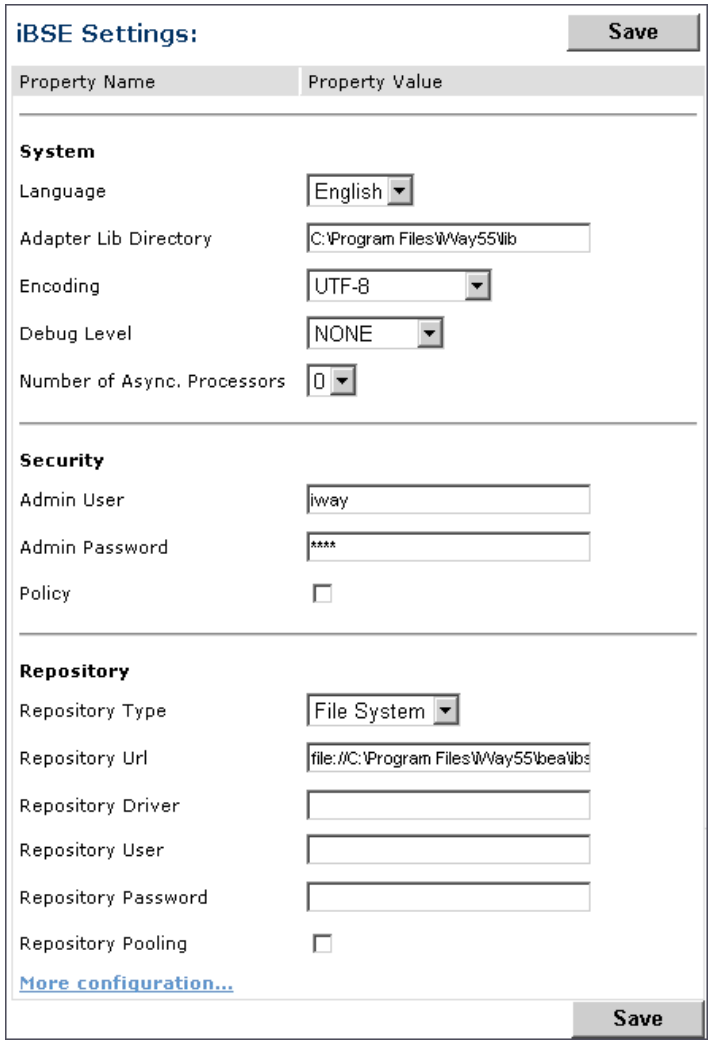
   port

   Is the HTTP port for the application server.

   The iBSE Settings window opens as shown in the following image. It consists of three panes. To configure system settings, the System pane includes drop-down lists for selecting language, encoding, the debug level, and the number of asynchronous processors. It also contains a field where you can enter a path to the adapter lib directory.

   To configure security settings, the Security pane contains fields for typing the Admin User name and the associated password and a check box for specifying policy.

To configure repository settings, the Repository pane contains a drop-down list for selecting the repository type; fields to type information for the repository URL, driver, user, and password; and a check box where you can enable repository pooling. In the upper and lower right of the window is a Save button. In the lower left of the window is an option to access more configuration settings.



3. Click *More configuration*.

Tip: To access the monitoring console directly, enter the following URL in your Web browser:

`http://hostname:port/ibse/IBSEStatus`

where:

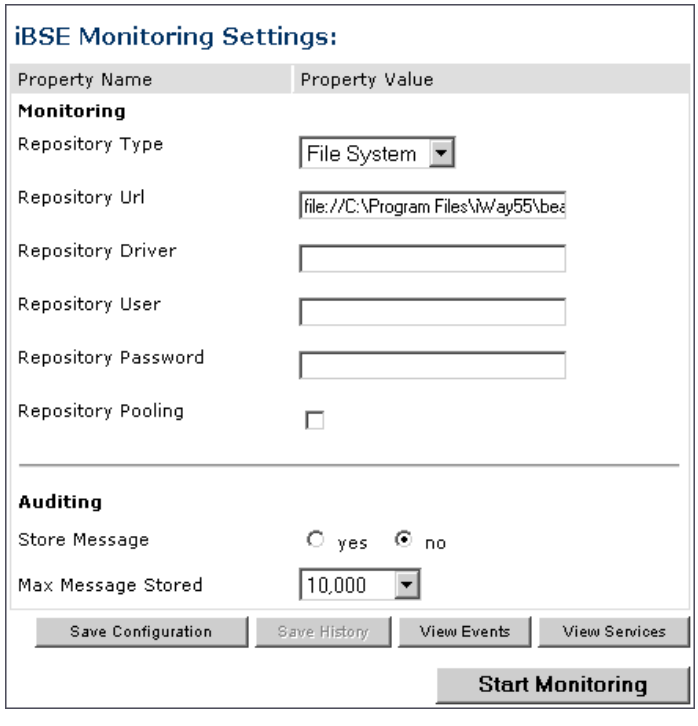
hostname

Is the machine where the application server is running.

port

Is the HTTP port for the application server.

The iBSE Monitoring Settings window which is divided into two panes opens as shown in the following image. At the bottom of the window is a row of command buttons that enable you to save your configuration, view events, or view services. The Save History button is inactive.



a. In the Monitoring pane, from the Repository Type drop-down list, select the type of repository you are using.

b. To connect to the database in the Repository Url field, type a JDBC URL.

c. To connect to the database in the Repository Driver field, type a JDBC Class.

d. To access the monitoring repository database, type a user ID and password.

e. To enable pooling, select the *Repository Pooling* check box.

f. In the Auditing pane, click *yes* if you want to store messages.

This option is disabled by default.

Note: You must start and then stop monitoring to enable this option.

g. From the drop-down list, select the maximum number of messages to store.

By default, 10,000 is selected.

Note: Depending on your environment and the number of messages that are exchanged, storing a large number of messages may affect system performance. If you require more information about your system resources, consult your system administrator.

h. Click *Save Configuration*.

4. Click *Start Monitoring*.

iBSE begins to monitor all services and events currently in use. If you selected the option to store messages, iBSE stores messages.

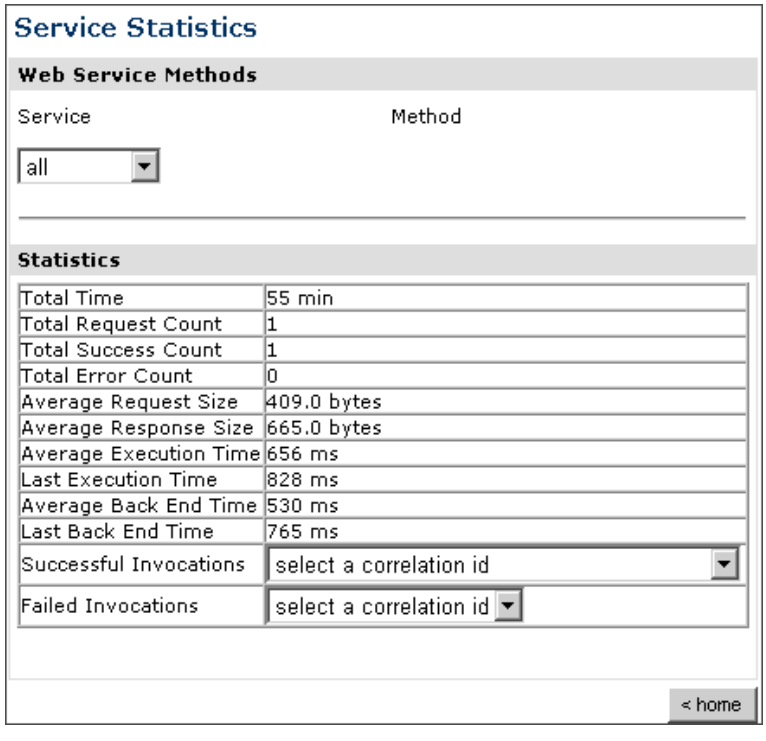
5. To stop monitoring, click *Stop Monitoring*.

Procedure: How to Monitor Services

To monitor services:

1. Ensure that your BEA WebLogic Server is started.
2. From the iBSE Monitoring Settings window, click *Start Monitoring*.
3. Click *View Services*.

   The System Level Summary (Service Statistics) window opens, as shown in the following image. The Web Service Methods pane contains a drop-down list where you select a service. On the right, space is reserved for a drop-down list of methods that will appear. The Statistics pane contains a table with a summary of service statistics and two drop-down lists where you can select a successful or failed invocation to view more information about that service. At the bottom of the window is a home button to click to return to the iBSE Monitoring Settings window.



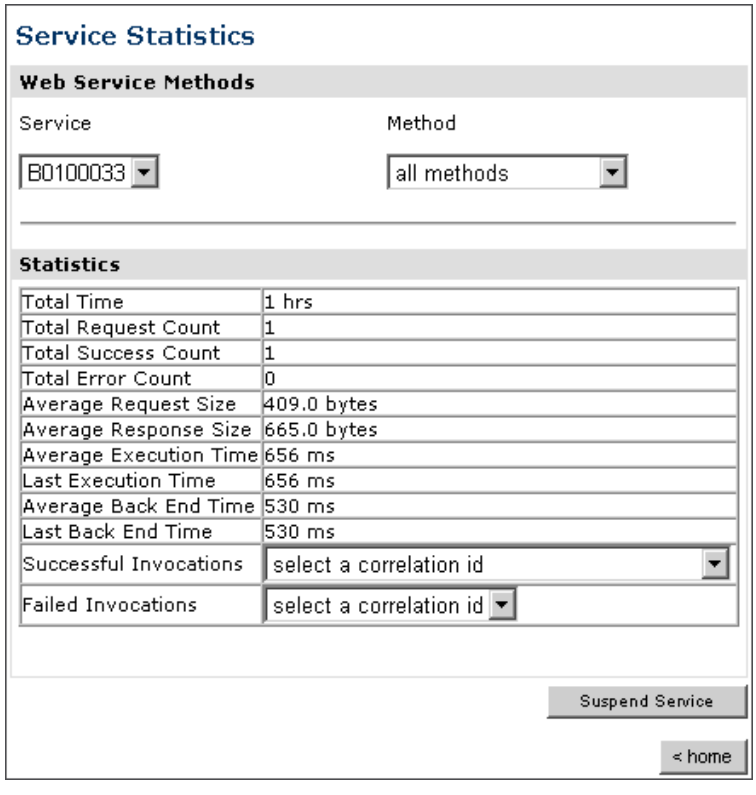
   The system level summary provides services statistics at a system level.

The following table lists and describes each service statistic.

Statistic	**Description**
Total Time	Total amount of time iBSE monitors services. The time starts after you click Start Monitoring in the iBSE Monitoring Settings window.
Total Request Count	Total number of services requests made during the monitoring session.
Total Success Count	Total number of successful service executions.
Total Error Count	Total number of errors encountered.
Average Request Size	Average size of an available service request.
Average Response Size	Average size of an available service response.
Average Execution Time	Average execution time for a service.
Last Execution Time	Last execution time for a service.
Average Back End Time	Average back-end time for a service.
Last Back End Time	Last back-end time for a service.
Successful Invocations	Successful services arranged by correlation ID. To retrieve more information for a service, select the service from the drop-down list.
Failed Invocations	Failed services arranged by correlation ID. To retrieve more information for a service, select the service from the drop-down list.

4. Select a service from the drop-down list.

   The System Level Summary (Service Statistics) window opens as shown in the following image. The Web Service Methods pane contains a drop-down list on the left where you select a service and a drop-down list on the right where you select a service method. The Statistics pane contains a table with a summary of service statistics and two drop-down lists. To view more information about that service, you can select it from the Successful Invocations or Failed Invocations drop-down list. To suspend or resume a service, you can click a button in the lower right. To return to the iBSE Monitoring Settings window, you click the home button.

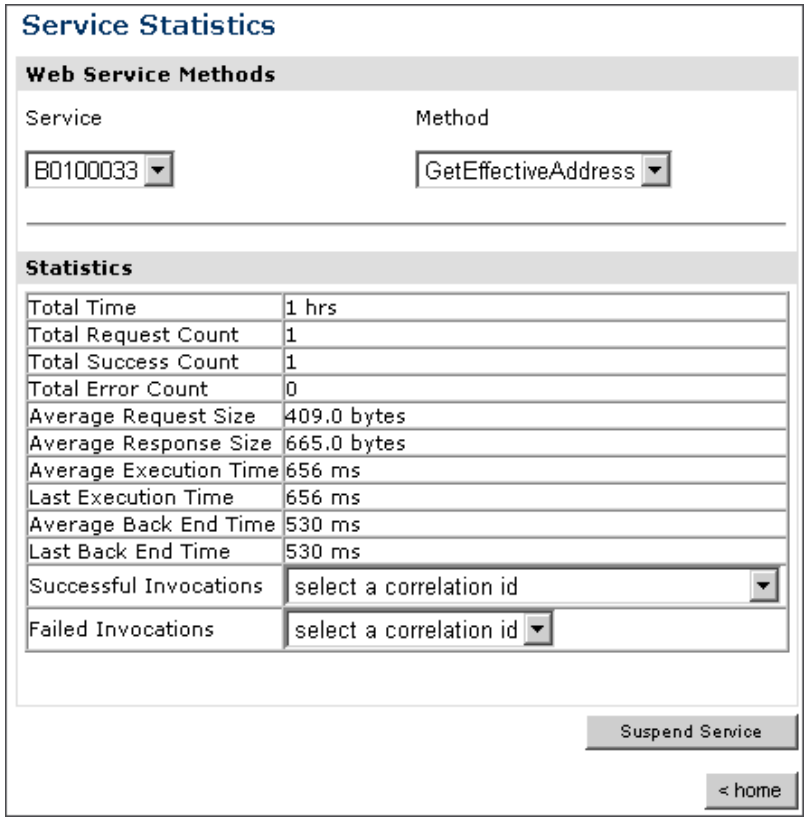


   a. To stop a service at any time, click *Suspend Service*.

   b. To restart the service, click *Resume Service*.

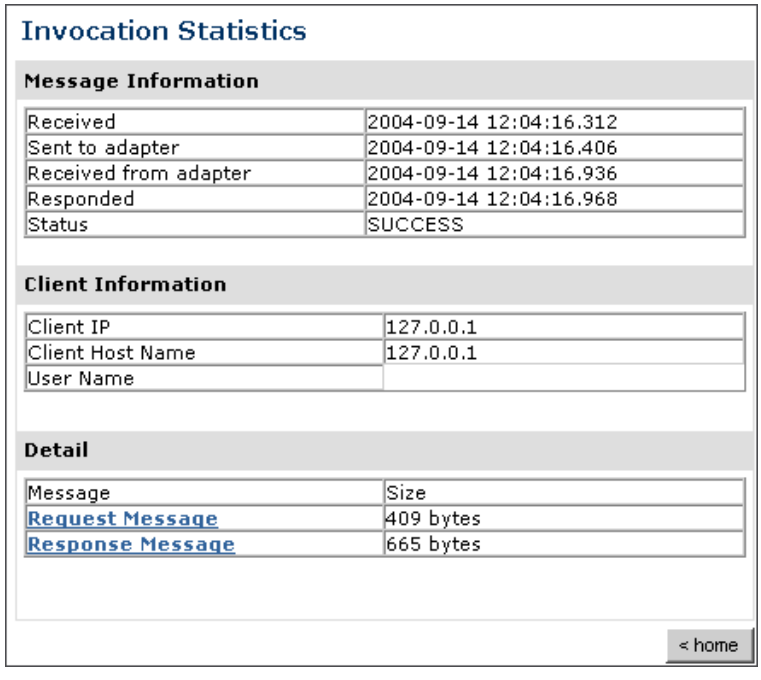
5. Select a method for the service from the Method drop-down list.

The Method Level Summary (Service Statistics) window opens as shown in the following image. The Web Service Methods pane contains a drop-down list on the left where you select a service and a drop-down list on the right where you select a service method. The Statistics pane contains a table with a summary of service statistics and two drop-down lists. To view more information about that service, you can select it from the Successful Invocations or Failed Invocations drop-down list. To suspend or resume a service, you can click a button in the lower right. To return to the iBSE Monitoring Settings window, you click the home button.



6. For additional information about a successful service and its method, select a service based on its correlation ID from the Successful Invocation drop-down list.

   The Invocation Level Statistics window opens as shown in the following image. The Message Information pane contains a table of information about the message. The Client Information pane contains a table of information about the client. The Detail pane contains a table that shows the size of the request and response messages.



7. To view the XML request document in your Web browser, click *Request Message*.

   You can also view the XML response document for the service.

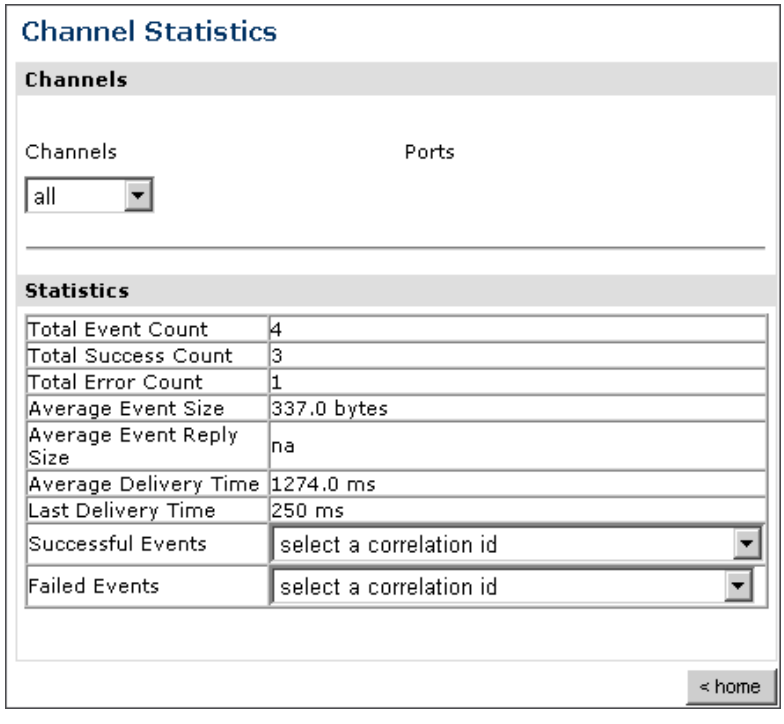
8. To return to the iBSE Monitoring Settings window, click *home*.

Procedure: How to Monitor Events

To monitor events:

1. Ensure that your BEA WebLogic Server is started.
2. In the iBSE Monitoring Settings window, click *Start Monitoring*.
3. Click *View Events*.

   The System Level Summary (Channel Statistics) window opens as shown in the following image. The Channels pane contains a drop-down list on the left where you select a channel. On the right, space is reserved for a drop-down list of ports that will appear. The Statistics pane contains a table with a summary of event statistics and two drop-down lists where you can select a successful or failed event to view more information about that event. In the lower right of the window is a home button.



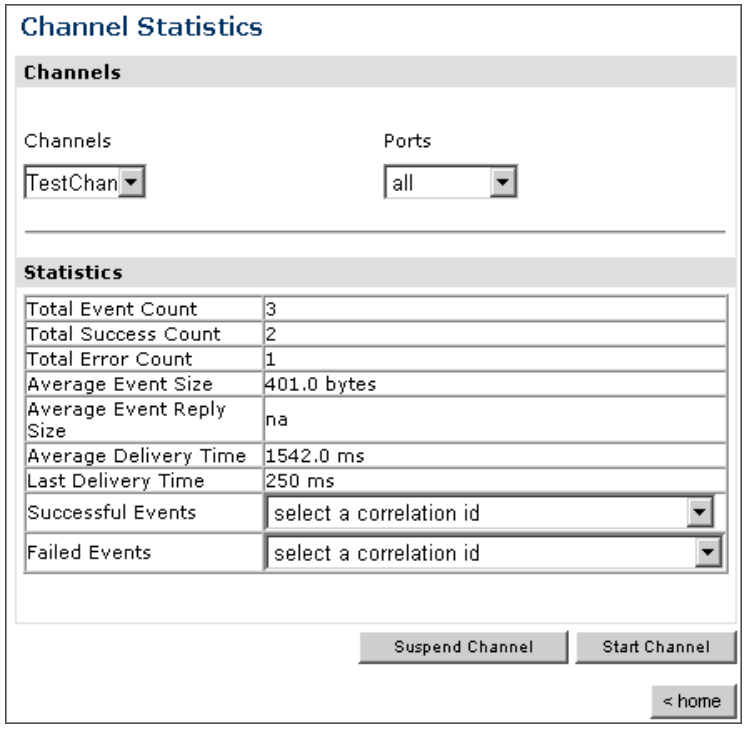
   The system level summary provides event statistics at a system level.

The following table lists and describes each event statistic.

Statistic	**Description**
Total Event Count	Total number of events.
Total Success Count	Total number of successful event executions.
Total Error Count	Total number of errors encountered.
Average Event Size	Average size of an available event request.
Average Event Reply Size	Average size of an available event response.
Average Delivery Time	Average delivery time for an event.
Last Delivery Time	Last delivery time for an event.
Successful Events	Successful events arranged by correlation ID. To retrieve more information for an event, select the event from the drop-down list.
Failed Events	Failed events arranged by correlation ID. To retrieve more information for an event, select the event from the drop-down list.

4. Select a channel from the drop-down list.

   The Channel Level Event Summary (Channel Statistics) window opens as shown in the following image. The Channels pane contains a drop-down list on the left where you select a channel and a drop-down list on the right where you select a port. The Statistics pane contains a table with a summary of event statistics and two drop-down lists where you can select a successful or failed event to view more information about that event. In the lower right of the window is a button to click to suspend or resume a channel and a home button to click to return to the iBSE Monitoring Settings window.

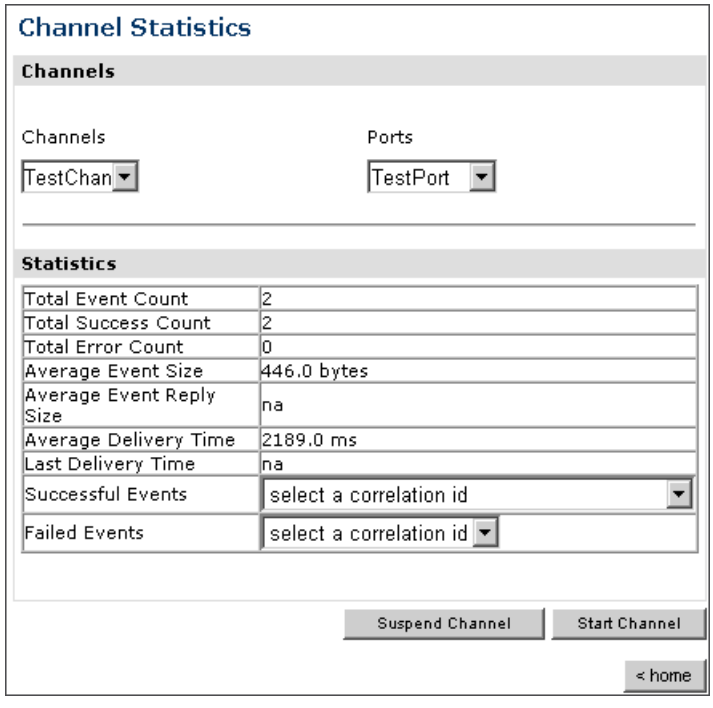


   a. To stop a channel at any time, click *Suspend Channel*.

   b. To start the channel, click *Start Channel*.

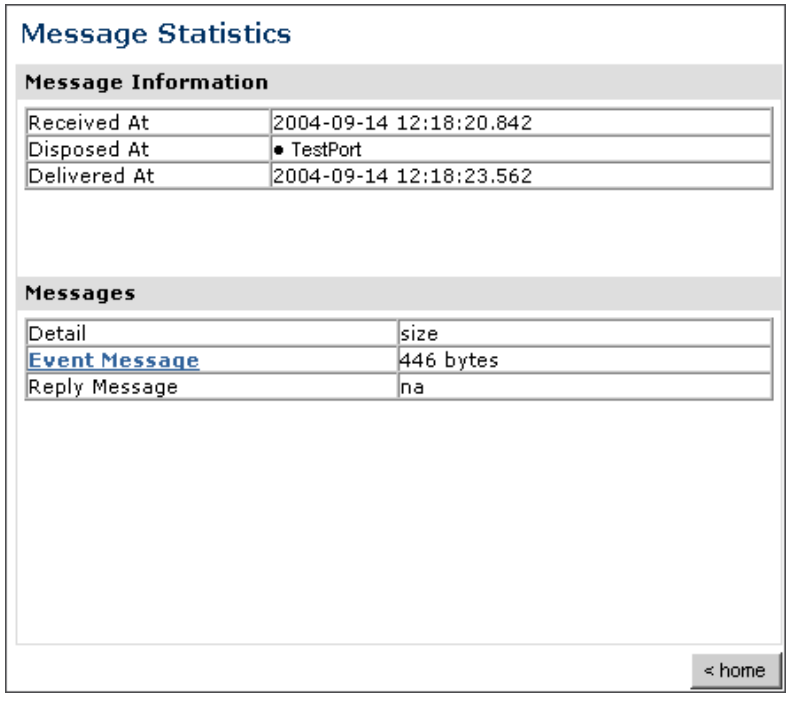
5. From the Ports drop-down list, select a port for the channel.

The Port Level Event Summary (Channel Statistics) window opens as shown in the following image. The Channels pane contains a drop-down list on the left where you select a channel and a drop-down list on the right where you select a port. The Statistics pane contains a table with a summary of event statistics and two drop-down lists where you can select a successful or failed event to view more information about that event. In the lower right of the window is a button to click to suspend or resume a channel and a home button to click to return to the iBSE Monitoring Settings window.



6. For more information about a successful event and its port, select an event based on its correlation ID from the Successful Events drop-down list.

   The Event Level Statistics (Message Statistics) window opens as shown in the following image. The Message Information pane contains a table of information pertaining to the event message. The Messages pane contains a table that shows the size of the event and reply messages.



   a. To view the XML event document in your Web browser, click *Event Message*.

   b. To return to the iBSE Monitoring Settings window, click *home*.

Managing and Monitoring Services and Events Using the JCA Test Tool

The JCA Test Tool, which is also known as the JCA Installation Verification Program (IVP), provides a console to manage and monitor services and events currently in use. The console also displays resource usage and invocation statistics. These indicators can help you adjust your environment for optimum efficiency.

Procedure: How to Manage and Monitor Services Using the JCA Test Tool

To manage and monitor services using the JCA Test Tool:

1. Open a Web browser to:

   `http://hostname:port/iwjcaivp`

   where:

   hostname

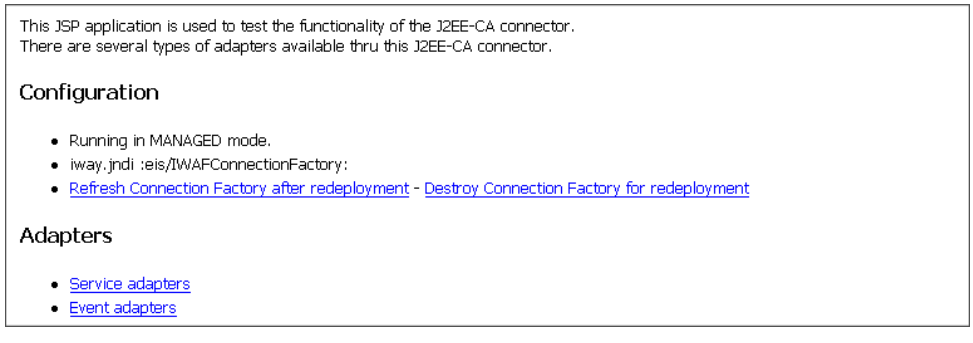
   Is the name of the machine where your application server is running.

   port

   Is the port for the domain you are using. The port for the default domain is 7001, for example:

   `http://localhost:7001/iwjcaivp`

   The following image shows the JCA Test Tool pane that opens. The pane contains a description of the function of the tool and configuration information, including options to change your connection settings. It also provides options for viewing service or event adapters.



   The JCA Test Tool runs in managed mode by default.

2. Perform the following steps to monitor the latest service adapter configuration.

   Note: You must perform these steps for every new adapter target that is created using a JCA implementation of Application Explorer. In addition, you also must perform these steps for every new JCA configuration that is created using Application Explorer.

   a. Click *Destroy Connection Factory for redeployment*.

   b. Redeploy the JCA connector.

   c. In the JCA Test Tool, click *Refresh Connection Factory after redeployment*.

3. Click *Service adapters*.

4. Select a service adapter to monitor.

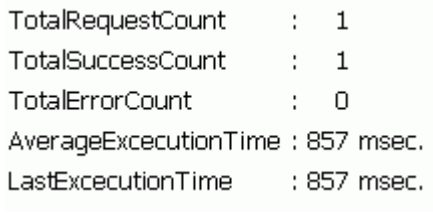
   a. Click the desired target for your service adapter.

   b. In the Request area, enter a user name and password.

   c. In the Input Doc area, enter a request document created from the request schema for your service.

5. Click *Send*.

   The following image shows the updated statistics that appear for your service if the request is successful. The statistics include the total number of requests, successes, and errors and the average and last execution time in milliseconds.



Procedure: How to Manage and Monitor Events Using the JCA Test Tool

To manage and monitor events using the JCA Test Tool:

1. Open a Web browser to:

   `http://hostname:port/iwjcaivp`

   where:

   hostname

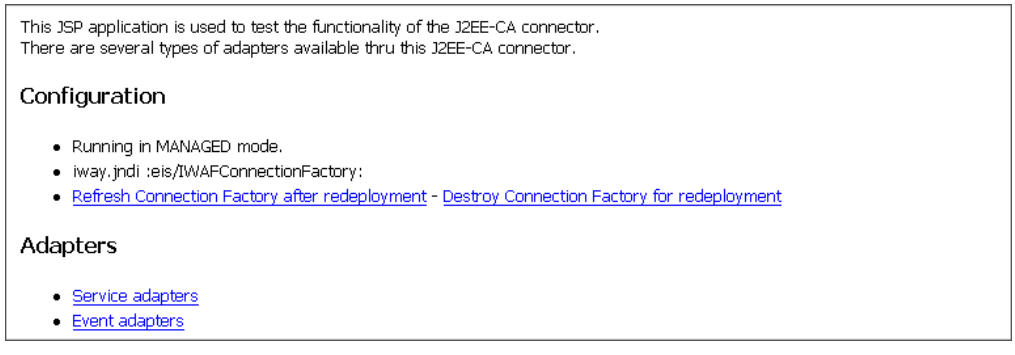
   Is the name of the machine where your application server is running.

   port

   Is the port for the domain you are using. The port for the default domain is 7001, for example:

   `http://localhost:7001/iwjcaivp`

   The JCA Test Tool pane opens as shown in the following image. The pane contains a description of the function of the tool and configuration information, including options to change your connection settings. It also provides options for viewing service or event adapters.



   The JCA Test Tool runs in managed mode by default.

2. To monitor the latest event adapter configuration, perform the following steps.

   Note: You must perform these steps for every new adapter target that is created using a JCA implementation of Application Explorer. In addition, you must also perform these steps for every new JCA configuration that is created using Application Explorer.

   a. Click *Destroy Connection Factory for redeployment*.

   b. Redeploy the JCA connector.

   c. In the JCA Test Tool, click *Refresh Connection Factory after redeployment*.

3. Click *Event adapters*.

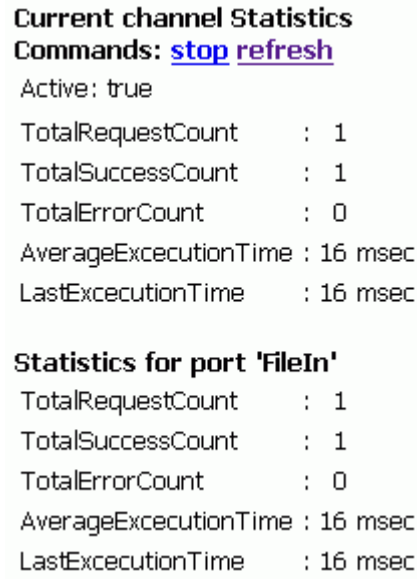
   The Event Adapters pane opens.

4. Select the event adapter to monitor.

5. Click the desired channel for your event adapter.

6. Click *start*.

   The following image shows the updated statistics for your channel and the port. The statistics include the total number of requests, successes, and errors and the average and last execution time in milliseconds. The upper right of the pane contains options to start or refresh the channel.



Setting Engine Log Levels

The following procedures describe how to set engine log levels. For more information, see the *BEA WebLogic ERP Adapter Installation and Configuration* manual.

Procedure: How to Enable Tracing for Servlet iBSE

To enable tracing for Servlet iBSE:

1. Open the Servlet iBSE configuration window at:

   `http://hostname:port/ibse/IBSEConfig`

   where:

   hostname

   Is the name of the machine where your application server is running.

   port

   Is the port for the domain you are using. The port for the default domain is 7001, for example:

   `http://localhost:7001/ibse/IBSEConfig`

2. In the System pane, from the Debug drop-down list, select the level of tracing.
3. Click *Save*.

The default location for the trace information on Windows is:

`C:\Program FIles\bea\ibse\ibselogs`

Procedure: How to Enable Tracing for JCA

To enable tracing for JCA:

1. Extract the ra.xml file from the iwafjca.rar file.
2. Open the extracted ra.xml file in a text editor.
3. Locate and change the following setting:

   LogLevel. This setting can be set to DEBUG, INFO, or ERROR.

   ```
   <context-param>
   <config-property>
       <config-property-name>LogLevel</config-property-name>
       <config-property-type>java.lang.String</config-property-type>
       <config-property-value></config-property-value>
   </config-property>
   ```

   For example:

   ```
   <config-property-value>DEBUG</config-property-value>
   ```

   Leave the remainder of the file unchanged.
4. Save the file and exit the editor.
5. Add the file back to the iwafjca.rar file.
6. Redeploy the connector.

A directory in the configuration directory contains the logs. For example, on Windows the default location is the following:

```
C:\Program Files\iway55\config\base\log
```

The log files are named jca_*.log.

You should also review the logs generated by your application server.

Configuring Connection Pool Sizes

The following topic describes how to configure connection pool sizes for the Connector for JCA.

Procedure: How to Configure Connection Pool Sizes

To configure connection pool sizes:

1. Extract the ra.xml file from the iwafjca.rar file.
2. Open the extracted ra.xml file in a text editor.
3. Locate and change the following setting:

   pool-params. The JCA Resource Connector has an initial capacity value of 0 by default and cannot be changed. The maximum capacity value is 10 by default and can be changed to a higher value.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE weblogic-connection-factory-dd (View Source for full
doctype...)>
- <weblogic-connection-factory-dd>
  <connection-factory-name>IWAFJCA</connection-factory-name>
  <jndi-name>eis/IWAFConnectionFactory</jndi-name>
  - <pool-params>
    <initial-capacity>0</initial-capacity>
    <max-capacity>10</max-capacity>
    <capacity-increment>1</capacity-increment>
    <shrinking-enabled>false</shrinking-enabled>
    <shrink-period-minutes>200</shrink-period-minutes>
  </pool-params>
  <security-principal-map />
</weblogic-connection-factory-dd>
```

4. Save the file and exit the editor.
5. Return the ra.xml file to the iwafjca.rar file.
6. Redeploy the connector.

Migrating Repositories

During design time, a repository is used to store metadata that is created when using Application Explorer to:

- Configure adapter connections.
- Browse EIS objects.
- Configure services.
- Configure listeners to listen for EIS events.

For more information on configuring repositories, see the *BEA WebLogic ERP Adapter Installation and Configuration* manual.

The information in the repository also is referenced at run time. For management purposes, you can migrate iBSE and JCA repositories to new destinations without affecting your existing configuration. For example, you can migrate a repository from a development environment to a production environment. The BEA WebLogic Server must be restarted to detect new repository changes.

For more information on migrating repositories, see the *iWay 5.5.011 for BEA WebLogic 9.1 Migration Guide*.

Exporting or Importing Targets

After you migrate your repository, you can export or import targets with their connection information and persistent data between repositories.

Procedure: How to Export a Target

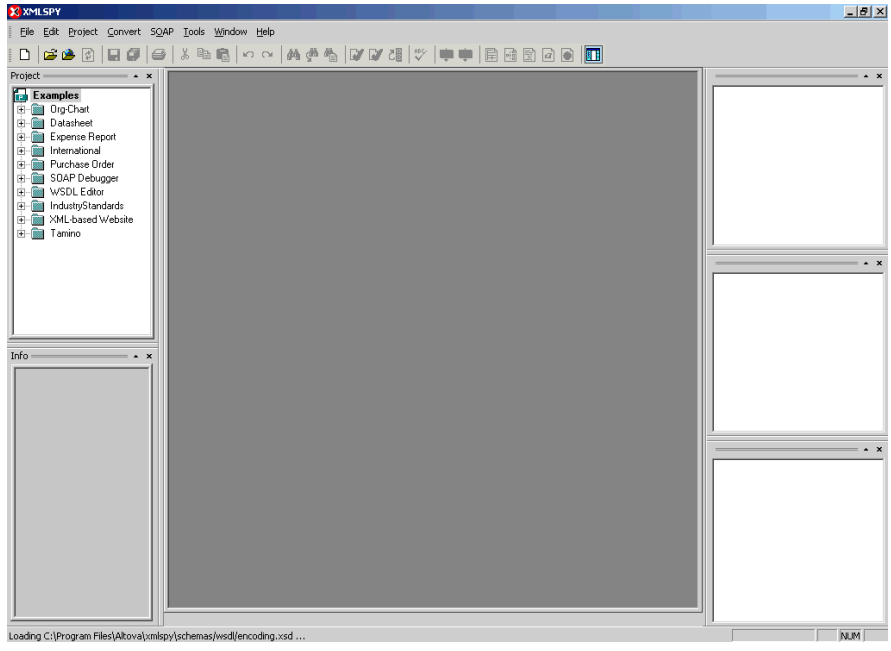
To export a target:

1. Copy the iBSE administrative services for Application Explorer URL, for example:

   `http://localhost:7777/ibse/IBSEServlet/admin/iwae.ibs?wsdl`

2. Open a third party XML editor, for example, XMLSPY.

The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE administrative services for Application Explorer URL.

5. Click *OK*.

   The soap operation name dialog box opens and lists the available control methods.

6. Select the *EXPORTTARGET(EXPORTTARGET parameters)* control method and click *OK*.

   A window opens that shows the structure of the SOAP envelope.

7. Locate the *Text view* icon in the tool bar.

8. To display the structure of the SOAP envelope as text, click the *Text view* icon.

   The <SOAP-ENV:Header> tag is not required and can be deleted from the SOAP envelope.

9. Locate the following section:

```
<m:EXPORTTARGET
xmlns:m="urn:schemas-iwaysoftware-com:dec2002:iwse:af">
<m:target>String</m:target>
<m:name>String</m:name>
</m:EXPORTTARGET>
```

a. For the <m:target> tag, replace the String placeholder with the EIS target system name as it appears in Application Explorer and verify whether this value is case sensitive.

b. For the <m:name> tag, replace the String placeholder with the name of the target you want to export.

10. From the SOAP menu, select *Send request to server*.

A response is returned that contains the <m: exporttime> and <m: contents> elements. You must use these elements when importing your target.

Procedure: How to Import a Target

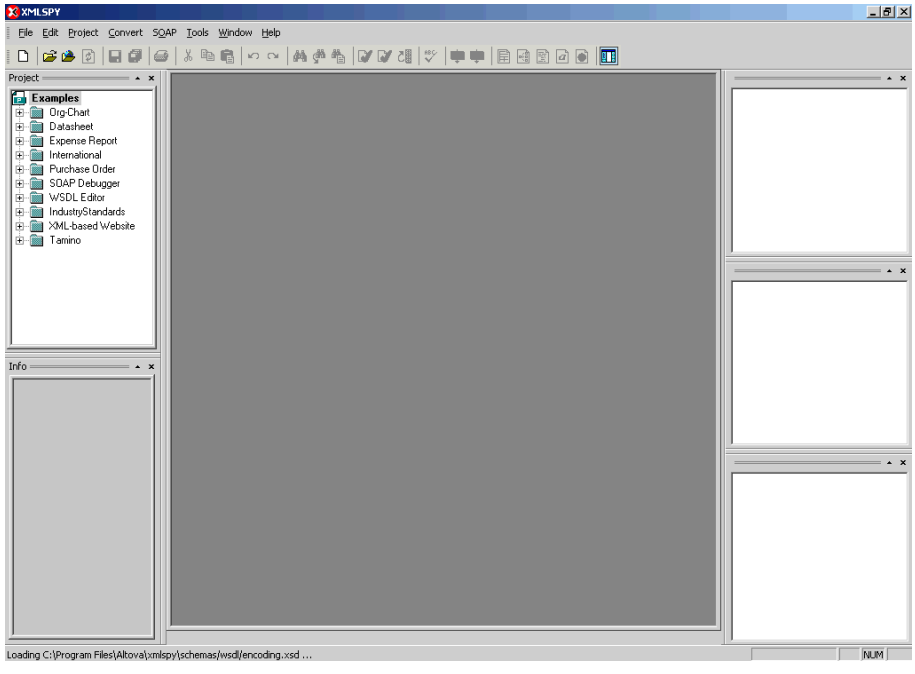
To import a target:

1. Copy the iBSE administrative services for Application Explorer URL, for example:

   `http://localhost:7777/ibse/IBSEServlet/admin/iwae.ibs?wsdl`

2. Open a third party XML editor, for example, XMLSPY.

   The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE administrative services for Application Explorer URL and click *OK*.

   The soap operation name dialog box opens and lists the available control methods.

5. Select the *IMPORTTARGET(IMPORTTARGET parameters)* control method and click *OK*.

   A window opens, which shows the structure of the SOAP envelope.

6. Locate the *Text view* icon in the tool bar.
7. To display the structure of the SOAP envelope as text, click the *Text view* icon.

   The <SOAP-ENV:Header> tag is not required and can be deleted from the SOAP envelope.
8. Locate the following section:

```
<m:IMPORTTARGET
xmlns:m="urn:schemas-iwaysoftware-com:dec2002:iwse:af">
<m:targetinstance>
<m:target>String</m:target>
<m:name>String</m:name>
<m:description>String</m:description>
<m:repositoryid>String</m:repositoryid>
<m:exporttime>2001-12-17T09:30:47-05:00</m:exporttime>
<m:contents>R0lGODlhcgGSALMAAAQCAEMmCZtuMFQxDS8b</m:contents>
</m:targetinstance>
</m:IMPORTTARGET>
```

   a. For the <m:target> tag, replace the String placeholder with the EIS target system name.

   b. For the <m:name> tag, replace the String placeholder with the new name of the target you want to import.

   c. For the <m:description> tag, replace the String placeholder with a description of the target.

   d. For the <m:repositoryid> tag, copy and paste the contents of the <m:repositoryid> tag that was returned when you exported your target.

   e. For the <m: exporttime> tag, copy and paste the contents of the <m: exporttime> tag that was returned when you exported your target.

   f. For the <m: contents> tag, copy and paste the contents of the <m: contents> tag that was returned when you exported your target.
9. From the SOAP menu, select *Send request to server.*

Retrieving or Updating Web Service Method Connection Information

After you migrate your repository, you can retrieve or update connection information for your Web service methods.

Procedure: How to Retrieve Web Service Method Connection Information

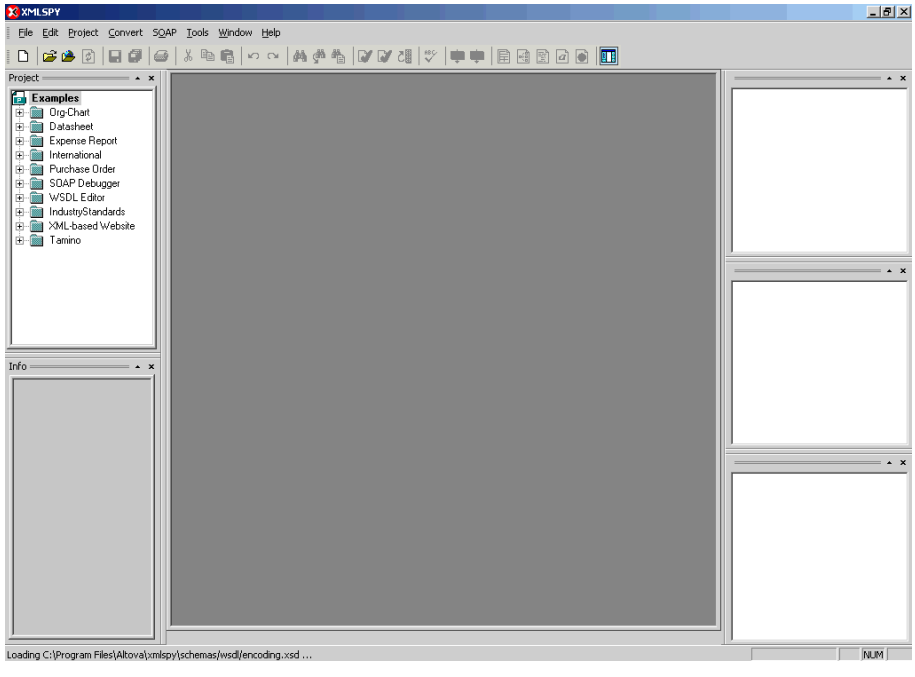
To retrieve Web service method connection information:

1. Copy the iBSE configuration service URL, for example:

   `http://localhost:7777/ibse/IBSEServlet/admin/iwconfig.ibs?wsdl`

2. Open a third party XML editor, for example, XMLSPY.

   The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE configuration service URL, and click *OK*.

   The soap operation name dialog box opens and lists the available control methods.

5. Select the *GETMTHCONNECTION(GETMTHCONNECTION parameters)* control method and click *OK*.

   A window opens, which shows the structure of the SOAP envelope.

6. Locate the *Text view* icon in the tool bar.

7. To display the structure of the SOAP envelope as text, click the *Text view* icon.

   The <SOAP-ENV:Header> tag is not required and can be deleted from the SOAP envelope.

8. Locate the following section:

```
<m:GETMTHCONNECTION
xmlns:m="urn:schemas-iwaysoftware-com:jul2003:ibse:config">
<m:servicename>String</m:servicename>
<m:methodname>String</m:methodname>
</m:GETMTHCONNECTION>
```

   a. For the <m:servicename> tag, replace the String placeholder with the name of the Web service.

   b. For the <m:methodname> tag, replace the String placeholder with name of the Web service method.

9. From the SOAP menu, select *Send request to server*.

   A response is returned that contains the <m: descriptor> element. You must use this element when updating your Web service method.

Procedure: How to Update Web Service Method Connection Information

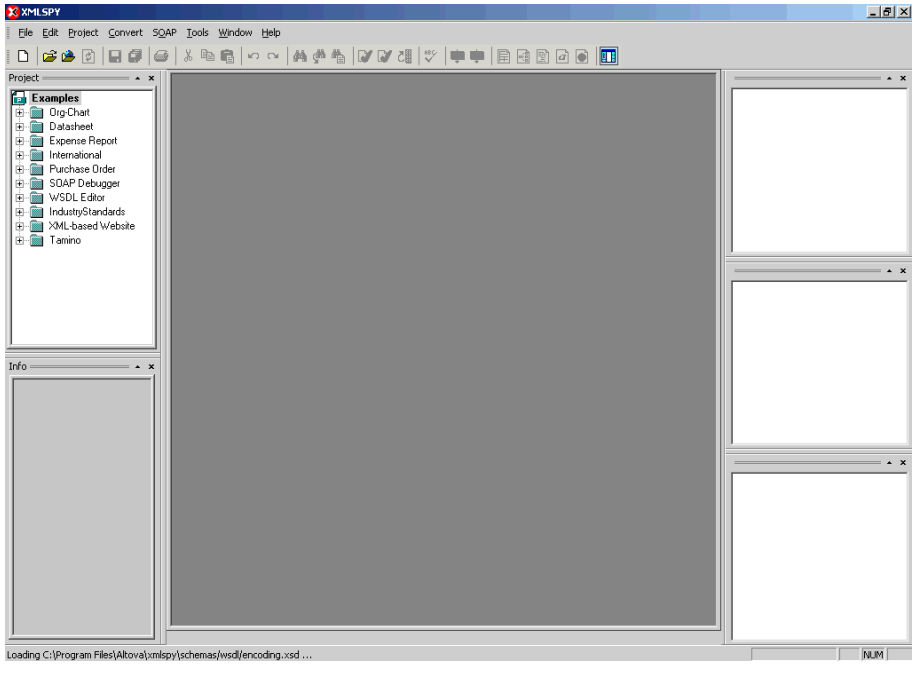
To update Web service method connection information:

1. Copy the iBSE configuration service URL, for example:

   `http://localhost:7777/ibse/IBSEServlet/admin/iwconfig.ibs?wsdl`

2. Open a third party XML editor, for example, XMLSPY.

   The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE configuration service URL, and click *OK*.

   The soap operation name dialog box opens and lists the available control methods.

5. Select the *SETMTHCONNECTION(SETMTHCONNECTION parameters)* control method and click *OK*.

   A window opens that shows the structure of the SOAP envelope.

6. Locate the *Text view* icon in the tool bar.
7. To display the structure of the SOAP envelope as text, click the *Text view* icon.

   The <SOAP-ENV:Header> tag is not required and can be deleted from the SOAP envelope.
8. Locate the following section:

```
<m:SETMTHCONNECTION
xmlns:m="urn:schemas-iwaysoftware-com:jul2003:ibse:config">
<m:servicename>String</m:servicename>
<m:methodname>String</m:methodname>
<m:descriptor format="" channel="">
     <m:option title="">
       <m:group title="">
          <m:param/>
       </m:group>
     </m:option>
</m:descriptor>
</m:SETMTHCONNECTION>
```

   a. For the <m:servicename> tag, replace the String placeholder with the name of the Web service.

   b. For the <m:methodname> tag, replace the String placeholder with the name of the Web service method.

   c. For the <m: descriptor> tag, copy and paste the contents of the <m: descriptor> tag that was returned when you retrieved Web Service method connection information.
9. Modify the contents of the <m: descriptor> tag to change the existing Web Service method connection information.
10. From the SOAP menu, select *Send request to server.*

Starting or Stopping a Channel Programmatically

The following topic describes how to start or stop a channel programmatically.

Procedure: How to Start a Channel Programmatically

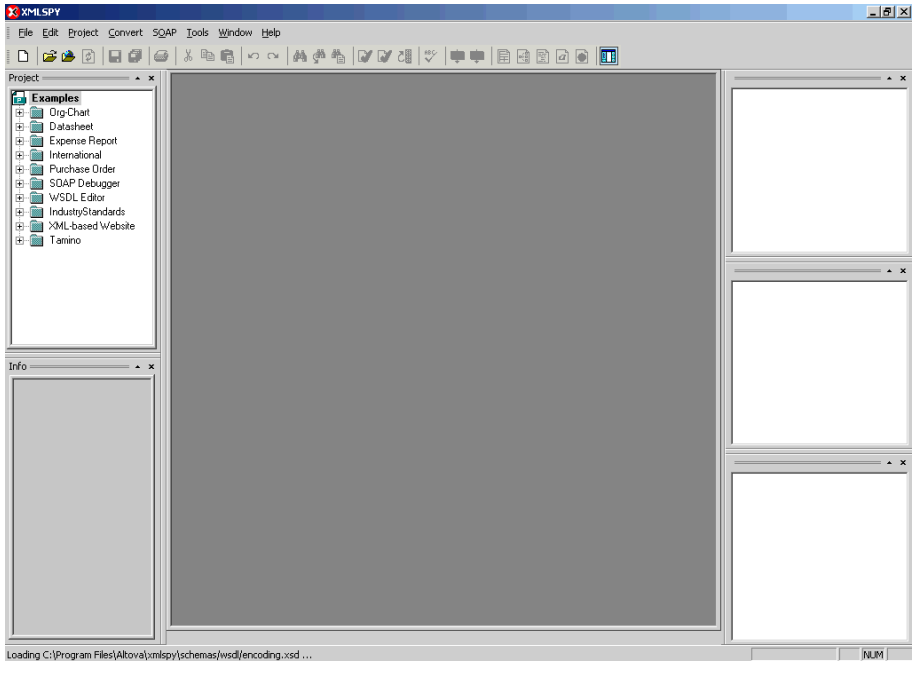
To start a channel programmatically:

1. Copy the iBSE control event URL, for example:

   `http://localhost:7777/ibse/IBSEServlet/admin/iwevent.ibs?wsdl`

2. Open a third party XML editor, for example, XMLSPY.

   The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.

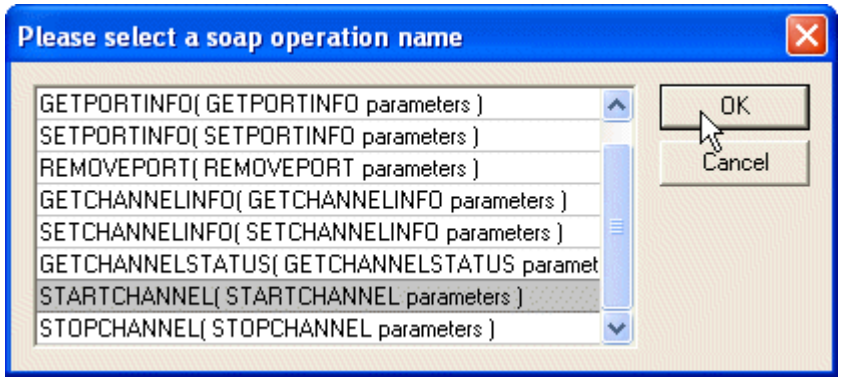


3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE control event URL, and click *OK*.

   The following image shows the soap operation name dialog box that opens with a list of available control methods.



5. Select the *STARTCHANNEL(STARTCHANNEL parameters)* control method and click *OK*.

   A window opens, which shows the structure of the SOAP envelope.

6. Locate the *Text view* icon in the tool bar.

7. To display the structure of the SOAP envelope as text, click the *Text view* icon.

   The <SOAP-ENV:Header> tag is not required and can be deleted from the SOAP envelope.

8. Locate the following section:

```
<SOAP-ENV:Body>
  <m:STARTCHANNEL
     xmlns:m="urn:schemas-iwaysoftware-com:dec2002:iwse:event">
           <m:channel>String</m:channel>
  </m:STARTCHANNEL>
</SOAP-ENV:Body>
```

9. For the <m:channel> tag, replace the String placeholder with the name of the channel you want to start.

10. From the SOAP menu, select *Send request to server.*

Procedure: How to Stop a Channel Programmatically

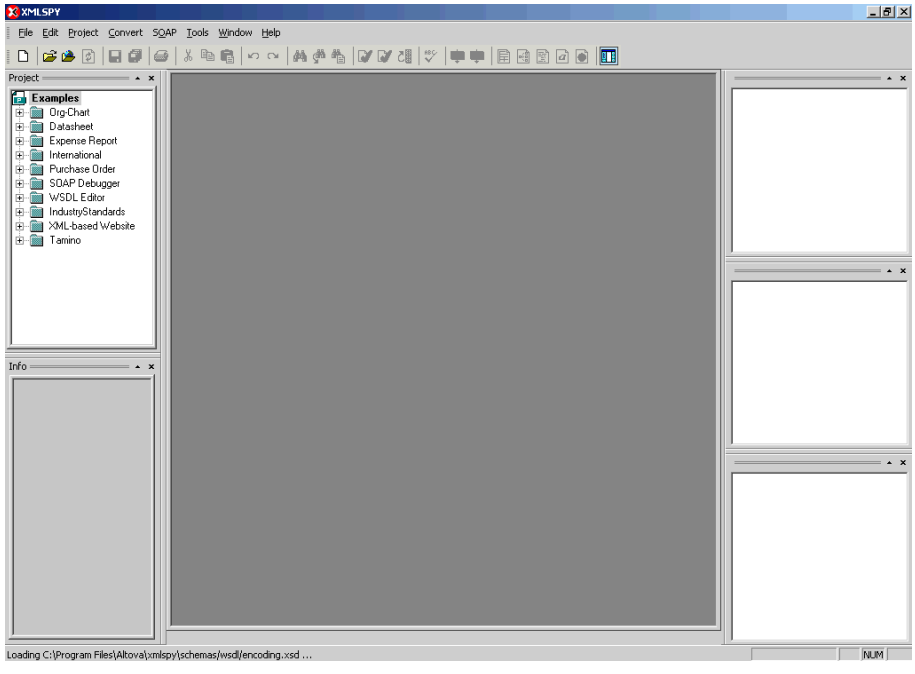
To stop a channel programmatically:

1. Copy the iBSE control event URL, for example:

   `http://localhost:7777/ibse/IBSEServlet/admin/iwevent.ibs?wsdl`

2. Open a third party XML editor, for example, XMLSPY.

   The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.

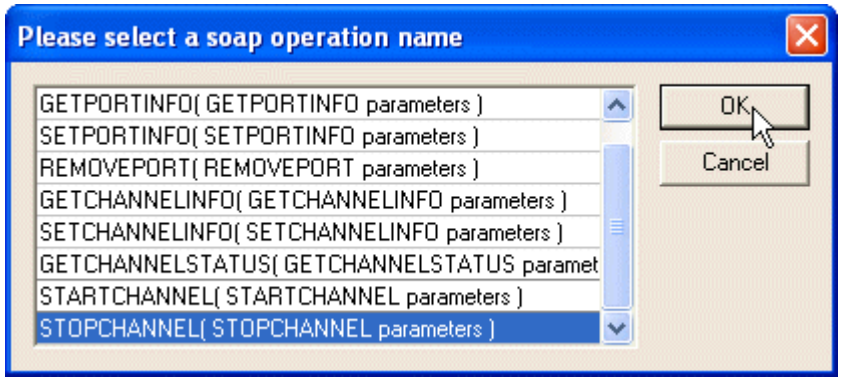


3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE control event URL, and click *OK*.

   The following image shows the soap operation name dialog box that opens with a list of available control methods.



5. Select the *STOPCHANNEL(STOPCHANNEL parameters)* control method and click *OK*.

   A window opens, which shows the structure of the SOAP envelope.

6. Locate the *Text view* icon in the tool bar.

7. To display the structure of the SOAP envelope as text, click the *Text view* icon.

   The <SOAP-ENV:Header> tag is not required and can be deleted from the SOAP envelope.

8. Locate the following section:

```
<SOAP-ENV:Body>
  <m:STOPCHANNEL
     xmlns:m="urn:schemas-iwaysoftware-com:dec2002:iwse:event">
           <m:channel>String</m:channel>
  </m:STOPCHANNEL>
</SOAP-ENV:Body>
```

9. For the <m:channel> tag, replace the String placeholder with the name of the channel you want to stop.

10. From the SOAP menu, select *Send request to Server.*

CHAPTER 5

Troubleshooting and Error Messages

Topics:

- Troubleshooting
- Error Messages in Application Explorer
- Error Messages in Siebel
- Error Messages in JCA
- Error Messages in iBSE
- Updating a Siebel Field

The following topics explain the limitations and workarounds when connecting to Siebel.

The adapter-specific errors listed in this section can arise whether you are using the adapter with a JCA or with an iBSE configuration.

Troubleshooting

This topic provides troubleshooting information, separated into the following categories:

- Application Explorer
- Siebel
- JCA
- iBSE

Note: Log file information that can be relevant in troubleshooting can be found in the following locations:

- The JCA trace information can be found under the

  C:\ProgramFiles\iWay55\config\base\log directory.
- iBSE trace information can be found under the

  C:\Program Files\iWay55\bea\ibse\ibselogs directory.
- The log file for Application Explorer can be found under the

  C:\Program File\iWay55\tools\iwae\bin directory.

Important: The BEA WebLogic Adapter for Siebel does not support Siebel Business Services that have Business Service method arguments of data type "Hierarchy." If you create Web services or create services using the JCA Connector for Siebel Business Services Hierarchy data types, the following error message appears when the Web service is invoked at run time:

```
'java.lang.Exception: Storage Type [Hierarchy] is not supported yet'
```

Error Messages in Application Explorer

The following table lists errors and solutions when using Application Explorer with the adapter.

Error	Solution
Siebel does not appear in the Application Explorer Adapter node list.	Ensure that the Siebel JAR files supplied with your Siebel distribution media were placed in the iway_home/lib directory. For example, for Siebel 7.03 environments, the SiebelJI_Common.jar and SiebelJI_enu.jar files should be placed in the iway_home/lib directory.

Error	Solution
Target Type drop-down list contains only Java Data Bean Connection, and COM connection type is required.	Ensure that the Siebel thin client is installed correctly on the machine hosting Application Explorer so that the appropriate COM environment is available.
An error message that includes the name of the Siebel Gateway server appears when you try to connect to a Siebel target, for example, `Problem activating adapter (ariba0x). Check logs for more information.`	Ensure that the name of the Siebel Gateway server is correctly defined for the target to which you want to connect.
When trying to connect to a Siebel target, you receive the following error: `Problem activating adapter. (You have entered an invalid set of logon parameters. Please type in your logon parameters again.). Check logs for more information.`	Ensure that the User ID and password parameter values to connect to your Siebel system are correct.
When trying to connect to a Siebel target, you receive the following error: `Problem activating adapter. (Couldn't get nameserver connection). Check logs for more information.`	Check the network connectivity to the Siebel environment. Correct the networking problem and retry the connection. Also, if Siebel was started recently, it might not be fully functional yet. If so, wait until Siebel starts completely. **Note:** If Siebel Server was restarted after servlet iBSE connected, then servlet iBSE also must be recycled. This is due to a known Siebel issue. For more information, see Siebel Alert 984.
When trying to connect to a Siebel target, you receive the following error: `Problem activating adapter. (NSReadKey request failed (no error information)...). Check logs for more information.`	Ensure that the values defined for Siebel Server, Enterprise Name, and Object Manager for the target to which you want to connect are correct and retry the connection.

Error	Solution
When trying to connect to a Siebel target, you receive the following error: `Problem activating adapter. (Error loading translatable messages: com.siebel.locale.enux.messages .SSAMessages_enux). Check logs for more information`	Ensure that the value of the Language parameter on the Advanced tab is defined correctly for the target you are using to connect to your Siebel system (for example, enu for English).
A successful connection is made to the Siebel environment, but no values are available in Business Object, Business Service, and Integration Object nodes in the Application Explorer tree.	The Repository Name specified on the Advanced tab in the Siebel target configuration is either void or empty of any components in the targeted Siebel environment, or the Repository Name is not valid for the targeted Siebel environment. Verify that the Repository Name is valid and contains components for interrogation and then, reconnect.

Error Messages in Siebel

The following table lists errors that occur when using the adapter with either an iBSE or JCA repository project and provides a solution to each error.

Error	Solution
A successful connection is made to Siebel environment, but no values are available in the Business Object, Business Service, and Integration Object nodes in the Application Explorer tree.	The Repository Name specified on the Advanced tab in the Siebel Target configuration is either void or empty of any components in the targeted Siebel environment, or the Repository Name is not valid for the targeted Siebel environment. Verify that the Repository Name is valid and contains components for interrogation and then, reconnect.
When executing a request, the following error message appears: `AdapterException: Unsupported Action: {0} Tquery`	Verify that the method is available for the specific request by verifying schema.

Error	Solution
When executing a request, the following error message appears: `AdapterException: Field 'NFame' does not exist in definition for business component 'Account'. Please ask your systems administrator to check your application configuration.`	Ensure that the field names are valid within the request document by referring to the schema for that specific object and then, resubmit the request.
When connecting to releases prior to Siebel 7.7 using the Java Data Bean Interface, you cannot reconnect after initial connection loss. This might occur when Application Explorer experiences a brief loss of network connection or if the Siebel Server or Gateway Service is restarted while Application Explorer is logged into the Siebel application.	Restart your application server and Application Explorer in order to log in successfully to the Siebel application. This is a known Siebel API issue. For more information, see Siebel Alert 984.

Error	Solution
The BEA WebLogic Adapter for Siebel does not interact with Integration Objects in a Siebel workflow using Web services or JCA in Application Explorer. This is a result of a limitation in Siebel software that does not handle the namespaces in the incoming XML document for the Integration Object. An error similar to the following might appear: `Cannot convert XML Hierarchy to Integration Object Hierarchy.--Field with XML tag 'xmlns:ns' and XML Style of 'Attribute' is not found in the definition of EAI Integration Component 'Account'--Error invoking service 'EAI XML Converter', method 'XMLDocToIntObjHier' at step 'XML to Property Set'.`	The preferred workaround is to add to the Integration Object an Integration Object User Property with the name *Ignore Undefined XML Tags* and a value of *Y*. Alternatively, a less preferred work around is to invoke a Siebel workflow that will run the Web service. Pass the XML document that represents the input via a protocol, for example, HTTP or MQ, that the workflow is listening on. You can build the emission of the XML document using that protocol as a Web service in Application Explorer.

Error Messages in JCA

The following table describes an error that occurs when connecting to a JCA configuration and provides a solution to the error.

Error	Solution
In Application Explorer, the following error message appears when you attempt to connect to a JCA configuration: `Could not initialize JCA`	In the Details tab in the right pane, ensure that the directory specified in the Home field points to the correct directory, for example, *iway_home*`/lib`

Error Messages in iBSE

This topic discusses the different types of errors that can occur when processing Integration Business Services through the Integration Business Services Engine (iBSE).

General Error Handling

The Integration Business Services Engine (iBSE) serves as both a SOAP gateway into the adapter framework and as the engine for some of the adapters. In both design time and execution time, various conditions can cause errors in iBSE when Web services that use adapters are running. Some of these conditions and resulting errors are exposed the same way, regardless of the specific adapter; others are exposed differently, based on the adapter being used. This topic explains what you can expect when you encounter some of the more common error conditions on an adapter-specific basis.

Usually, the SOAP gateway (agent) inside iBSE passes a SOAP request message to the adapter required for the Web service. If an error occurs, the way it is exposed depends on the adapter and the API or interfaces that the adapter uses. A few scenarios cause the SOAP gateway to generate a SOAP fault. In general, when the SOAP agent inside iBSE receives an invalid SOAP request, a SOAP fault element is generated in the SOAP response. The SOAP fault element contains fault string and fault code elements. The fault code contains a description of the SOAP agent error.

The following SOAP response document results when iBSE receives an invalid SOAP request:

```
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
 <SOAP-ENV:Body>
      <SOAP-ENV:Fault>
         <faultcode>SOAP-ENV:Client</faultcode>
         <faultstring>Parameter node is missing</faultstring>
      </SOAP-ENV:Fault>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

In the previous example, iBSE did not receive an element in the SOAP request message that is mandatory for the WSDL for this Web service.

Adapter-Specific Error Handling

When an adapter raises an exception during execution, the SOAP agent in iBSE produces a SOAP fault element in the generated SOAP response. The SOAP fault element contains fault code and fault string elements. The fault string contains the native error description from the adapter target system. Because adapters use the target system interfaces and APIs, whether an exception is raised depends on how the target systems interface or API treats the error condition. If a SOAP request message is passed to an adapter by the SOAP agent in iBSE, and that request is invalid based on the WSDL for that service, the adapter may raise an exception yielding a SOAP fault.

Although it is almost impossible to anticipate every error condition that an adapter may encounter, the following describes how adapters handle common error conditions and how they are then exposed to the Web service consumer application.

Example: BEA WebLogic Adapter for Siebel Invalid SOAP Request

When the adapter receives a SOAP request message that does not conform to the WSDL for the Web service being executed, the following SOAP response is generated:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
  soap/envelope/">
   <SOAP-ENV:Body>
     <SOAP-ENV:Fault>
       <faultcode>SOAP-ENV:Server</faultcode>
       <faultstring>XD[FAIL] Parse failure (IS) 3:
         org.xml.sax.SAXParseException: Premature end of
         file.</faultstring>
     </SOAP-ENV:Fault>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Example: Failure to Connect to Siebel

When the adapter cannot connect to Siebel when executing a Web service, the following SOAP response is generated.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
 <SOAP-ENV:Body>
  <SOAP-ENV:Fault>
  <faultcode>SOAP-ENV:Server</faultcode>
  <faultstring><Exception> - major:4096 minor: -1 message:NSReadKey
request 11 was abandoned after 37846ms connection:12a due to Connection
shutdown request Connection reset by peer:JVM_recv in socket input stream
stream read DetailedMessage:Unknown</Exception></faultstring>
  </SOAP-ENV:Fault>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Invalid SOAP Request

When the adapter receives a SOAP request message that does not conform to the WSDL for the Web service being executed, the following SOAP response is generated.

```
<?xml version="1.0" encoding="ISO-8859-1"
 ?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
 <SOAP-ENV:Fault>
  <faultcode>SOAP-ENV:Server</faultcode>
  <faultstring>RPC server connection failed: Connection refused:
connect</faultstring>
 </SOAP-ENV:Fault>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Empty Result From a Request

Note: The condition for this adapter does not yield a SOAP fault.

When the adapter executes a SOAP request using input parameters passed that do not match records in the target system, the following SOAP response is generated.

```
<SOAP-ENV:Envelope xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/
"xmlns:xsd="http://www.w3.org/1999/XMLSchema">
   <SOAP-ENV:Body>
      <m:RunDBQueryResponse xmlns:m="urn:schemas-iwaysoftware-com:iwse"
        xmlns="urn:schemas-iwaysoftware-com:iwse"
        cid="2A3CB42703EB20203F91951B89F3C5AF">
        <RunDBQueryResult run="1" />
     </m:RunDBQueryResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Updating a Siebel Field

If you cannot update a Siebel field, you might be required to perform configuration changes on the Siebel system. The following procedure describes how to verify whether a Siebel field is activated for updates.

Procedure: How to Verify a Siebel Field is Activated for Updates

To verify whether a Siebel field is activated for updates:

1. Obtain the *siebel.srf* file from the Siebel server where you cannot update a field.

   The SRF file is usually located in the \tools\OBJECTS\ENU folder of the Siebel server install, for example:

   ```
   D:\sea752\siebsrvr\OBJECTS\ENU
   ```

2. Under the Siebel\tools directory, locate the *siebel.srf* file.

   The file is usually located under a similar folder, for example:

   ```
   D:\sea752\tools\OBJECTS\ENU
   ```

   a. Rename the siebel.srf file in the Siebel\tools directory, for example, to siebel.srf.original.

   b. Place a copy of the siebel.srf file from the Siebel server into the Siebel\tools directory and then, make a copy of it in the same directory and call it *siebeltest.srf*.

3. Start Siebel Tools and connect to the Siebel server.

4. Identify the Business Component for the Siebel field that is not being updated, for example, the Business Component Account for the Home Page field.

5. With the Account Business Component highlighted, click *Field* in the Object Explorer.

6. Identify the field which is not activated for updates or not subject to adapter update request, for example, Home Page or Email Address.
7. Check the properties for the Siebel field, specifically for the Force Active property.

   If the property is false, that is, it has no check box selected or it has the value FALSE (this depends on the Siebel Tools GUI configuration that you are using), then you must change it to true, as explained in the following steps.

   Note: This procedure refers to the check box as the method for making the property TRUE.

   a. If the Force Active property is false, select *Lock Project* from the Tools menu.

      Note the name of the project being locked. It will be specified in the Project property of the Business Component for the field being made Force Active.

   b. Click *Force Active* to set it to True.

      This places a check mark in the property field.

   c. After setting the Force Active property to True for the Account Business Component, select *Unlock Project* from the Tools menu.
8. Check the Multivalue Link property of the field.

   If there is no value in the Multivalue Link property, then proceed to the next numbered step.

   If there is a value in this property, it is a reference to another Business Component. This indicates that the field in question is linked to a field in the Business Component identified by this property.

   a. Ensure that the linked field in the Business Component specified by the Multivalue Link property of the first field has a Force Active property of True.

   b. For example, if the Email Address field has a Multivalue link property with a value of Business Address, you must go to the Business Address Business Component and ensure that the linked field in the Business Address Business Component also has a value of True in the Force Active field.

   c. Repeat Step 4 through Step 7, including substeps, for the field in the Business Component specified in the Multivalue link property.

      You must compile the project so that the changes take effect.
9. Select *Compile Projects* from the Tools menu.

   The Object Compiler appears.

   a. Select the *Selected projects* option button.

b. Select the project(s) to compile. To select multiple projects, hold down the *Ctrl* key as you select each project.

If the field had a Business Component defined in the Multivalue Link property, you also must select the project(s) for the additional Business Component(s) you updated.

c. Click *Browse* and navigate to the location of the copied siebel.srf file in the Siebel\tools directory.

In this procedure, siebeltest.srf is the copied file.

10. Click *Compile*.

Compiling could take several minutes.

11. Transfer the newly compiled siebeltest.srf file to the Siebel Server system and replace the siebel.srf file located under \siebsrvr\OBJECTS\ENU with this new one. You must:

a. First stop the Siebel Server.

b. Make a backup copy of the current SRF file.

c. Replace the SRF file with siebeltest.srf by placing the SRF file in the \siebsrvr\OBJECTS\ENU directory and renaming it to siebel.srf (the original file name).

12. Restart the Siebel server and wait until it starts up completely.

This could take 5-10 minutes.

a. Open Task Manager and wait to see that CPU usage stops hovering at 100% and returns to a more normal range for activity on your machine.

b. Retest.

You can now update the fields through the adapter or Siebel application.

APPENDIX A

Usage Considerations and Sample Files

Topics:

- Usage Considerations
- Sample Files

This section briefly describes usage considerations for Siebel. It also provides sample XML schemas for Siebel Business Components, Siebel Business Services, and Integration Objects.

Usage Considerations

For Business Components, the BEA WebLogic Adapter for Siebel enables Insert, Update, Delete, and Query. It also enables a method called QueryWithView. The View modes are a visibility feature provided by Siebel.

By using QueryWithView, you can specify a Siebel View mode as a parameter. The API parameters allow different presentations of data depending on the Siebel environment that you configured.

You can use Query except when you want to enable a user to retrieve records based on different view modes. In this case, use QueryWithView. For more information on QueryWithView mode or Siebel "Visibility" concepts, see your Siebel Administrator.

The following levels are available:

- Sales Rep View
- Manager View
- Personal View
- All View
- Organization View
- Group View
- Catalog View
- SubOrganization View

Sample Files

The following topics includes samples of schemas for Account Business Component.

Account Business Component

The following examples include samples of:

- QueryWithView request schema
- QueryWithView response schema

Example: Account QueryWithView Request Schema

```
  <?xml version="1.0" encoding="UTF-8" ?>
  <!--  Generated by the iBSE 2004-04-09T19:36:01Z
  -->
  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:iwaysoftware:adapter:siebel:bo:oct2003"
xmlns:z="urn:iwaysoftware:adapter:siebel:bo:oct2003" elementFormDefault="qualified">
   <xsd:element name="Siebel">
  <xsd:complexType>
  <xsd:sequence>
  <xsd:element name="select" type="z:record" />
  <xsd:element name="field" minOccurs="0" maxOccurs="unbounded">
  <xsd:simpleType>
  <xsd:restriction base="xsd:string">
  <xsd:enumeration value="Account Competitors" />
  <xsd:enumeration value="Account Condition" />
  <xsd:enumeration value="Account Markets" />
  <xsd:enumeration value="Account Organization Integration Id" />
  <xsd:enumeration value="Account Products" />
  <xsd:enumeration value="Account Role" />
  <xsd:enumeration value="Account Status" />
  <xsd:enumeration value="Account Trend" />
  <xsd:enumeration value="Address Active Status" />
  <xsd:enumeration value="Address Id" />
  <xsd:enumeration value="Address Integration Id" />
  <xsd:enumeration value="Agreement End Date" />
  <xsd:enumeration value="Agreement Name" />
  <xsd:enumeration value="Agreement Start Date" />
  <xsd:enumeration value="Agreement Status" />
  <xsd:enumeration value="Algorithm Type" />
  <xsd:enumeration value="Alias" />
  <xsd:enumeration value="Annual Revenue" />
  <xsd:enumeration value="Assignment Area Code" />
  <xsd:enumeration value="Assignment Country Code" />
  <xsd:enumeration value="Assignment Denorm Flag" />
  <xsd:enumeration value="Assignment Excluded" />
  <xsd:enumeration value="Assignment Manual Flag" />
  <xsd:enumeration value="Assignment System Flag" />
  <xsd:enumeration value="Assignment Type" />
  <xsd:enumeration value="BO Export Status" />
  <xsd:enumeration value="Back Office Distribution Channel" />
  <xsd:enumeration value="Back Office Order Query End Dt" />
  <xsd:enumeration value="Back Office Order Query Start Dt" />
  <xsd:enumeration value="Back Office Sales Area" />
  <xsd:enumeration value="Back Office Sales Area Division Code" />
  <xsd:enumeration value="Back Office Sales Organization" />
  <xsd:enumeration value="Bill Address Flag" />
  <xsd:enumeration value="Bill To City" />
```

```
<xsd:enumeration value="Bill To Country" />
<xsd:enumeration value="Bill To First Name" />
<xsd:enumeration value="Bill To Id" />
<xsd:enumeration value="Bill To Job Title" />
<xsd:enumeration value="Bill To Last Name" />
<xsd:enumeration value="Bill To Postal Code" />
<xsd:enumeration value="Bill To State" />
<xsd:enumeration value="Bill To Street Address" />
<xsd:enumeration value="Block Credit Flag" />
<xsd:enumeration value="Business Profile" />
<xsd:enumeration value="CSN" />
<xsd:enumeration value="Category" />
<xsd:enumeration value="Category Value" />
<xsd:enumeration value="City" />
<xsd:enumeration value="City State" />
<xsd:enumeration value="Competitor" />
<xsd:enumeration value="Country" />
<xsd:enumeration value="County" />
<xsd:enumeration value="Credit Control Area Code" />
<xsd:enumeration value="Credit Currency Code" />
<xsd:enumeration value="Credit Limit Amount" />
<xsd:enumeration value="Credit Profile Id" />
<xsd:enumeration value="Culture" />
<xsd:enumeration value="Currency Code" />
<xsd:enumeration value="Current Volume" />
<xsd:enumeration value="Current Volume Currency Code" />
<xsd:enumeration value="Current Volume Exchange Date" />
<xsd:enumeration value="Customer Account Group" />
<xsd:enumeration value="DNBReport" />
<xsd:enumeration value="DUNS Intcode" />
<xsd:enumeration value="DUNS Number" />
<xsd:enumeration value="Date Formed" />
<xsd:enumeration value="DeDup Key Modification Date" />
<xsd:enumeration value="DeDup Key Update" />
<xsd:enumeration value="DeDup Keys" />
<xsd:enumeration value="DeDup Last Match Date" />
<xsd:enumeration value="DeDup Token" />
<xsd:enumeration value="Deduplication Match Score" />
<xsd:enumeration value="Deduplication Object Id" />
<xsd:enumeration value="Description" />
<xsd:enumeration value="Disable DataCleansing" />
<xsd:enumeration value="Disable Mailings" />
<xsd:enumeration value="Division" />
<xsd:enumeration value="Domestic Ultimate DUNS" />
<xsd:enumeration value="Dummy" />
<xsd:enumeration value="EAI Sync Date" />
<xsd:enumeration value="EAI Sync Error Text" />
<xsd:enumeration value="EAI Sync Status Code" />
```

```
<xsd:enumeration value="Email Address" />
<xsd:enumeration value="Employee Here" />
<xsd:enumeration value="Employees" />
<xsd:enumeration value="Expertise" />
<xsd:enumeration value="Explorer Label" />
<xsd:enumeration value="Fax Number" />
<xsd:enumeration value="First Name" />
<xsd:enumeration value="Fiscal Year End" />
<xsd:enumeration value="Freight Terms" />
<xsd:enumeration value="Freight Terms Info" />
<xsd:enumeration value="Full Address" />
<xsd:enumeration value="Full Name" />
<xsd:enumeration value="GSA Flag" />
<xsd:enumeration value="Global Ultimate DUNS" />
<xsd:enumeration value="Goals" />
<xsd:enumeration value="Group Type Code" />
<xsd:enumeration value="Home Page" />
<xsd:enumeration value="Industry" />
<xsd:enumeration value="Industry Condition" />
<xsd:enumeration value="Industry Trend" />
<xsd:enumeration value="Integration Id" />
<xsd:enumeration value="Internal Org Flag" />
<xsd:enumeration value="Joined Synonym" />
<xsd:enumeration value="Key Competitors" />
<xsd:enumeration value="Language Code" />
<xsd:enumeration value="Last Clnse Date" />
<xsd:enumeration value="Last Manager Review Date" />
<xsd:enumeration value="Last Name" />
<xsd:enumeration value="Last Review Manager Id" />
<xsd:enumeration value="Last Update - SDQ" />
<xsd:enumeration value="Line of Business" />
<xsd:enumeration value="Location" />
<xsd:enumeration value="Location Level" />
<xsd:enumeration value="Main Address Flag" />
<xsd:enumeration value="Main Fax Number" />
<xsd:enumeration value="Main Phone Number" />
<xsd:enumeration value="Managers Review" />
<xsd:enumeration value="Marketing" />
<xsd:enumeration value="Merge Sequence Number" />
<xsd:enumeration value="Mission" />
<xsd:enumeration value="Name" />
<xsd:enumeration value="Name and Location" />
<xsd:enumeration value="Not Manager Flag" />
<xsd:enumeration value="Notes" />
<xsd:enumeration value="Objectives" />
<xsd:enumeration value="Organization" />
<xsd:enumeration value="Organization Id" />
<xsd:enumeration value="Organization Integration Id" />
```

```
<xsd:enumeration value="Our Position" />
<xsd:enumeration value="Outline Number" />
<xsd:enumeration value="PO Approved Flag" />
<xsd:enumeration value="PO Auto Approval Currency Code" />
<xsd:enumeration value="PO Auto Approval Date" />
<xsd:enumeration value="PO Auto Approval Limit" />
<xsd:enumeration value="Parent Account Division" />
<xsd:enumeration value="Parent Account Id" />
<xsd:enumeration value="Parent Account Integration Id" />
<xsd:enumeration value="Parent Account Location" />
<xsd:enumeration value="Parent Account Location Level" />
<xsd:enumeration value="Parent Account Name" />
<xsd:enumeration value="Parent Account Region" />
<xsd:enumeration value="Parent HQ DUNS" />
<xsd:enumeration value="Partner Flag" />
<xsd:enumeration value="Partners" />
<xsd:enumeration value="Party Name" />
<xsd:enumeration value="Party Type Code" />
<xsd:enumeration value="Party UId" />
<xsd:enumeration value="Philosophy" />
<xsd:enumeration value="Phone Number" />
<xsd:enumeration value="Position" />
<xsd:enumeration value="Position Id" />
<xsd:enumeration value="Position Integration Id" />
<xsd:enumeration value="Position Territory" />
<xsd:enumeration value="Postal Code" />
<xsd:enumeration value="Price List" />
<xsd:enumeration value="Price List End Date" />
<xsd:enumeration value="Price List Id" />
<xsd:enumeration value="Price List Integration Id" />
<xsd:enumeration value="Price List Start Date" />
<xsd:enumeration value="Primary Account City" />
<xsd:enumeration value="Primary Account Country" />
<xsd:enumeration value="Primary Account Postal Code" />
<xsd:enumeration value="Primary Account State" />
<xsd:enumeration value="Primary Account Street Address" />
<xsd:enumeration value="Primary Address Id" />
<xsd:enumeration value="Primary Assignment Denorm Flag" />
<xsd:enumeration value="Primary Assignment Manual Flag" />
<xsd:enumeration value="Primary Assignment System Flag" />
<xsd:enumeration value="Primary Assignment Type" />
<xsd:enumeration value="Primary Bill To Address Id" />
<xsd:enumeration value="Primary Bill To City" />
<xsd:enumeration value="Primary Bill To Country" />
<xsd:enumeration value="Primary Bill To First Name" />
<xsd:enumeration value="Primary Bill To Job Title" />
<xsd:enumeration value="Primary Bill To Last Name" />
<xsd:enumeration value="Primary Bill To Person Id" />
```

```
<xsd:enumeration value="Primary Bill To Postal Code" />
<xsd:enumeration value="Primary Bill To State" />
<xsd:enumeration value="Primary Bill To Street Address" />
<xsd:enumeration value="Primary Category Id" />
<xsd:enumeration value="Primary Contact Address Id" />
<xsd:enumeration value="Primary Employee Id" />
<xsd:enumeration value="Primary Employee Login" />
<xsd:enumeration value="Primary Fulfill InvLoc Integration Id" />
<xsd:enumeration value="Primary Fulfillment InvLoc ID" />
<xsd:enumeration value="Primary Fulfillment Inventory Location" />
<xsd:enumeration value="Primary Industry Id" />
<xsd:enumeration value="Primary Organization" />
<xsd:enumeration value="Primary Organization Id" />
<xsd:enumeration value="Primary Payer Account" />
<xsd:enumeration value="Primary Payer Account Id" />
<xsd:enumeration value="Primary Position Id" />
<xsd:enumeration value="Primary Service Agreement Id" />
<xsd:enumeration value="Primary Ship To Address Id" />
<xsd:enumeration value="Primary Ship To City" />
<xsd:enumeration value="Primary Ship To Country" />
<xsd:enumeration value="Primary Ship To First Name" />
<xsd:enumeration value="Primary Ship To Job Title" />
<xsd:enumeration value="Primary Ship To Last Name" />
<xsd:enumeration value="Primary Ship To Person Id" />
<xsd:enumeration value="Primary Ship To Postal Code" />
<xsd:enumeration value="Primary Ship To State" />
<xsd:enumeration value="Primary Ship To Street Address" />
<xsd:enumeration value="Primary Synonym Id" />
<xsd:enumeration value="Primary Territory Id" />
<xsd:enumeration value="Primary Type Id" />
<xsd:enumeration value="Profit" />
<xsd:enumeration value="Project Bill Type" />
<xsd:enumeration value="Project Comments" />
<xsd:enumeration value="Project Fix Fee" />
<xsd:enumeration value="Project Hour Limit" />
<xsd:enumeration value="Project Id" />
<xsd:enumeration value="Project Name" />
<xsd:enumeration value="Project Percentage of Fee" />
<xsd:enumeration value="Project Purchase Order" />
<xsd:enumeration value="Project Relationship Type" />
<xsd:enumeration value="Project Role" />
<xsd:enumeration value="Prospect Flag" />
<xsd:enumeration value="Province" />
<xsd:enumeration value="Public" />
<xsd:enumeration value="Reference Date" />
<xsd:enumeration value="Reference Flag" />
<xsd:enumeration value="Reference Stage" />
<xsd:enumeration value="Region" />
```

```
<xsd:enumeration value="Relationship Level" />
<xsd:enumeration value="Relationship Type" />
<xsd:enumeration value="Response Time" />
<xsd:enumeration value="Revenue" />
<xsd:enumeration value="Revenue Growth" />
<xsd:enumeration value="Revision Number" />
<xsd:enumeration value="Row Status" />
<xsd:enumeration value="Row Status Asterisk" />
<xsd:enumeration value="S-S Instance" />
<xsd:enumeration value="S-S Instance Id" />
<xsd:enumeration value="S-S Key Id" />
<xsd:enumeration value="Sales Rep" />
<xsd:enumeration value="Service Calendar" />
<xsd:enumeration value="Service Type" />
<xsd:enumeration value="Ship Address Flag" />
<xsd:enumeration value="Ship To City" />
<xsd:enumeration value="Ship To Country" />
<xsd:enumeration value="Ship To First Name" />
<xsd:enumeration value="Ship To Job Title" />
<xsd:enumeration value="Ship To Last Name" />
<xsd:enumeration value="Ship To Postal Code" />
<xsd:enumeration value="Ship To State" />
<xsd:enumeration value="Ship To Street Address" />
<xsd:enumeration value="Start Date" />
<xsd:enumeration value="State" />
<xsd:enumeration value="Strategies" />
<xsd:enumeration value="Strategy" />
<xsd:enumeration value="Street Address" />
<xsd:enumeration value="Street Address 2" />
<xsd:enumeration value="Success Factors" />
<xsd:enumeration value="Synonym" />
<xsd:enumeration value="Territory" />
<xsd:enumeration value="Territory Id" />
<xsd:enumeration value="Territory Item Type" />
<xsd:enumeration value="Timestamp" />
<xsd:enumeration value="Today" />
<xsd:enumeration value="Total Potential Volume" />
<xsd:enumeration value="Total Potential Volume Currency Code" />
<xsd:enumeration value="Total Potential Volume Exchange Date" />
<xsd:enumeration value="Type" />
<xsd:enumeration value="Type MVF" />
<xsd:enumeration value="VAT registration number" />
<xsd:enumeration value="Value Proposition" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
</xsd:sequence>
<xsd:attribute name="location" type="xsd:string" use="optional"
```

```
default="S/BO/Account/Account/queryWithView" />
  <xsd:attribute name="view" use="required">
  <xsd:simpleType>
  <xsd:restriction base="xsd:string">
  <xsd:enumeration value="SalesRepView" />
  <xsd:enumeration value="ManagerView" />
  <xsd:enumeration value="PersonalView" />
  <xsd:enumeration value="AllView" />
  <xsd:enumeration value="NoneSetView" />
  <xsd:enumeration value="OrganizationView" />
  <xsd:enumeration value="ContactView" />
  <xsd:enumeration value="GroupView" />
  <xsd:enumeration value="CatalogView" />
  <xsd:enumeration value="SubOrganizationView" />
  </xsd:restriction>
  </xsd:simpleType>
  </xsd:attribute>
  </xsd:complexType>
  </xsd:element>
  <xsd:complexType name="record">
  <xsd:sequence>
  <xsd:element name="Account_spcCompetitors" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcCondition" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcMarkets" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcOrganization_spcIntegration_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Account_spcProducts" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcRole" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcStatus" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcTrend" type="xsd:string" minOccurs="0" />
  <xsd:element name="Address_spcActive_spcStatus" type="xsd:string" minOccurs="0" />
  <xsd:element name="Address_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Address_spcIntegration_spcId" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Agreement_spcEnd_spcDate" type="xsd:string" minOccurs="0" />
  <xsd:element name="Agreement_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Agreement_spcStart_spcDate" type="xsd:string" minOccurs="0" />
  <xsd:element name="Agreement_spcStatus" type="xsd:string" minOccurs="0" />
  <xsd:element name="Algorithm_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="Alias" type="xsd:string" minOccurs="0" />
  <xsd:element name="Annual_spcRevenue" type="xsd:string" minOccurs="0" />
  <xsd:element name="Assignment_spcArea_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Assignment_spcCountry_spcCode" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Assignment_spcDenorm_spcFlag" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Assignment_spcExcluded" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Assignment_spcManual_spcFlag" type="xsd:string" minOccurs="0"
```

```
/>
  <xsd:element name="Assignment_spcSystem_spcFlag" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Assignment_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="BO_spcExport_spcStatus" type="xsd:string" minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcDistribution_spcChannel" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcOrder_spcQuery_spcEnd_spc_spcDt"
type="xsd:date" minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcOrder_spcQuery_spcStart_spc_spcDt"
type="xsd:date" minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcSales_spcArea" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcSales_spcArea_spcDivision_spcCode"
type="xsd:string" minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcSales_spcOrganization" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Bill_spcAddress_spcFlag" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcCity" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcCountry" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcFirst_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcJob_spcTitle" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcLast_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcPostal_spcCode" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Bill_spcTo_spcState" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcStreet_spcAddress" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Block_spcCredit_spcFlag" type="xsd:string" minOccurs="0" />
  <xsd:element name="Business_spcProfile" type="xsd:string" minOccurs="0" />
  <xsd:element name="CSN" type="xsd:string" minOccurs="0" />
  <xsd:element name="Category" type="xsd:string" minOccurs="0" />
  <xsd:element name="Category_spcValue" type="xsd:string" minOccurs="0" />
  <xsd:element name="City" type="xsd:string" minOccurs="0" />
  <xsd:element name="City_spcState" type="xsd:string" minOccurs="0" />
  <xsd:element name="Competitor" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Country" type="xsd:string" minOccurs="0" />
  <xsd:element name="County" type="xsd:string" minOccurs="0" />
  <xsd:element name="Credit_spcControl_spcArea_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Credit_spcCurrency_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Credit_spcLimit_spcAmount" type="xsd:string" minOccurs="0" />
  <xsd:element name="Credit_spcProfile_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Culture" type="xsd:string" minOccurs="0" />
  <xsd:element name="Currency_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Current_spcVolume" type="xsd:string" minOccurs="0" />
  <xsd:element name="Current_spcVolume_spcCurrency_spcCode" type="xsd:string"
```

```
minOccurs="0" />
  <xsd:element name="Current_spcVolume_spcExchange_spcDate" type="xsd:date"
minOccurs="0" />
  <xsd:element name="Customer_spcAccount_spcGroup" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="DNBReport" type="xsd:string" minOccurs="0" />
  <xsd:element name="DUNS_spcIntcode" type="xsd:string" minOccurs="0" />
  <xsd:element name="DUNS_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="Date_spcFormed" type="xsd:date" minOccurs="0" />
  <xsd:element name="DeDup_spcKey_spcModification_spcDate" type="xsd:string"
minOccurs="0" />
  <xsd:element name="DeDup_spcKey_spcUpdate" type="xsd:string" minOccurs="0" />
  <xsd:element name="DeDup_spcKeys" type="xsd:string" minOccurs="0" />
  <xsd:element name="DeDup_spcLast_spcMatch_spcDate" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="DeDup_spcToken" type="xsd:string" minOccurs="0" />
  <xsd:element name="Deduplication_spcMatch_spcScore" type="xsd:int" minOccurs="0"
/>
  <xsd:element name="Deduplication_spcObject_spcId" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Description" type="xsd:string" minOccurs="0" />
  <xsd:element name="Disable_spcDataCleansing" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Disable_spcMailings" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Division" type="xsd:string" minOccurs="0" />
  <xsd:element name="Domestic_spcUltimate_spcDUNS" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Dummy" type="xsd:string" minOccurs="0" />
  <xsd:element name="EAI_spcSync_spcDate" type="xsd:string" minOccurs="0" />
  <xsd:element name="EAI_spcSync_spcError_spcText" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="EAI_spcSync_spcStatus_spcCode" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Email_spcAddress" type="xsd:string" minOccurs="0" />
  <xsd:element name="Employee_spcHere" type="xsd:int" minOccurs="0" />
  <xsd:element name="Employees" type="xsd:int" minOccurs="0" />
  <xsd:element name="Expertise" type="xsd:string" minOccurs="0" />
  <xsd:element name="Explorer_spcLabel" type="xsd:string" minOccurs="0" />
  <xsd:element name="Fax_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="First_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Fiscal_spcYear_spcEnd" type="xsd:date" minOccurs="0" />
  <xsd:element name="Freight_spcTerms" type="xsd:string" minOccurs="0" />
  <xsd:element name="Freight_spcTerms_spcInfo" type="xsd:string" minOccurs="0" />
  <xsd:element name="Full_spcAddress" type="xsd:string" minOccurs="0" />
  <xsd:element name="Full_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="GSA_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Global_spcUltimate_spcDUNS" type="xsd:string" minOccurs="0" />
  <xsd:element name="Goals" type="xsd:string" minOccurs="0" />
  <xsd:element name="Group_spcType_spcCode" type="xsd:string" minOccurs="0" />
```

```
  <xsd:element name="Home_spcPage" type="xsd:string" minOccurs="0" />
  <xsd:element name="Industry" type="xsd:string" minOccurs="0" />
  <xsd:element name="Industry_spcCondition" type="xsd:string" minOccurs="0" />
  <xsd:element name="Industry_spcTrend" type="xsd:string" minOccurs="0" />
  <xsd:element name="Integration_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Internal_spcOrg_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Joined_spcSynonym" type="xsd:string" minOccurs="0" />
  <xsd:element name="Key_spcCompetitors" type="xsd:string" minOccurs="0" />
  <xsd:element name="Language_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Last_spcClnse_spcDate" type="xsd:string" minOccurs="0" />
  <xsd:element name="Last_spcManager_spcReview_spcDate" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Last_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Last_spcReview_spcManager_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Last_spcUpdate_spc-_spcSDQ" type="xsd:string" minOccurs="0" />
  <xsd:element name="Line_spcof_spcBusiness" type="xsd:string" minOccurs="0" />
  <xsd:element name="Location" type="xsd:string" minOccurs="0" />
  <xsd:element name="Location_spcLevel" type="xsd:string" minOccurs="0" />
  <xsd:element name="Main_spcAddress_spcFlag" type="xsd:string" minOccurs="0" />
  <xsd:element name="Main_spcFax_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="Main_spcPhone_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="Managers_spcReview" type="xsd:string" minOccurs="0" />
  <xsd:element name="Marketing" type="xsd:string" minOccurs="0" />
  <xsd:element name="Merge_spcSequence_spcNumber" type="xsd:int" minOccurs="0" />
  <xsd:element name="Mission" type="xsd:string" minOccurs="0" />
  <xsd:element name="Name" type="xsd:string" minOccurs="0" />
  <xsd:element name="Name_spcand_spcLocation" type="xsd:string" minOccurs="0" />
  <xsd:element name="Not_spcManager_spcFlag" type="xsd:string" minOccurs="0" />
  <xsd:element name="Notes" type="xsd:string" minOccurs="0" />
  <xsd:element name="Objectives" type="xsd:string" minOccurs="0" />
  <xsd:element name="Organization" type="xsd:string" minOccurs="0" />
  <xsd:element name="Organization_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Organization_spcIntegration_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Our_spcPosition" type="xsd:string" minOccurs="0" />
  <xsd:element name="Outline_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="PO_spcApproved_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="PO_spcAuto_spcApproval_spcCurrency_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="PO_spcAuto_spcApproval_spcDate" type="xsd:date" minOccurs="0"
/>
  <xsd:element name="PO_spcAuto_spcApproval_spcLimit" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Parent_spcAccount_spcDivision" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Parent_spcAccount_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Parent_spcAccount_spcIntegration_spcId" type="xsd:string"
```

```
minOccurs="0" />
  <xsd:element name="Parent_spcAccount_spcLocation" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Parent_spcAccount_spcLocation_spcLevel" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Parent_spcAccount_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Parent_spcAccount_spcRegion" type="xsd:string" minOccurs="0" />
  <xsd:element name="Parent_spcHQ_spcDUNS" type="xsd:string" minOccurs="0" />
  <xsd:element name="Partner_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Partners" type="xsd:string" minOccurs="0" />
  <xsd:element name="Party_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Party_spcType_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Party_spcUId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Philosophy" type="xsd:string" minOccurs="0" />
  <xsd:element name="Phone_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="Position" type="xsd:string" minOccurs="0" />
  <xsd:element name="Position_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Position_spcIntegration_spcId" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Position_spcTerritory" type="xsd:string" minOccurs="0" />
  <xsd:element name="Postal_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Price_spcList" type="xsd:string" minOccurs="0" />
  <xsd:element name="Price_spcList_spcEnd_spcDate" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Price_spcList_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Price_spcList_spcIntegration_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Price_spcList_spcStart_spcDate" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcAccount_spcCity" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcAccount_spcCountry" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcAccount_spcPostal_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcAccount_spcState" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcAccount_spcStreet_spcAddress" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcAddress_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcAssignment_spcDenorm_spcFlag" type="xsd:boolean"
minOccurs="0" />
  <xsd:element name="Primary_spcAssignment_spcManual_spcFlag" type="xsd:boolean"
minOccurs="0" />
  <xsd:element name="Primary_spcAssignment_spcSystem_spcFlag" type="xsd:boolean"
minOccurs="0" />
  <xsd:element name="Primary_spcAssignment_spcType" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcBill_spcTo_spcAddress_spcId" type="xsd:string"
minOccurs="0" />
```

```
  <xsd:element name="Primary_spcBill_spcTo_spcCity" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcBill_spcTo_spcCountry" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcFirst_spcName" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcJob_spcTitle" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcLast_spcName" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcPerson_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcPostal_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcState" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcBill_spcTo_spcStreet_spcAddress" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcCategory_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcContact_spcAddress_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcEmployee_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcEmployee_spcLogin" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcFulfill_spcInvLoc_spcIntegration_spcId"
type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcFulfillment_spcInvLoc_spcID" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcFulfillment_spcInventory_spcLocation"
type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcIndustry_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcOrganization" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcOrganization_spcId" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcPayer_spcAccount" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcPayer_spcAccount_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcPosition_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcService_spcAgreement_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcAddress_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcCity" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcShip_spcTo_spcCountry" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcFirst_spcName" type="xsd:string"
minOccurs="0" />
```

```
  <xsd:element name="Primary_spcShip_spcTo_spcJob_spcTitle" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcLast_spcName" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcPerson_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcPostal_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcState" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcShip_spcTo_spcStreet_spcAddress" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcSynonym_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcTerritory_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcType_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Profit" type="xsd:string" minOccurs="0" />
  <xsd:element name="Project_spcBill_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="Project_spcComments" type="xsd:string" minOccurs="0" />
  <xsd:element name="Project_spcFix_spcFee" type="xsd:int" minOccurs="0" />
  <xsd:element name="Project_spcHour_spcLimit" type="xsd:int" minOccurs="0" />
  <xsd:element name="Project_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Project_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Project_spcPercentage_spcof_spcFee" type="xsd:int"
minOccurs="0" />
  <xsd:element name="Project_spcPurchase_spcOrder" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Project_spcRelationship_spcType" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Project_spcRole" type="xsd:string" minOccurs="0" />
  <xsd:element name="Prospect_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Province" type="xsd:string" minOccurs="0" />
  <xsd:element name="Public" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Reference_spcDate" type="xsd:date" minOccurs="0" />
  <xsd:element name="Reference_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Reference_spcStage" type="xsd:string" minOccurs="0" />
  <xsd:element name="Region" type="xsd:string" minOccurs="0" />
  <xsd:element name="Relationship_spcLevel" type="xsd:string" minOccurs="0" />
  <xsd:element name="Relationship_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="Response_spcTime" type="xsd:string" minOccurs="0" />
  <xsd:element name="Revenue" type="xsd:string" minOccurs="0" />
  <xsd:element name="Revenue_spcGrowth" type="xsd:int" minOccurs="0" />
  <xsd:element name="Revision_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="Row_spcStatus" type="xsd:string" minOccurs="0" />
  <xsd:element name="Row_spcStatus_spcAsterisk" type="xsd:string" minOccurs="0" />
  <xsd:element name="S-S_spcInstance" type="xsd:string" minOccurs="0" />
  <xsd:element name="S-S_spcInstance_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="S-S_spcKey_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Sales_spcRep" type="xsd:string" minOccurs="0" />
```

```
  <xsd:element name="Service_spcCalendar" type="xsd:string" minOccurs="0" />
  <xsd:element name="Service_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcAddress_spcFlag" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcCity" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcCountry" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcFirst_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcJob_spcTitle" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcLast_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcPostal_spcCode" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Ship_spcTo_spcState" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcStreet_spcAddress" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Start_spcDate" type="xsd:date" minOccurs="0" />
  <xsd:element name="State" type="xsd:string" minOccurs="0" />
  <xsd:element name="Strategies" type="xsd:string" minOccurs="0" />
  <xsd:element name="Strategy" type="xsd:string" minOccurs="0" />
  <xsd:element name="Street_spcAddress" type="xsd:string" minOccurs="0" />
  <xsd:element name="Street_spcAddress_spc2" type="xsd:string" minOccurs="0" />
  <xsd:element name="Success_spcFactors" type="xsd:string" minOccurs="0" />
  <xsd:element name="Synonym" type="xsd:string" minOccurs="0" />
  <xsd:element name="Territory" type="xsd:string" minOccurs="0" />
  <xsd:element name="Territory_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Territory_spcItem_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="Timestamp" type="xsd:string" minOccurs="0" />
  <xsd:element name="Today" type="xsd:date" minOccurs="0" />
  <xsd:element name="Total_spcPotential_spcVolume" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Total_spcPotential_spcVolume_spcCurrency_spcCode"
type="xsd:string" minOccurs="0" />
  <xsd:element name="Total_spcPotential_spcVolume_spcExchange_spcDate"
type="xsd:date" minOccurs="0" />
  <xsd:element name="Type" type="xsd:string" minOccurs="0" />
  <xsd:element name="Type_spcMVF" type="xsd:string" minOccurs="0" />
  <xsd:element name="VAT_spcregistration_spcnumber" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Value_spcProposition" type="xsd:string" minOccurs="0" />
  </xsd:sequence>
  </xsd:complexType>
  </xsd:schema>
```

Example: Account QueryWithView Response Schema

```
  <?xml version="1.0" encoding="UTF-8" ?>
  <!--  Generated by the iBSE 2004-04-09T19:44:34Z
  -->
  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:iwaysoftware:adapter:siebel:bo:oct2003"
xmlns:z="urn:iwaysoftware:adapter:siebel:bo:oct2003" elementFormDefault="qualified">
  <xsd:element name="SiebelResponse">
  <xsd:complexType>
  <xsd:sequence>
  <xsd:element name="record" minOccurs="0" maxOccurs="unbounded" type="z:record" />
  </xsd:sequence>
  <xsd:attribute name="status" use="required">
  <xsd:simpleType>
  <xsd:restriction base="xsd:string">
  <xsd:enumeration value="success" />
  <xsd:enumeration value="failure" />
  </xsd:restriction>
  </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="reason" type="xsd:string" use="optional" />
  </xsd:complexType>
  </xsd:element>
  <xsd:complexType name="record">
  <xsd:sequence>
  <xsd:element name="Account_spcCompetitors" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcCondition" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcMarkets" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcOrganization_spcIntegration_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Account_spcProducts" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcRole" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcStatus" type="xsd:string" minOccurs="0" />
  <xsd:element name="Account_spcTrend" type="xsd:string" minOccurs="0" />
  <xsd:element name="Address_spcActive_spcStatus" type="xsd:string" minOccurs="0" />
  <xsd:element name="Address_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Address_spcIntegration_spcId" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Agreement_spcEnd_spcDate" type="xsd:string" minOccurs="0" />
  <xsd:element name="Agreement_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Agreement_spcStart_spcDate" type="xsd:string" minOccurs="0" />
  <xsd:element name="Agreement_spcStatus" type="xsd:string" minOccurs="0" />
  <xsd:element name="Algorithm_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="Alias" type="xsd:string" minOccurs="0" />
  <xsd:element name="Annual_spcRevenue" type="xsd:string" minOccurs="0" />
  <xsd:element name="Assignment_spcArea_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Assignment_spcCountry_spcCode" type="xsd:string" minOccurs="0"
/>
```

```
  <xsd:element name="Assignment_spcDenorm_spcFlag" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Assignment_spcExcluded" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Assignment_spcManual_spcFlag" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Assignment_spcSystem_spcFlag" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Assignment_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="BO_spcExport_spcStatus" type="xsd:string" minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcDistribution_spcChannel" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcOrder_spcQuery_spcEnd_spc_spcDt"
type="xsd:date" minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcOrder_spcQuery_spcStart_spc_spcDt"
type="xsd:date" minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcSales_spcArea" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcSales_spcArea_spcDivision_spcCode"
type="xsd:string" minOccurs="0" />
  <xsd:element name="Back_spcOffice_spcSales_spcOrganization" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Bill_spcAddress_spcFlag" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcCity" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcCountry" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcFirst_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcJob_spcTitle" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcLast_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcPostal_spcCode" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Bill_spcTo_spcState" type="xsd:string" minOccurs="0" />
  <xsd:element name="Bill_spcTo_spcStreet_spcAddress" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Block_spcCredit_spcFlag" type="xsd:string" minOccurs="0" />
  <xsd:element name="Business_spcProfile" type="xsd:string" minOccurs="0" />
  <xsd:element name="CSN" type="xsd:string" minOccurs="0" />
  <xsd:element name="Category" type="xsd:string" minOccurs="0" />
  <xsd:element name="Category_spcValue" type="xsd:string" minOccurs="0" />
  <xsd:element name="City" type="xsd:string" minOccurs="0" />
  <xsd:element name="City_spcState" type="xsd:string" minOccurs="0" />
  <xsd:element name="Competitor" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Country" type="xsd:string" minOccurs="0" />
  <xsd:element name="County" type="xsd:string" minOccurs="0" />
  <xsd:element name="Credit_spcControl_spcArea_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Credit_spcCurrency_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Credit_spcLimit_spcAmount" type="xsd:string" minOccurs="0" />
  <xsd:element name="Credit_spcProfile_spcId" type="xsd:string" minOccurs="0" />
```

```
  <xsd:element name="Culture" type="xsd:string" minOccurs="0" />
  <xsd:element name="Currency_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Current_spcVolume" type="xsd:string" minOccurs="0" />
  <xsd:element name="Current_spcVolume_spcCurrency_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Current_spcVolume_spcExchange_spcDate" type="xsd:date"
minOccurs="0" />
  <xsd:element name="Customer_spcAccount_spcGroup" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="DNBReport" type="xsd:string" minOccurs="0" />
  <xsd:element name="DUNS_spcIntcode" type="xsd:string" minOccurs="0" />
  <xsd:element name="DUNS_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="Date_spcFormed" type="xsd:date" minOccurs="0" />
  <xsd:element name="DeDup_spcKey_spcModification_spcDate" type="xsd:string"
minOccurs="0" />
  <xsd:element name="DeDup_spcKey_spcUpdate" type="xsd:string" minOccurs="0" />
  <xsd:element name="DeDup_spcKeys" type="xsd:string" minOccurs="0" />
  <xsd:element name="DeDup_spcLast_spcMatch_spcDate" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="DeDup_spcToken" type="xsd:string" minOccurs="0" />
  <xsd:element name="Deduplication_spcMatch_spcScore" type="xsd:int" minOccurs="0"
/>
  <xsd:element name="Deduplication_spcObject_spcId" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Description" type="xsd:string" minOccurs="0" />
  <xsd:element name="Disable_spcDataCleansing" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Disable_spcMailings" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Division" type="xsd:string" minOccurs="0" />
  <xsd:element name="Domestic_spcUltimate_spcDUNS" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Dummy" type="xsd:string" minOccurs="0" />
  <xsd:element name="EAI_spcSync_spcDate" type="xsd:string" minOccurs="0" />
  <xsd:element name="EAI_spcSync_spcError_spcText" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="EAI_spcSync_spcStatus_spcCode" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Email_spcAddress" type="xsd:string" minOccurs="0" />
  <xsd:element name="Employee_spcHere" type="xsd:int" minOccurs="0" />
  <xsd:element name="Employees" type="xsd:int" minOccurs="0" />
  <xsd:element name="Expertise" type="xsd:string" minOccurs="0" />
  <xsd:element name="Explorer_spcLabel" type="xsd:string" minOccurs="0" />
  <xsd:element name="Fax_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="First_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Fiscal_spcYear_spcEnd" type="xsd:date" minOccurs="0" />
  <xsd:element name="Freight_spcTerms" type="xsd:string" minOccurs="0" />
  <xsd:element name="Freight_spcTerms_spcInfo" type="xsd:string" minOccurs="0" />
  <xsd:element name="Full_spcAddress" type="xsd:string" minOccurs="0" />
  <xsd:element name="Full_spcName" type="xsd:string" minOccurs="0" />
```

```
  <xsd:element name="GSA_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Global_spcUltimate_spcDUNS" type="xsd:string" minOccurs="0" />
  <xsd:element name="Goals" type="xsd:string" minOccurs="0" />
  <xsd:element name="Group_spcType_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Home_spcPage" type="xsd:string" minOccurs="0" />
  <xsd:element name="Industry" type="xsd:string" minOccurs="0" />
  <xsd:element name="Industry_spcCondition" type="xsd:string" minOccurs="0" />
  <xsd:element name="Industry_spcTrend" type="xsd:string" minOccurs="0" />
  <xsd:element name="Integration_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Internal_spcOrg_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Joined_spcSynonym" type="xsd:string" minOccurs="0" />
  <xsd:element name="Key_spcCompetitors" type="xsd:string" minOccurs="0" />
  <xsd:element name="Language_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Last_spcClnse_spcDate" type="xsd:string" minOccurs="0" />
  <xsd:element name="Last_spcManager_spcReview_spcDate" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Last_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Last_spcReview_spcManager_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Last_spcUpdate_spc-_spcSDQ" type="xsd:string" minOccurs="0" />
  <xsd:element name="Line_spcof_spcBusiness" type="xsd:string" minOccurs="0" />
  <xsd:element name="Location" type="xsd:string" minOccurs="0" />
  <xsd:element name="Location_spcLevel" type="xsd:string" minOccurs="0" />
  <xsd:element name="Main_spcAddress_spcFlag" type="xsd:string" minOccurs="0" />
  <xsd:element name="Main_spcFax_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="Main_spcPhone_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="Managers_spcReview" type="xsd:string" minOccurs="0" />
  <xsd:element name="Marketing" type="xsd:string" minOccurs="0" />
  <xsd:element name="Merge_spcSequence_spcNumber" type="xsd:int" minOccurs="0" />
  <xsd:element name="Mission" type="xsd:string" minOccurs="0" />
  <xsd:element name="Name" type="xsd:string" minOccurs="0" />
  <xsd:element name="Name_spcand_spcLocation" type="xsd:string" minOccurs="0" />
  <xsd:element name="Not_spcManager_spcFlag" type="xsd:string" minOccurs="0" />
  <xsd:element name="Notes" type="xsd:string" minOccurs="0" />
  <xsd:element name="Objectives" type="xsd:string" minOccurs="0" />
  <xsd:element name="Organization" type="xsd:string" minOccurs="0" />
  <xsd:element name="Organization_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Organization_spcIntegration_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Our_spcPosition" type="xsd:string" minOccurs="0" />
  <xsd:element name="Outline_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="PO_spcApproved_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="PO_spcAuto_spcApproval_spcCurrency_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="PO_spcAuto_spcApproval_spcDate" type="xsd:date" minOccurs="0"
/>
  <xsd:element name="PO_spcAuto_spcApproval_spcLimit" type="xsd:string"
minOccurs="0" />
```

```
  <xsd:element name="Parent_spcAccount_spcDivision" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Parent_spcAccount_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Parent_spcAccount_spcIntegration_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Parent_spcAccount_spcLocation" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Parent_spcAccount_spcLocation_spcLevel" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Parent_spcAccount_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Parent_spcAccount_spcRegion" type="xsd:string" minOccurs="0" />
  <xsd:element name="Parent_spcHQ_spcDUNS" type="xsd:string" minOccurs="0" />
  <xsd:element name="Partner_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Partners" type="xsd:string" minOccurs="0" />
  <xsd:element name="Party_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Party_spcType_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Party_spcUId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Philosophy" type="xsd:string" minOccurs="0" />
  <xsd:element name="Phone_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="Position" type="xsd:string" minOccurs="0" />
  <xsd:element name="Position_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Position_spcIntegration_spcId" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Position_spcTerritory" type="xsd:string" minOccurs="0" />
  <xsd:element name="Postal_spcCode" type="xsd:string" minOccurs="0" />
  <xsd:element name="Price_spcList" type="xsd:string" minOccurs="0" />
  <xsd:element name="Price_spcList_spcEnd_spcDate" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Price_spcList_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Price_spcList_spcIntegration_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Price_spcList_spcStart_spcDate" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcAccount_spcCity" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcAccount_spcCountry" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcAccount_spcPostal_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcAccount_spcState" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcAccount_spcStreet_spcAddress" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcAddress_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcAssignment_spcDenorm_spcFlag" type="xsd:boolean"
minOccurs="0" />
  <xsd:element name="Primary_spcAssignment_spcManual_spcFlag" type="xsd:boolean"
minOccurs="0" />
  <xsd:element name="Primary_spcAssignment_spcSystem_spcFlag" type="xsd:boolean"
minOccurs="0" />
```

```
  <xsd:element name="Primary_spcAssignment_spcType" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcBill_spcTo_spcAddress_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcCity" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcBill_spcTo_spcCountry" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcFirst_spcName" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcJob_spcTitle" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcLast_spcName" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcPerson_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcPostal_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcBill_spcTo_spcState" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcBill_spcTo_spcStreet_spcAddress" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcCategory_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcContact_spcAddress_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcEmployee_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcEmployee_spcLogin" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcFulfill_spcInvLoc_spcIntegration_spcId"
type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcFulfillment_spcInvLoc_spcID" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcFulfillment_spcInventory_spcLocation"
type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcIndustry_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcOrganization" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcOrganization_spcId" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcPayer_spcAccount" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcPayer_spcAccount_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcPosition_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcService_spcAgreement_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcAddress_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcCity" type="xsd:string" minOccurs="0"
/>
```

```
  <xsd:element name="Primary_spcShip_spcTo_spcCountry" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcFirst_spcName" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcJob_spcTitle" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcLast_spcName" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcPerson_spcId" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcPostal_spcCode" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcShip_spcTo_spcState" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Primary_spcShip_spcTo_spcStreet_spcAddress" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Primary_spcSynonym_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcTerritory_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Primary_spcType_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Profit" type="xsd:string" minOccurs="0" />
  <xsd:element name="Project_spcBill_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="Project_spcComments" type="xsd:string" minOccurs="0" />
  <xsd:element name="Project_spcFix_spcFee" type="xsd:int" minOccurs="0" />
  <xsd:element name="Project_spcHour_spcLimit" type="xsd:int" minOccurs="0" />
  <xsd:element name="Project_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Project_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Project_spcPercentage_spcof_spcFee" type="xsd:int"
minOccurs="0" />
  <xsd:element name="Project_spcPurchase_spcOrder" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Project_spcRelationship_spcType" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Project_spcRole" type="xsd:string" minOccurs="0" />
  <xsd:element name="Prospect_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Province" type="xsd:string" minOccurs="0" />
  <xsd:element name="Public" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Reference_spcDate" type="xsd:date" minOccurs="0" />
  <xsd:element name="Reference_spcFlag" type="xsd:boolean" minOccurs="0" />
  <xsd:element name="Reference_spcStage" type="xsd:string" minOccurs="0" />
  <xsd:element name="Region" type="xsd:string" minOccurs="0" />
  <xsd:element name="Relationship_spcLevel" type="xsd:string" minOccurs="0" />
  <xsd:element name="Relationship_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="Response_spcTime" type="xsd:string" minOccurs="0" />
  <xsd:element name="Revenue" type="xsd:string" minOccurs="0" />
  <xsd:element name="Revenue_spcGrowth" type="xsd:int" minOccurs="0" />
  <xsd:element name="Revision_spcNumber" type="xsd:string" minOccurs="0" />
  <xsd:element name="Row_spcStatus" type="xsd:string" minOccurs="0" />
  <xsd:element name="Row_spcStatus_spcAsterisk" type="xsd:string" minOccurs="0" />
```

```
  <xsd:element name="S-S_spcInstance" type="xsd:string" minOccurs="0" />
  <xsd:element name="S-S_spcInstance_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="S-S_spcKey_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Sales_spcRep" type="xsd:string" minOccurs="0" />
  <xsd:element name="Service_spcCalendar" type="xsd:string" minOccurs="0" />
  <xsd:element name="Service_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcAddress_spcFlag" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcCity" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcCountry" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcFirst_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcJob_spcTitle" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcLast_spcName" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcPostal_spcCode" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Ship_spcTo_spcState" type="xsd:string" minOccurs="0" />
  <xsd:element name="Ship_spcTo_spcStreet_spcAddress" type="xsd:string"
minOccurs="0" />
  <xsd:element name="Start_spcDate" type="xsd:date" minOccurs="0" />
  <xsd:element name="State" type="xsd:string" minOccurs="0" />
  <xsd:element name="Strategies" type="xsd:string" minOccurs="0" />
  <xsd:element name="Strategy" type="xsd:string" minOccurs="0" />
  <xsd:element name="Street_spcAddress" type="xsd:string" minOccurs="0" />
  <xsd:element name="Street_spcAddress_spc2" type="xsd:string" minOccurs="0" />
  <xsd:element name="Success_spcFactors" type="xsd:string" minOccurs="0" />
  <xsd:element name="Synonym" type="xsd:string" minOccurs="0" />
  <xsd:element name="Territory" type="xsd:string" minOccurs="0" />
  <xsd:element name="Territory_spcId" type="xsd:string" minOccurs="0" />
  <xsd:element name="Territory_spcItem_spcType" type="xsd:string" minOccurs="0" />
  <xsd:element name="Timestamp" type="xsd:string" minOccurs="0" />
  <xsd:element name="Today" type="xsd:date" minOccurs="0" />
  <xsd:element name="Total_spcPotential_spcVolume" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Total_spcPotential_spcVolume_spcCurrency_spcCode"
type="xsd:string" minOccurs="0" />
  <xsd:element name="Total_spcPotential_spcVolume_spcExchange_spcDate"
type="xsd:date" minOccurs="0" />
  <xsd:element name="Type" type="xsd:string" minOccurs="0" />
  <xsd:element name="Type_spcMVF" type="xsd:string" minOccurs="0" />
  <xsd:element name="VAT_spcregistration_spcnumber" type="xsd:string" minOccurs="0"
/>
  <xsd:element name="Value_spcProposition" type="xsd:string" minOccurs="0" />
  </xsd:sequence>
  </xsd:complexType>
  </xsd:schema>
```

Example: XML Input for Account QueryWithView

```
<Siebel location="S/BO/Account/Account/queryWithView" view="AllView">
            <select>
                        <Name>SIEBEL*</Name>
            </select>
            <field>Name</field>
            <field>Location</field>
  </Siebel>
Sample XML for Account Add Service Response
```

Example: XML for Account QueryWithView Response

```
<?xml version="1.0" encoding="UTF-8" ?>
  <SOAP-ENV:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
  <NewAccountQueryResponse
xmlns="urn:iwaysoftware:ibse:jul2003:NewAccountQuery:response"
cid="B24B35E1A39476D1D9EFB8F2A0F4818A">
  <SiebelResponse status="success">
  <record>
  <Name>SIEBEL1 ACCOUNT</Name>
  <Location>ONE</Location>
  </record>
  <record>
  <Name>SIEBEL2 ACCOUNT</Name>
  <Location>TWO</Location>
  </record>
  <record>
  <Name>SIEBEL3</Name>
  <Location>RR</Location>
  </record>
  <record>
  <Name>SIEBEL4</Name>
  <Location>FOUR</Location>
  </record>
  </SiebelResponse>
  </NewAccountQueryResponse>
  </SOAP-ENV:Body>

  </SOAP-ENV:Envelope>
```

APPENDIX B

Siebel Workflows

Topics:

- Overview
- Creating a Siebel Workflow

This section describes Siebel Workflows relating to the processing of Siebel Integration Objects using Siebel XML.

Overview

When using Siebel XML to integrate with Siebel Integration Objects, the interface uses a Siebel Workflow. A Siebel Workflow is defined within Siebel to emit or to receive Siebel XML. In either case, emitting or receiving is handled by Siebel transport services for MQSeries, File, or HTTP. The following topics discuss the use and creation of workflows for Siebel version 7.0 that employ the supported transport services.

Note: This section is intended as a supplement to the documentation designed for the BEA WebLogic Adapter for Siebel user and is not intended as a substitute for Siebel documentation. For complete and up-to-date information on Siebel Workflow and policy topics, see the *Siebel Bookshelf* for your Siebel system.

Siebel Workflows

A Siebel Workflow is a series of Siebel Business Services linked together to accomplish a business task. You create workflows using the Siebel Client Workflow Administration screens. Workflows are invoked through one of the following methods:

- Using a workflow policy
- Using a run-time event (Siebel Event)
- Using a script (eScript or Siebel VB)

The following topic briefly describes how to invoke the workflow through a policy condition. For more information on policy and other methods, see the documentation on the *Siebel Bookshelf*.

Using a Policy to Invoke a Siebel EAI Workflow

A workflow policy is defined by a set of conditions that executes a set of defined actions. A Siebel workflow policy consists of:

- Conditions that define circumstances, based on changes in the state of a Siebel database.
- Actions that define steps taken when conditions are fulfilled.

Creating a policy to invoke a workflow as an action involves the following steps:

1. Define an action to be executed after a policy is triggered. Use the Run Integration Process program.
2. Create a policy by setting conditions and selecting appropriate policy groups and actions.
3. Activate the policy by choosing an activation date.

4. Run the Generate Triggers server task from Server Administration windows to set the conditions to be monitored.
5. Start the Workflow Monitor agent after editing with the appropriate policy group (to which your policy belongs) to evaluate whether to perform an action.
6. Start the Workflow Action Agent server task from Server Administration windows to perform the action.

For more information on the previous steps, see the documentation on the *Siebel Bookshelf*.

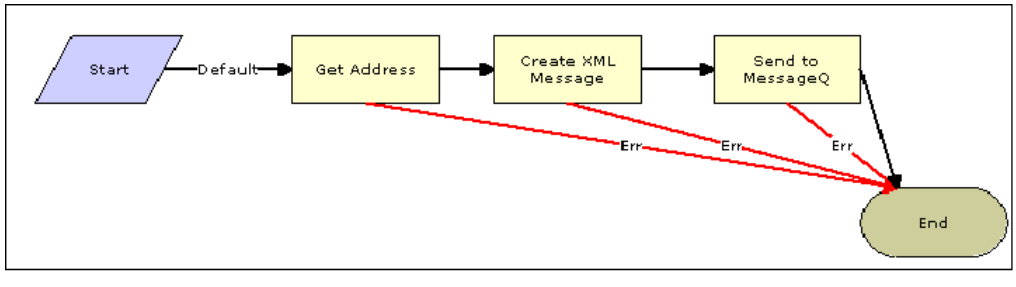
Siebel Workflow - Outbound

When a Siebel Workflow is triggered based on a Siebel policy, run-time, or script (eScript or Siebel VB) event, the result is the generation of a Siebel XML document that is placed on one of the Siebel transports. For example, when you add a new account in the Siebel Call Center application, you can design and configure a workflow to be triggered on the account transaction. You can design the workflow to extract the data for the new record, convert it to Siebel XML, and then, place it on an MQSeries message queue.

In this example, the Siebel Workflow process executes the following series of Siebel Business Services:

1. Calls the Siebel EAI Siebel Adapter, which queries for the newly updated account record, and places the data in its original internal structure into memory.
2. Calls the Siebel EAI XML Converter, which converts the data into an XML message.
3. Calls the Siebel EAI MQSeries Transport, which places the newly created XML message into the appropriate MQSeries message queue.

   After the message is placed in the message queue, it is retrieved by the BEA WebLogic Adapter for Siebel. The following illustration shows the Workflow sequence described in the previous steps. The flow boxes are from left to right: Start, Get Address, Create XML Message, Send to MessageQ, and End.



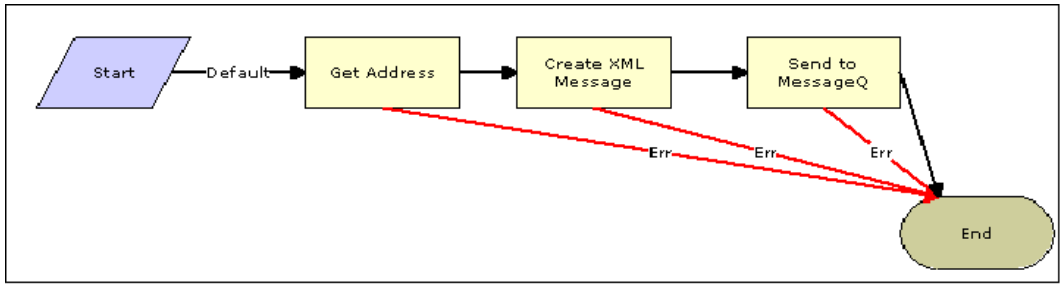
Siebel Workflow - Inbound

A Siebel Workflow that is triggered by an external event begins by receiving a Siebel XML document placed on one of its transports. The result might be the update of a Siebel record using the XML as input, for example, when a new account is added in another CRM system but also must be updated in the Siebel Call Center application. You can design and configure a Workflow to receive or listen on an MQSeries message queue. Upon receipt of the XML message, the Workflow processes the transaction into the Siebel system to update the record.

In this example, upon receipt of the Siebel XML message in the message queue, the Siebel MQSeries Receiver server task initiates a Siebel Workflow process, which in turn executes a series of Siebel Business Services as follows:

1. Calls the Siebel EAI XML Converter, which converts the XML message into Siebel internal format.
2. Calls the Siebel EAI Siebel Adapter, which applies the newly updated account record based on the methods defined in its service.

The following illustration shows the inbound Workflow process based on the previous description. The flow boxes are from left to right: Start, Get Address, Create XML Message, Send to MessageQ, and End.



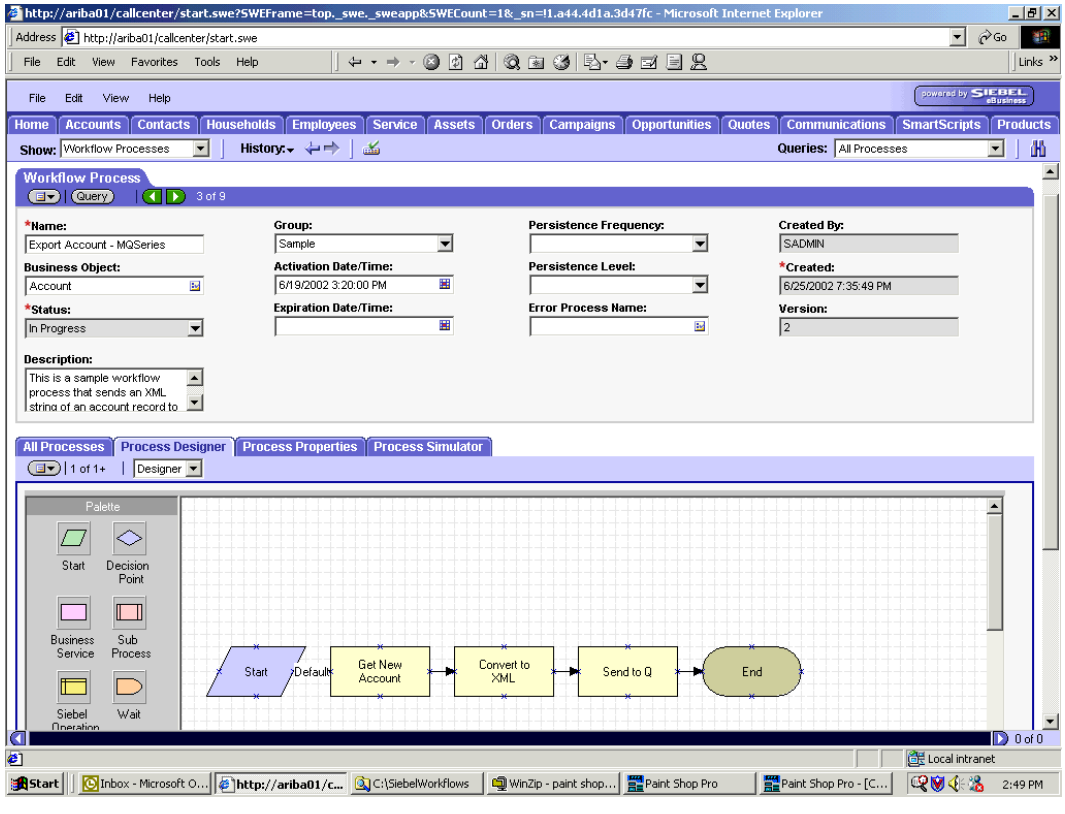
Creating a Siebel Workflow

The following topics include procedures for creating Siebel Workflows in the Siebel Workflow Administration window.

Creating a Siebel Workflow for an Event Using MQSeries Transport

The following procedure is an example of a Siebel Workflow illustrated in the Siebel Workflow Administration window. The Workflow was designed for exporting Siebel Account record information using the MQSeries transport.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The lower pane includes four tabs. The Process Designer tab is active and displays an illustration of a Workflow process. The Workflow process can be modified using the palette to the left of the diagram.



Procedure: How to Create a Siebel Workflow for an Event Using MQSeriesTransport

The following procedure describes how to create a Siebel Workflow that generates Siebel XML when an Account record is updated in the Siebel Call Center application. The Workflow is then placed on an MQSeries message queue.

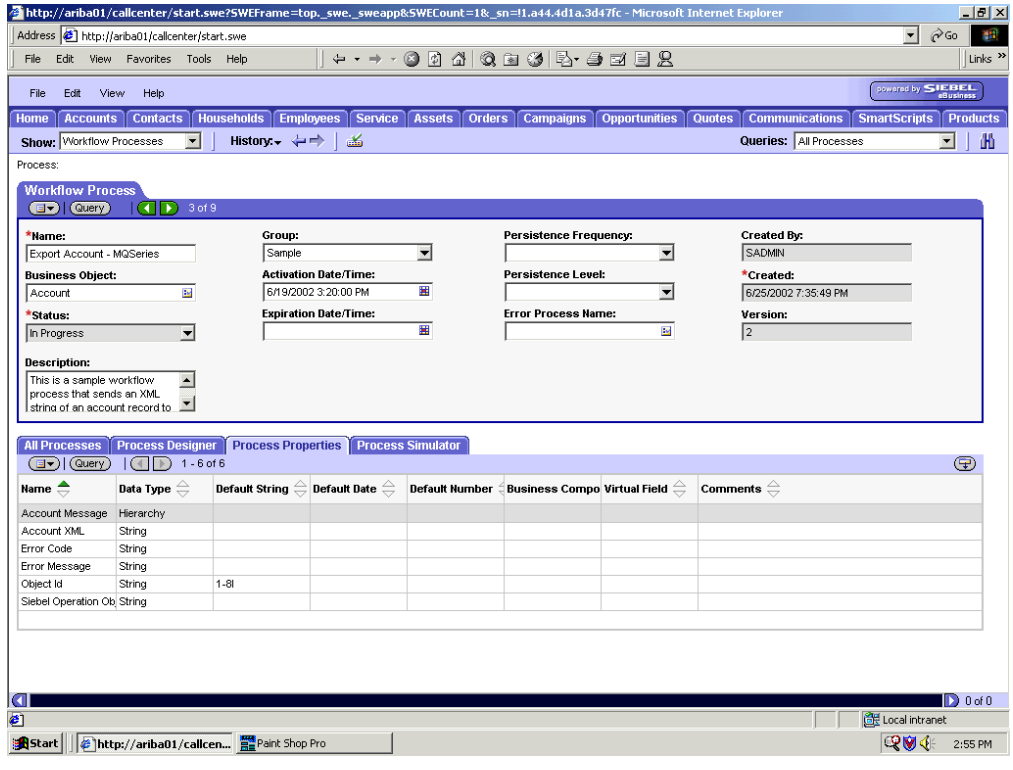
To create a Siebel Workflow:

1. In the Process Properties tab of the Workflow Process window, define the Account message and Account XML process properties.

   The Account message contains Siebel Account data in hierarchical format.

   Account XML specifies the Siebel Account data that the workflow has converted to XML.

   The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account information for a new Workflow. The lower pane includes four tabs. The Process Properties tab is active and includes a chart of Siebel Account data properties.

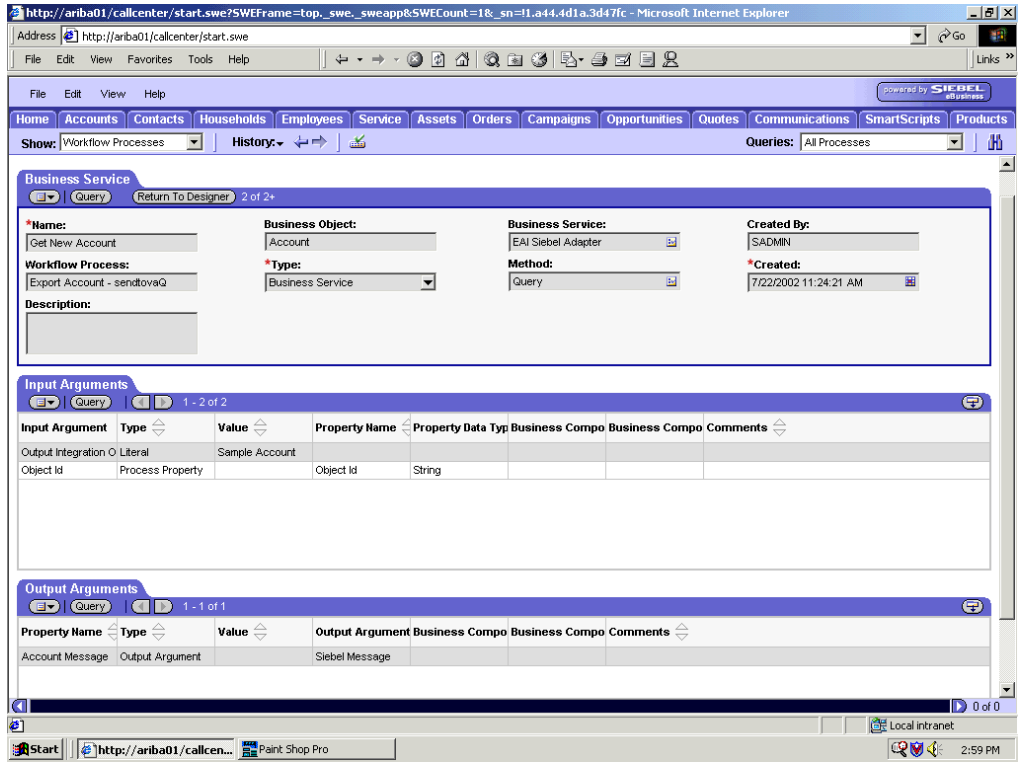


2. Use the Siebel Workflow Administration windows to create a Workflow.
3. Define an EAI Siebel Adapter Business Service step to receive an instance of Account data and call it *Get New Account*.

   Using the Query method, the Business Service obtains the Account information from Siebel.

   Output from this Business Service is generated in hierarchical format.

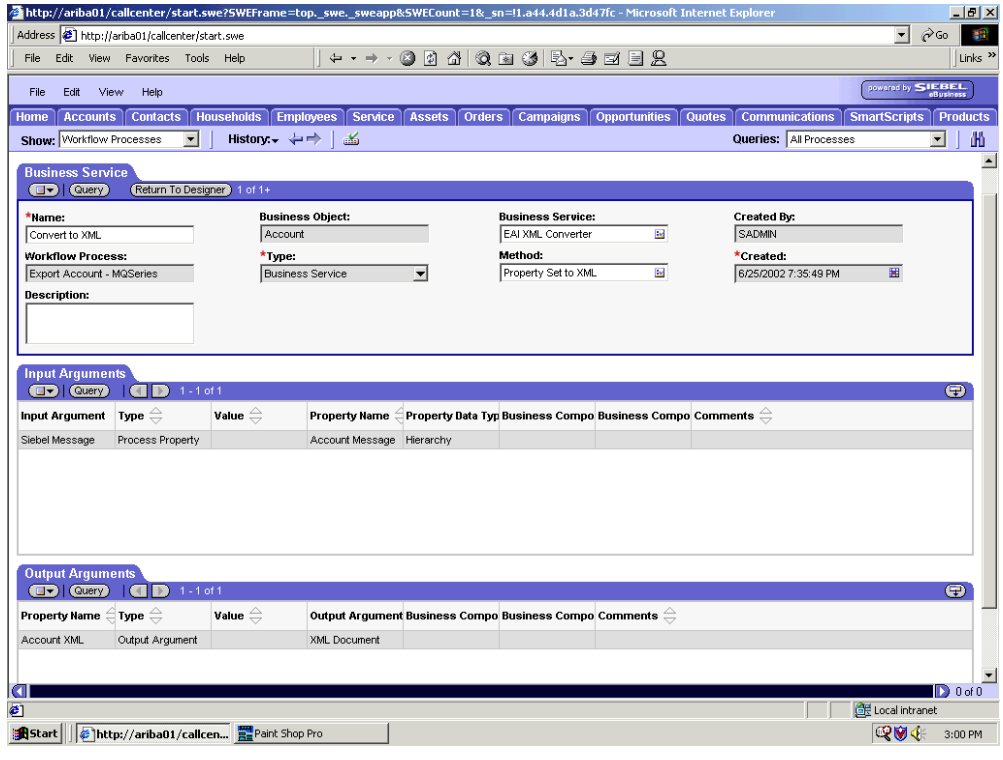
   The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining a new Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



4. Define an EAI XML Converter Business Service step and call it *Convert to XML*.

It is defined to receive the Account data from the EAI Siebel Adapter Business Service in hierarchical format and convert it to XML format.

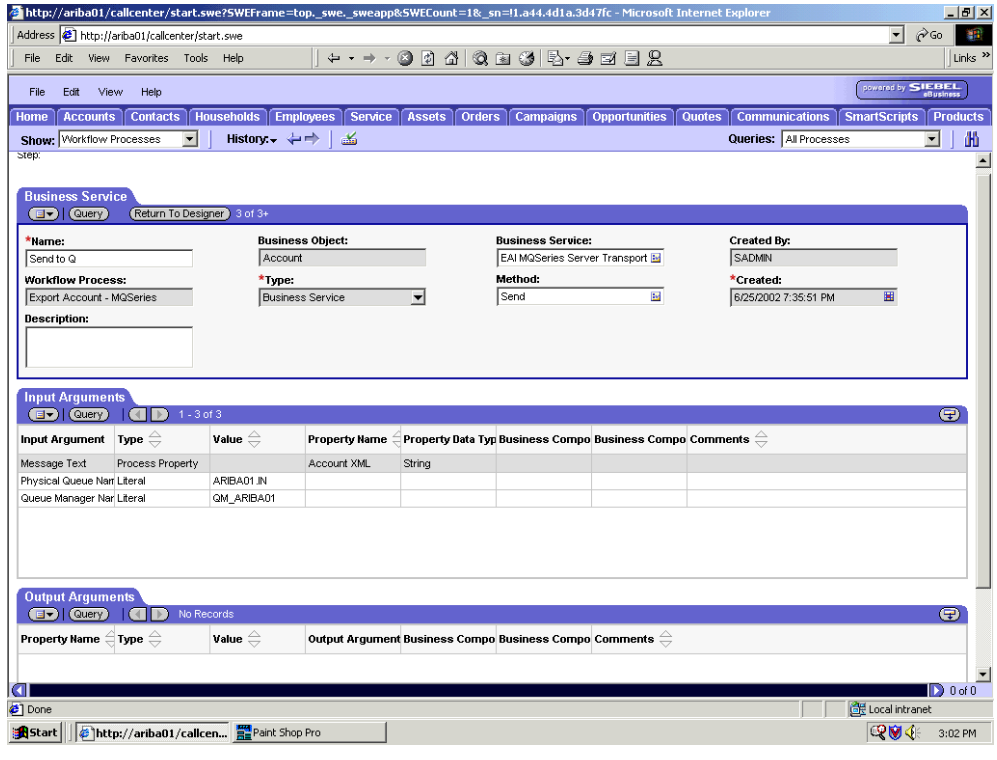
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Convert to XML Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



5. Define an EAI MQSeries server transport Business Service step and call it *Send to Q*.

It is defined to receive the Account data from the EAI XML Converter Business Service in Siebel XML format and send the Account XML to MQSeries using the Send method.

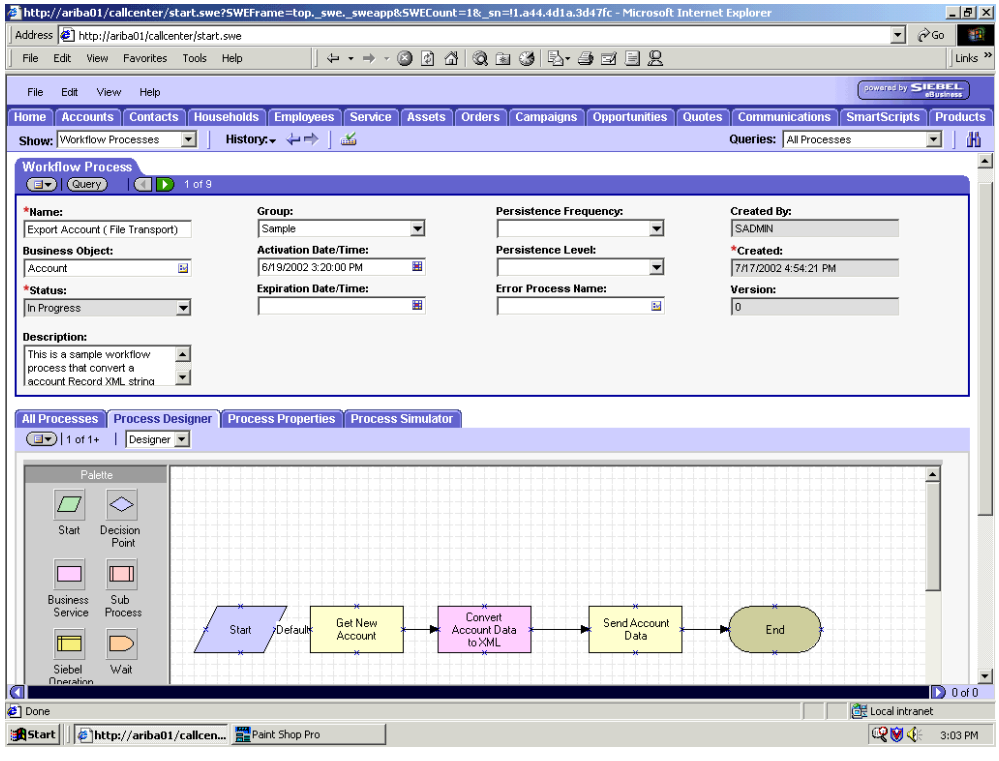
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Send to Q Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



Creating a Siebel Workflow for an Event Using File Transport

The following procedure is an example of a Siebel Workflow illustrated in the Siebel Workflow Administration window. The Workflow is designed for exporting Siebel Account record information using the File transport.

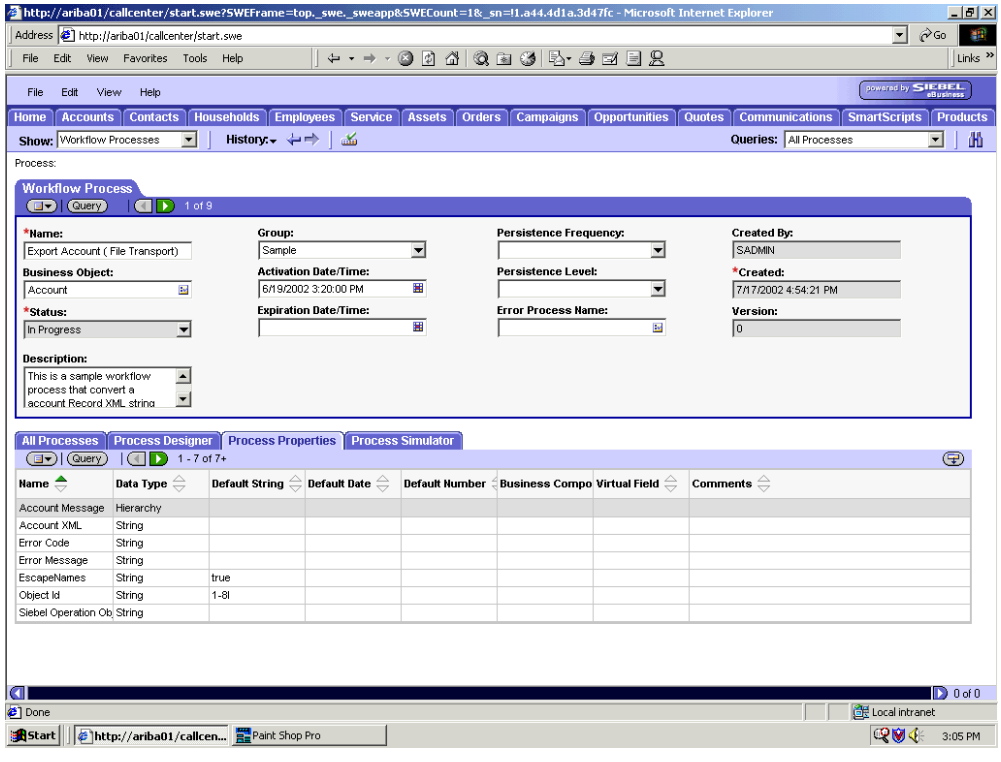
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The lower pane includes four tabs. The Process Designer tab is active and shows an illustration of a Workflow process. The Workflow process can be modified using the palette to the left of the diagram.



Procedure: How to Create a Siebel Workflow for an Event Using File Transport

The following procedure describes how to create a Siebel Workflow that generates Siebel XML when an Account record is updated in the Siebel Call Center application and then places Siebel XML on the file system.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The lower pane includes four tabs. The Process Properties tab is active and includes a chart of Siebel Account data properties.



To create a Siebel Workflow:

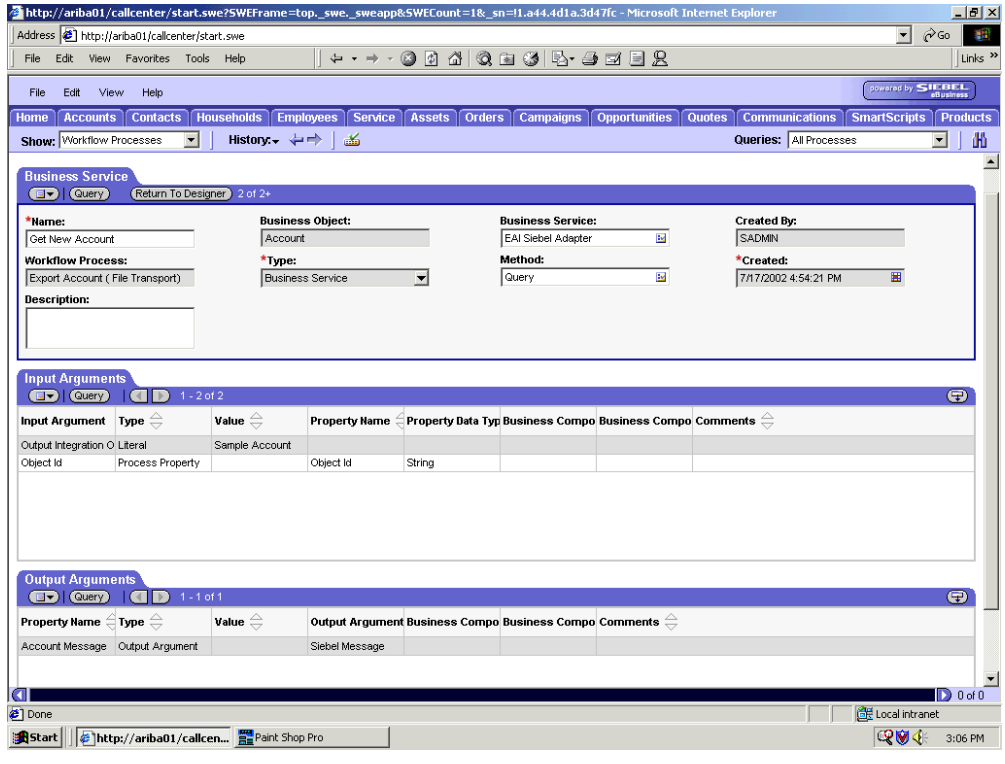
1. On the Process Properties tab of the Workflow Process window, define the Account message and Account XML process properties.

   Account message contains the Siebel Account data in hierarchical format.

   Account XML specifies which Siebel Account data the Workflow converted to XML.

2. Use the Siebel Workflow Administration windows to create a Workflow.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Get New Account Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.

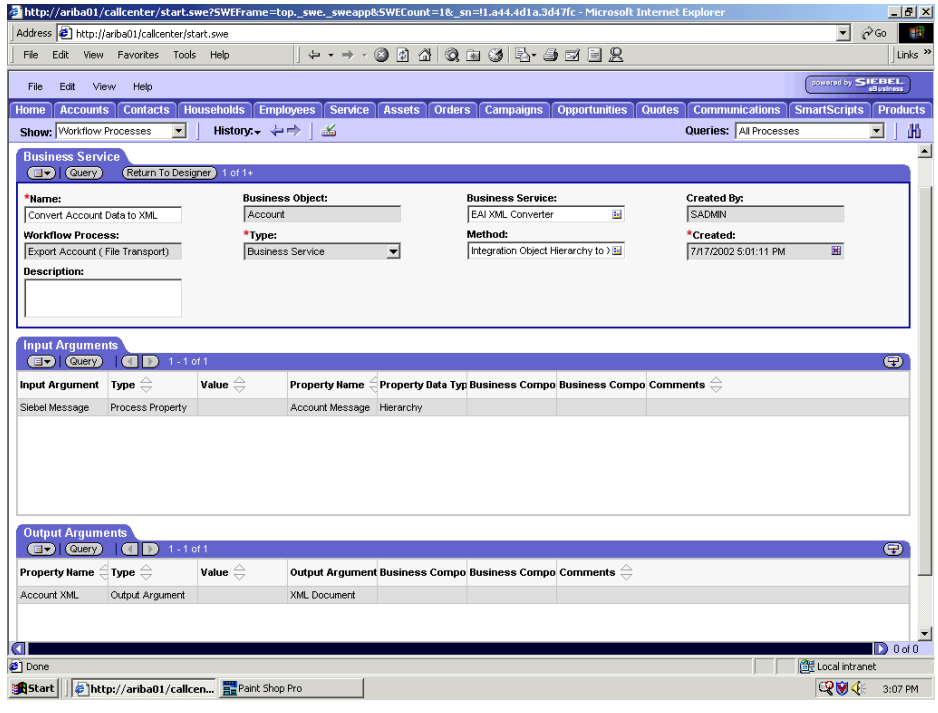


3. Define an EAI Siebel Adapter Business Service step to receive an instance of Account data and call it *Get New Account*.

Using the Query method, the Business Service obtains the Account information from Siebel.

Output from this Business Service is generated in hierarchical format.

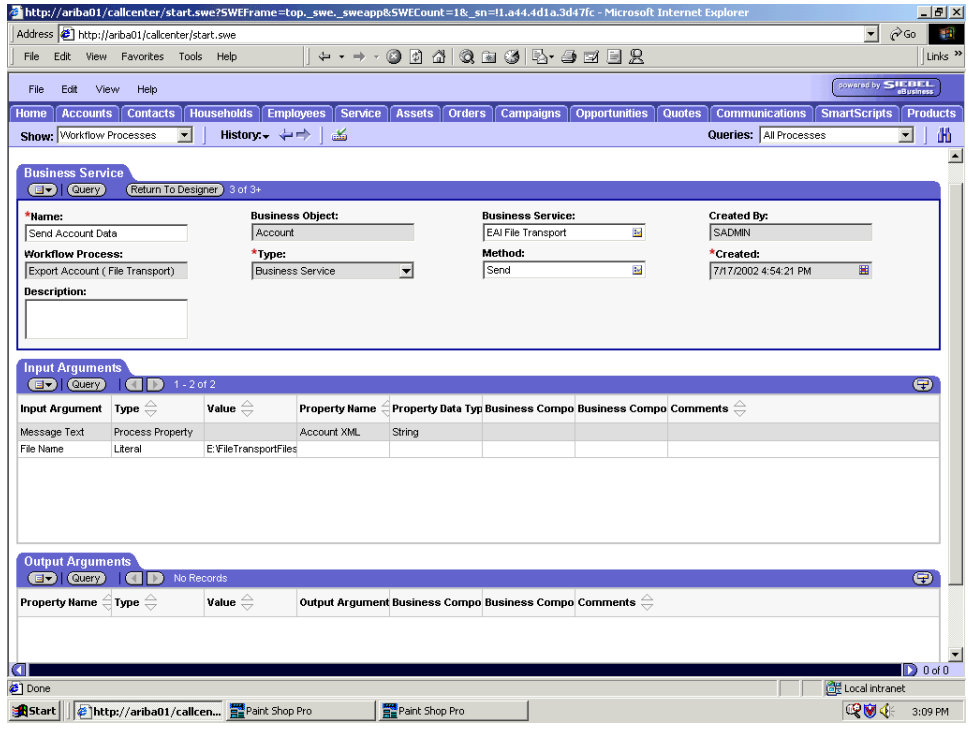
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Convert Account Data to XML Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



4. Define an EAI XML Converter Business Service step and call it *Convert Account Data to XML*.

This Business Service is defined to receive the Account data from the EAI Siebel Adapter Business Service in hierarchical format and convert it to XML format.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Send Account Data Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



5. Define an EAI File transport Business Service step and call it *Send Account Data*.

This Business Service is defined to receive the Account data from the EAI XML Converter Business Service in Siebel XML format and send the Account XML to the file system in a specified directory using the Send method.

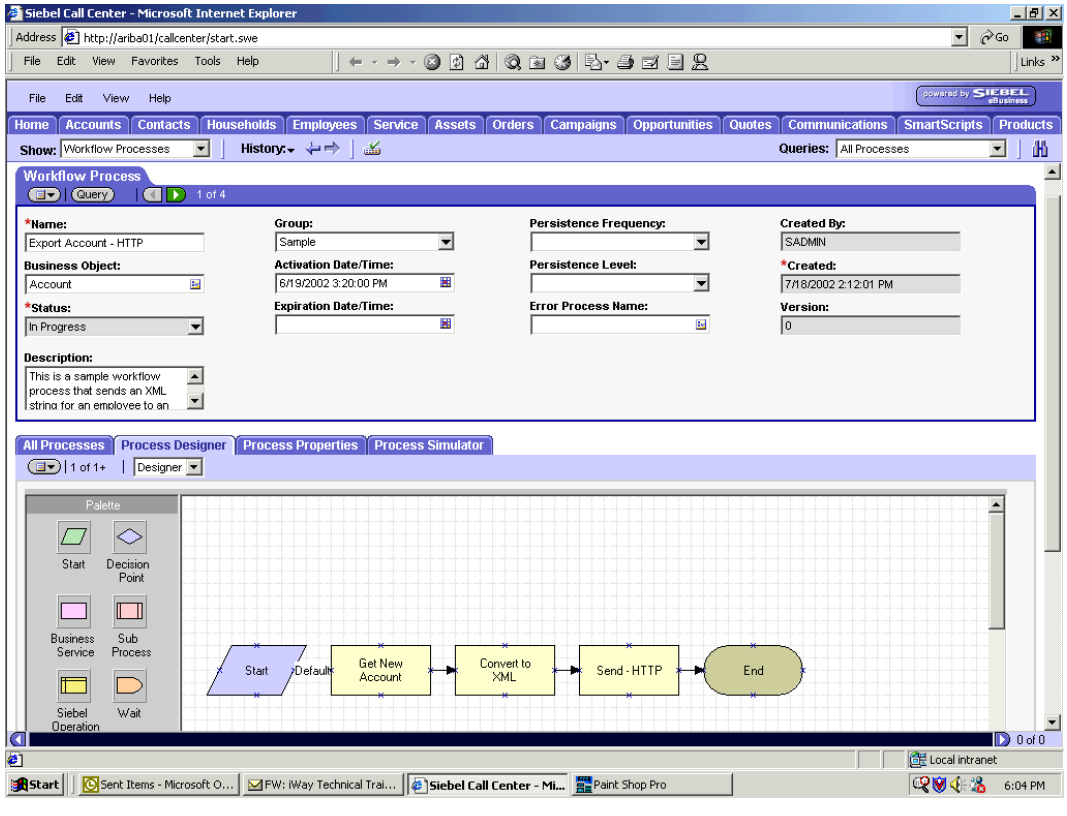
Creating a Siebel Workflow for an Event Using HTTP Transport

The following procedure is an example of a Siebel Workflow illustrated in the Siebel Workflow Administration window. The Workflow was designed for exporting Siebel Account record information using the HTTP transport.

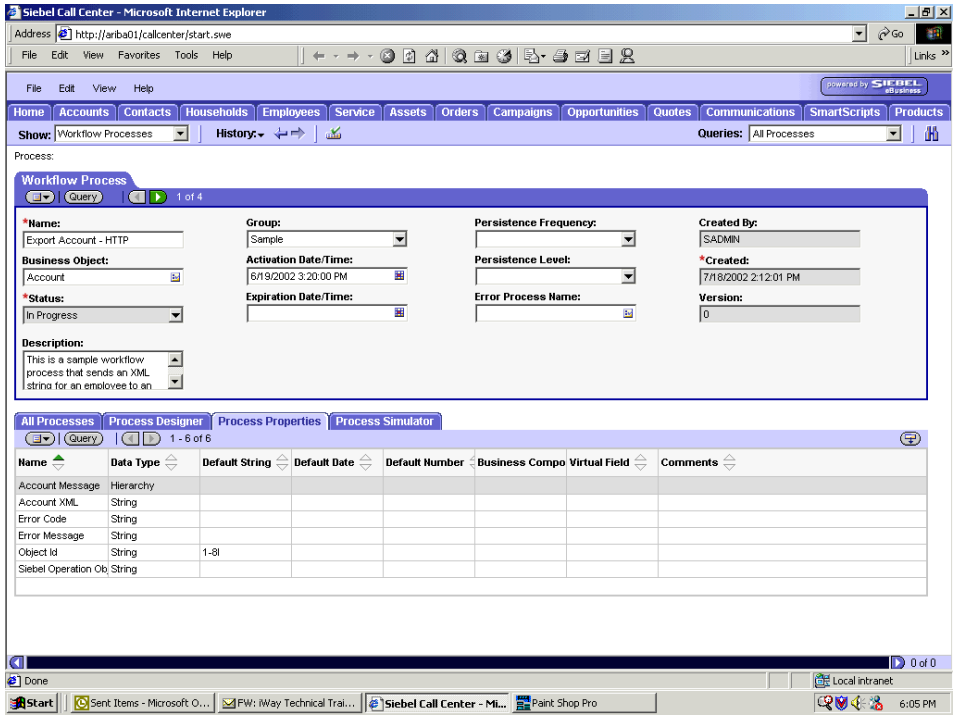
Procedure: How to Create a Siebel Workflow for an Event Using HTTP Transport

The following procedure describes how to create a Siebel Workflow that generates Siebel XML when an Account record is updated in the Siebel Call Center application.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The lower pane includes four tabs. The Process Designer tab is active and displays an illustration of a Workflow process. The Workflow process can be modified using the palette to the left of the diagram.



The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The lower pane includes four tabs. The Process Properties tab is active and includes a chart of Siebel Account data properties.



To create a Siebel Workflow:

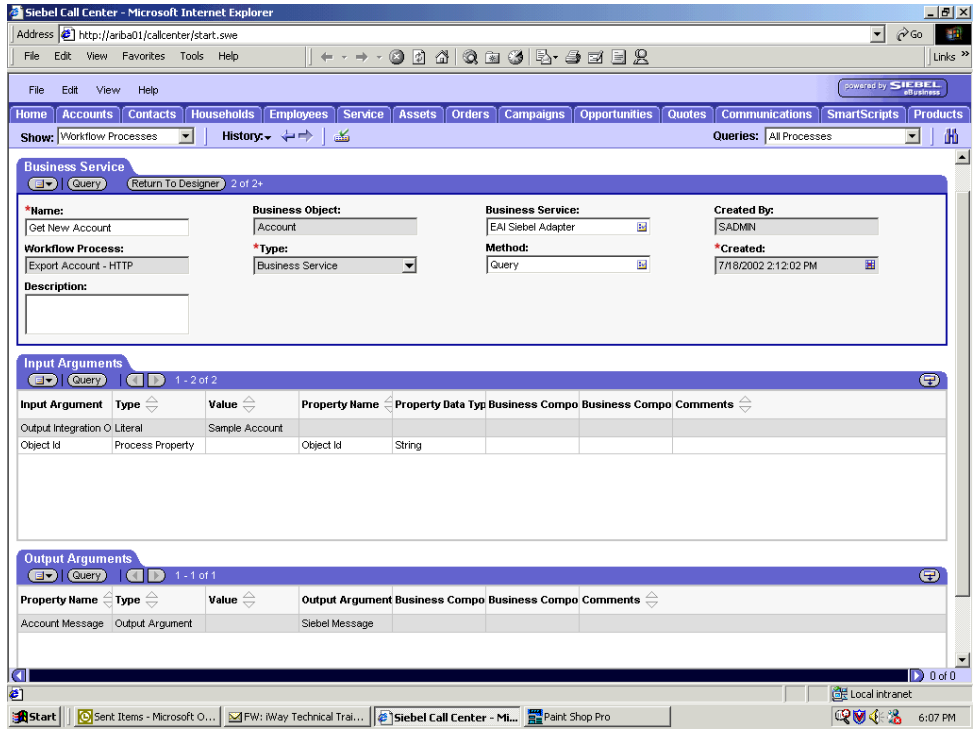
1. In the Process Properties tab of the Workflow Process window, define the Account message and Account XML process properties.

   Account message contains the Siebel Account data in hierarchical format.

   Account XML specifies the Siebel Account data that the Workflow has converted to XML.

2. Use the Siebel Workflow Administration windows to create a Workflow.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Get New Account Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.

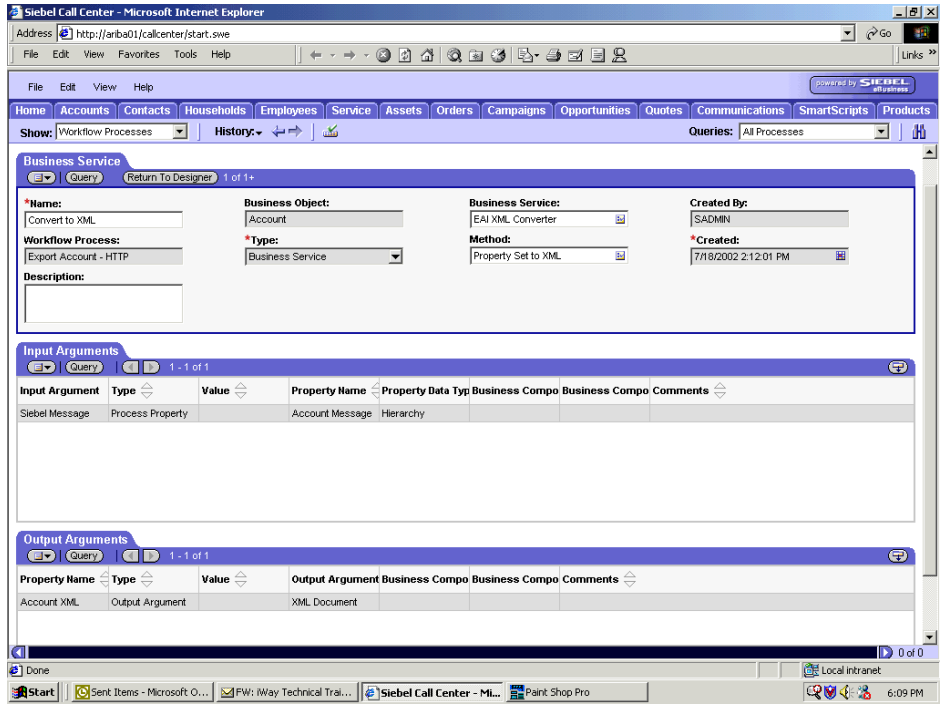


3. Define an EAI Siebel Adapter Business Service step to receive an instance of Account data and call it *Get New Account*.

Using the Query method, the Business Service obtains the Account information from Siebel.

Output from this Business Service is generated in hierarchical format.

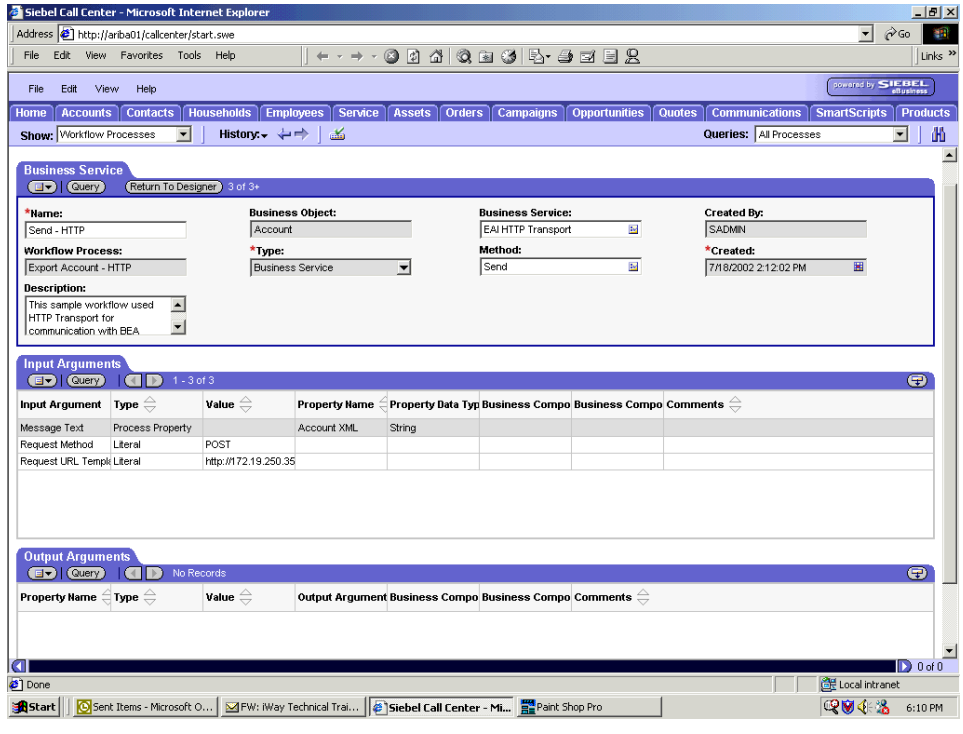
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Convert to XML Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



4. Define an EAI XML Converter Business Service step and call it *Convert to XML*.

This Business Service is defined to receive the Account data from the EAI Siebel Adapter Business Service in hierarchical format and convert it to XML format.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Send - HTTP Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



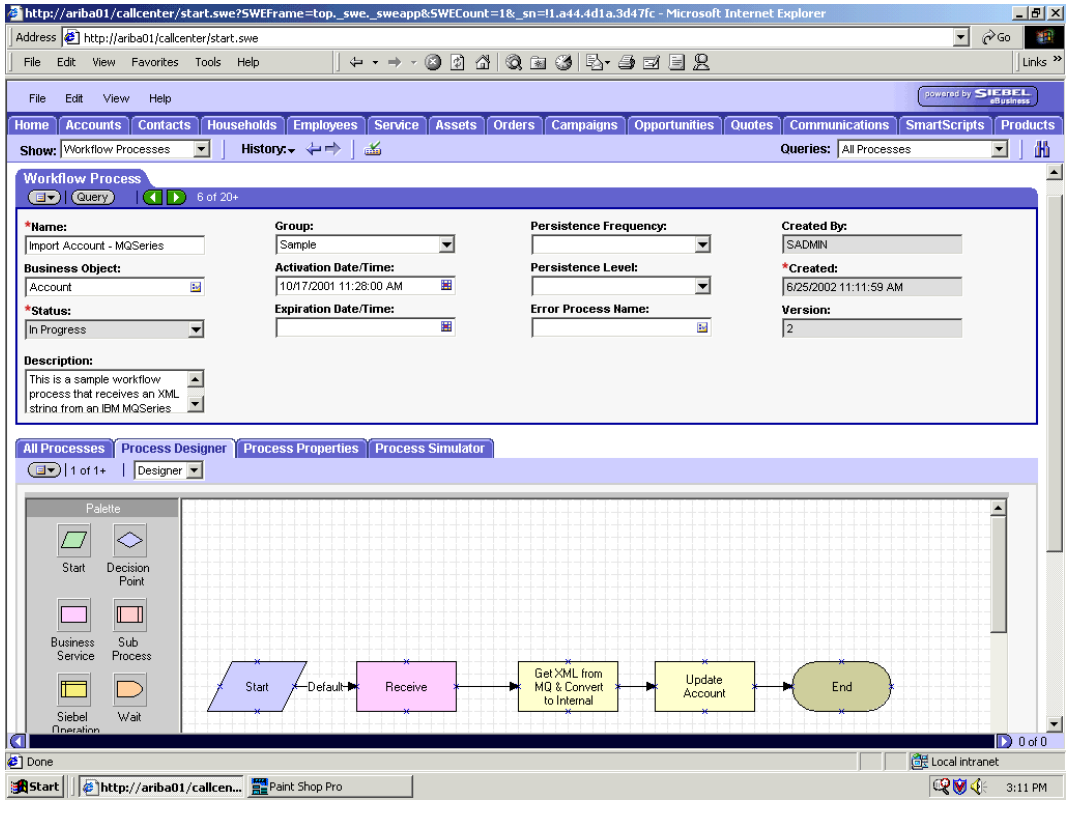
5. Define an EAI HTTP Transport Business Service step and call it *Send - HTTP*.

This Business Service is defined to receive the Account data from the EAI XML Converter Business Service in Siebel XML format and send the Account XML to HTTP using the Send method.

Creating a Siebel Workflow for a Service Using MQSeries Transport

The following procedure is an example of a Siebel Workflow illustrated in the Siebel Workflow Administration window. The Workflow was designed for importing Siebel Account record information through the MQSeries Transport.

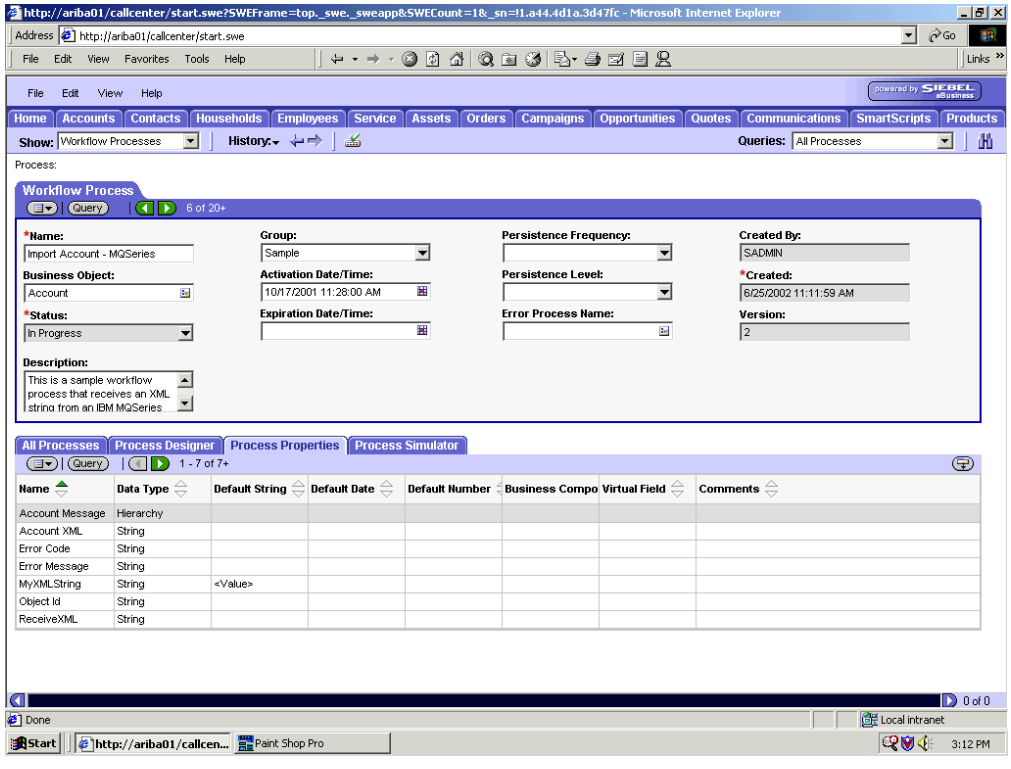
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The Process Designer tab is active and shows an illustration of a Workflow process. The Workflow process can be modified using the palette to the left of the diagram.



Procedure: How to Create a Siebel Workflow for a Service Using MQSeries Transport

The following procedure describes how to create a Siebel Workflow that generates Siebel XML when an Account record is updated in the Siebel Call Center application.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The lower pane includes four tabs. The Process Properties tab is active and includes a chart of Siebel Account data properties.



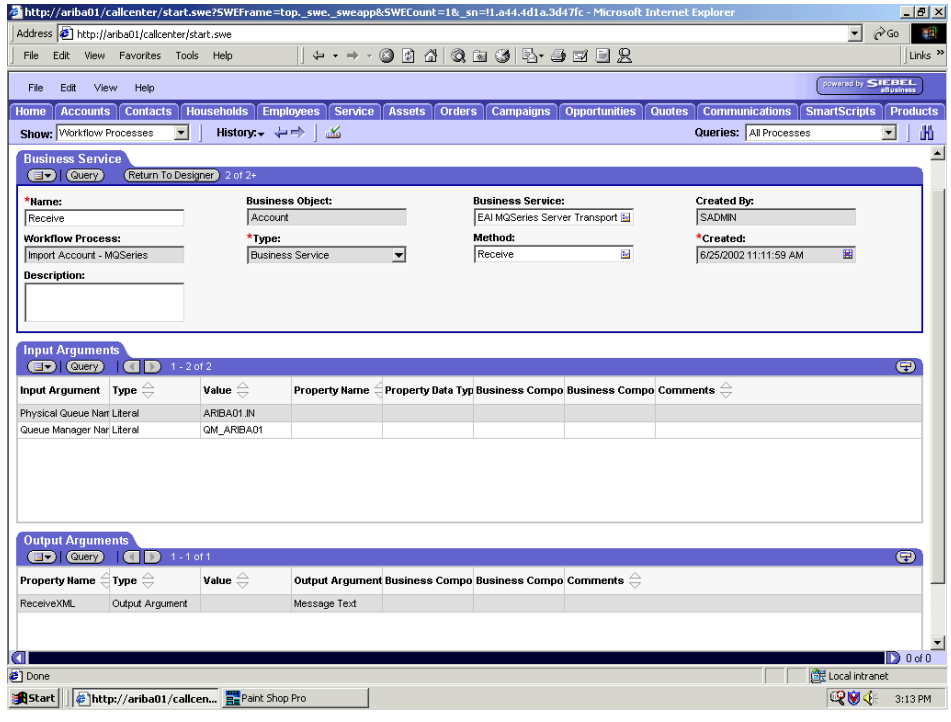
To create a Siebel Workflow:

1. In the Process Properties tab of the Workflow Process window, define the Account message and Account XML process properties.

   Account message contains the Siebel Account data in hierarchical format.

   Account XML specifies the Siebel Account data that the Workflow converted to XML.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Receive Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.

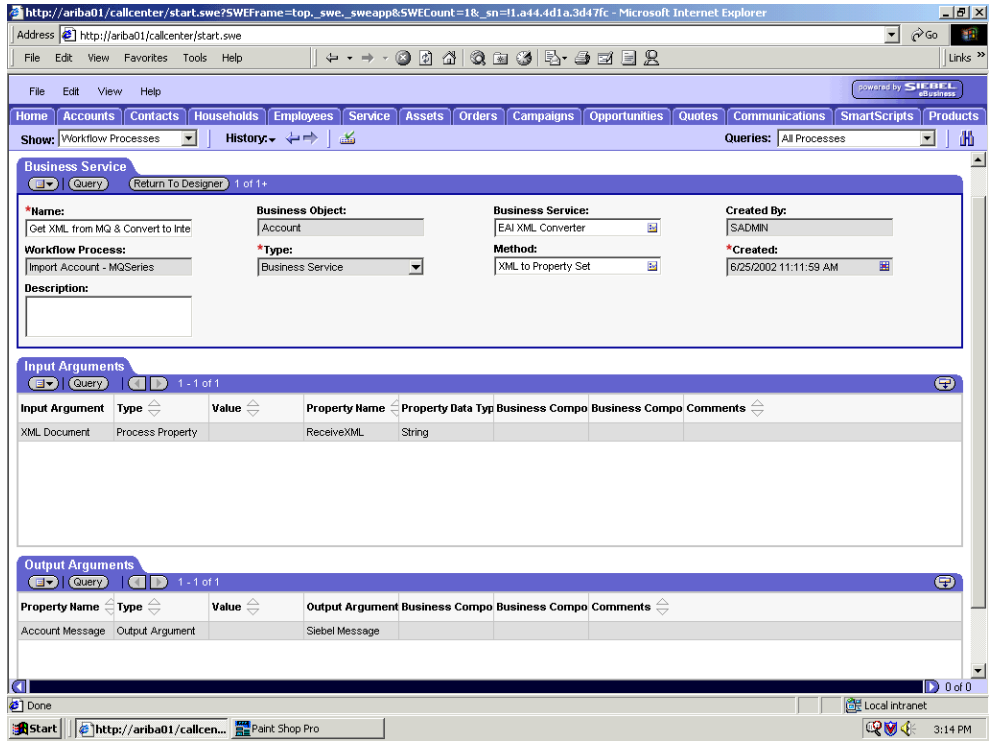


2. Define an EAI MQSeries Server Transport Business Service step and call it *Receive*.

The Business Service is defined to receive the Account data from the MQSeries message queue.

The EAI MQSeries Server Transport Business Service receives the Account data in Siebel XML format and sends it to the EAI XML Converter Business Service.

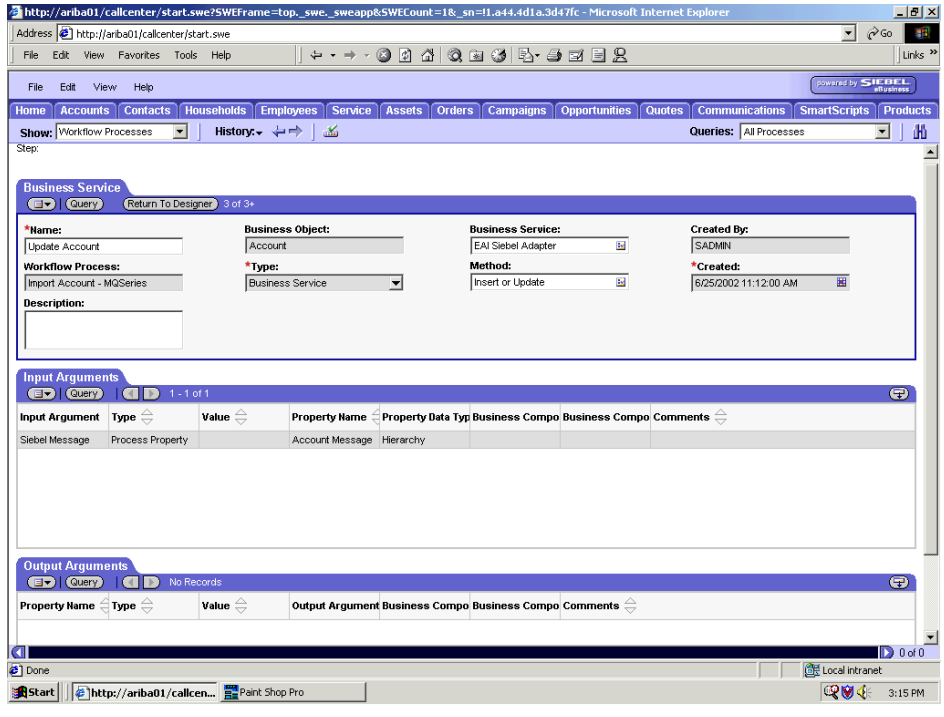
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Get XML from MQ & Convert to XML Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



3. Define an EAI XML Converter Business Service step and call it *Get XML from MQ & Convert to XML*.

This Business Service is defined to receive the Account data from the EAI MQSeries Server Transport Business Service in XML format and convert it to hierarchical format.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Update Account Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



4. Define an EAI Siebel Adapter Business Service step and call it *Update Account*.

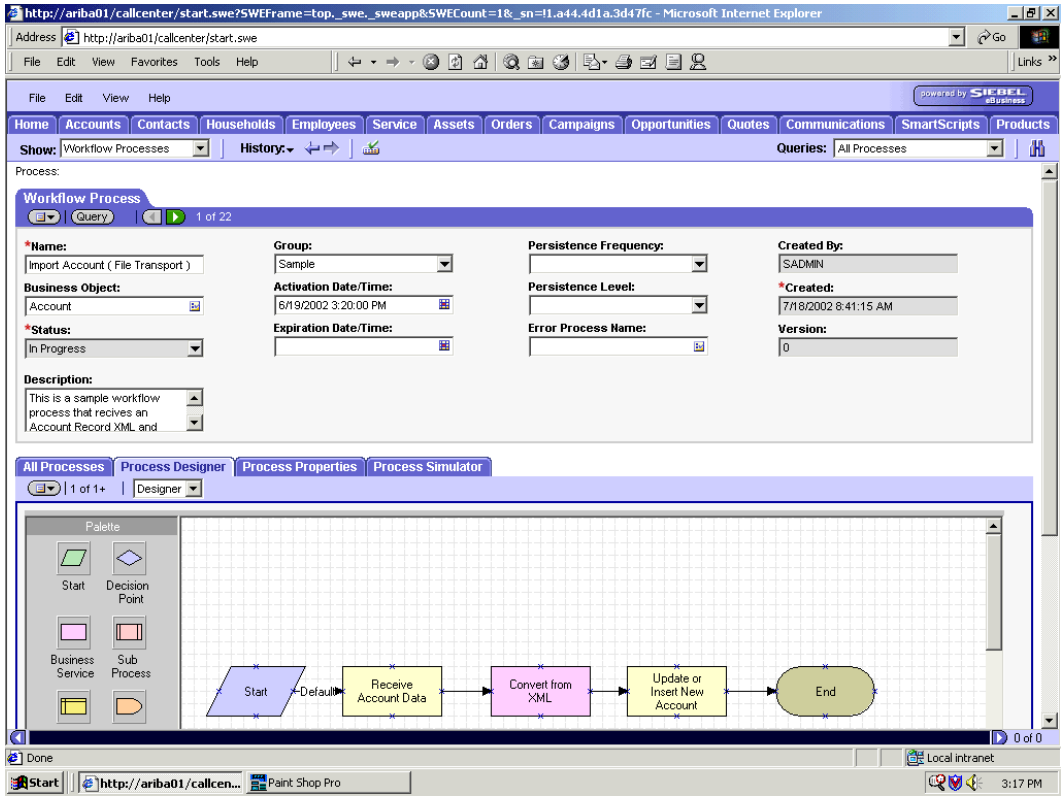
This Business Service is defined to receive from the EAI XML Converter Business Service the instance of Account data in hierarchical format.

The Business Service applies the Account information into Siebel using the Insert or Update method.

Creating a Siebel Workflow for a Service Using File Transport

The following procedure is an example of a Siebel Workflow illustrated in the Siebel Workflow Administration window. The workflow was designed for importing Siebel Account record information through the File transport.

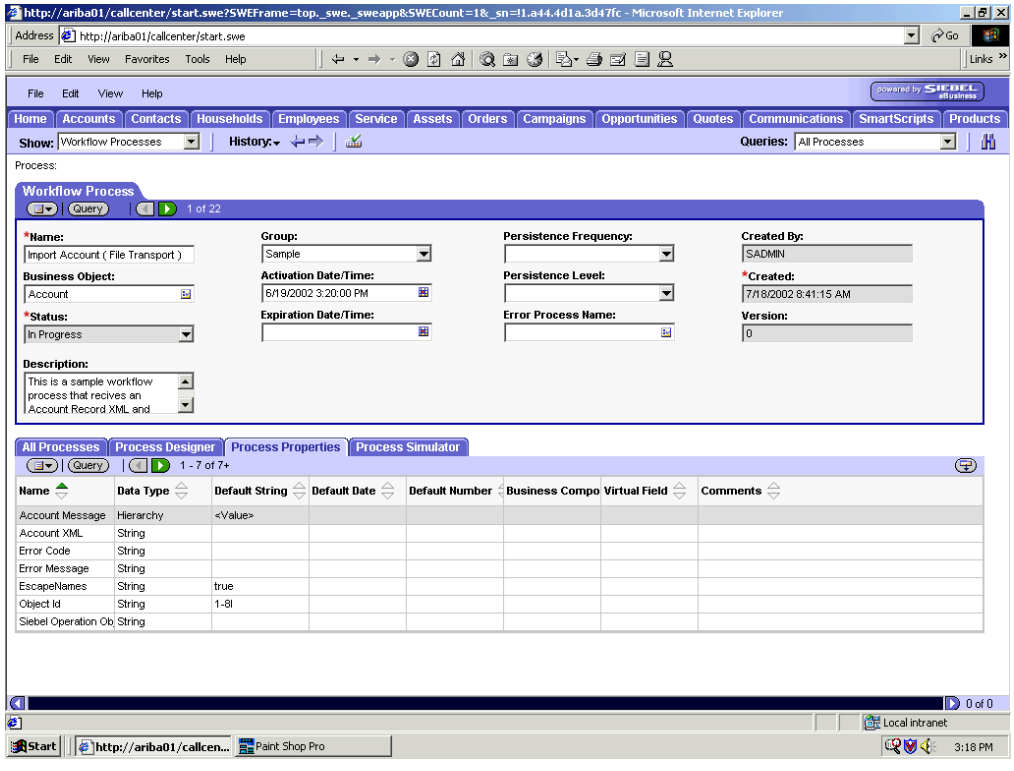
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The lower pane includes four tabs. The Process Designer tab is active and displays an illustration of a Workflow process. The Workflow process can be modified using the palette to the left of the diagram.



Procedure: How to Create a Siebel Workflow for a Service Using File Transport

The following procedure describes how to create a Siebel Workflow that generates Siebel XML when an Account record is updated in the Siebel Call Center application and then places Siebel XML on the file system.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The lower pane includes four tabs. The Process Properties tab is active and includes a chart of Siebel Account data properties.



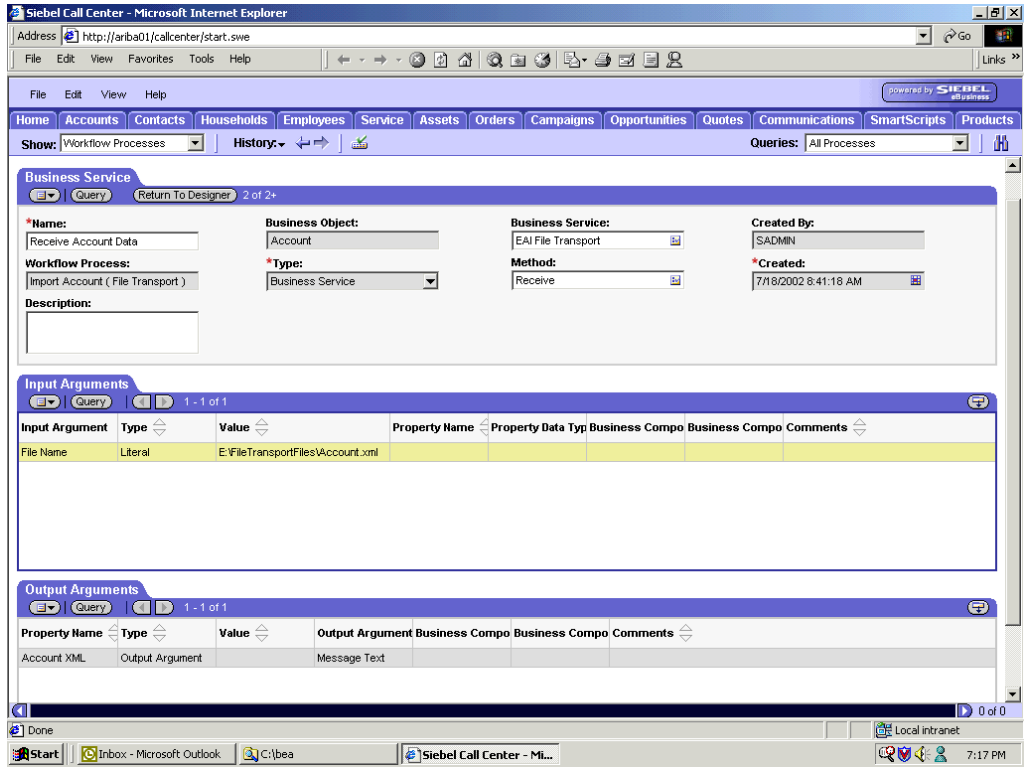
To create a Siebel Workflow:

1. In the Process Properties tab of the Workflow Process window, define the Account message and Account XML process properties.

   Account message contains the Siebel Account data in hierarchical format.

   Account XML specifies the Siebel Account data that the workflow converted to XML.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Receive Account Data Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.

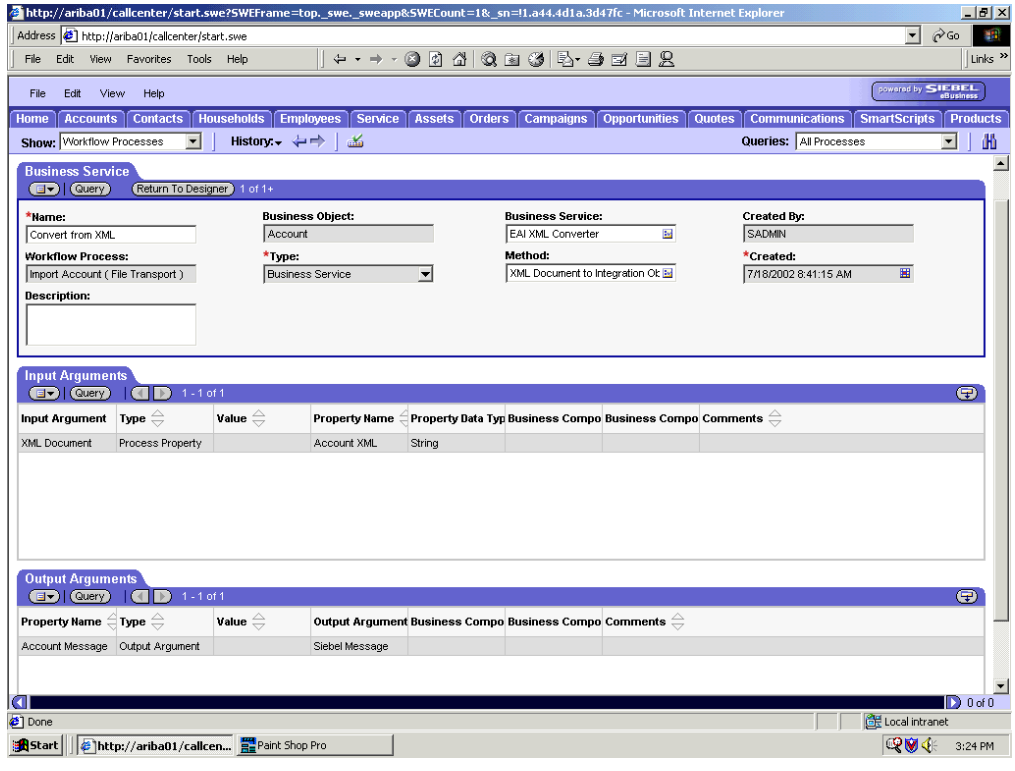


2. Define an EAI FileTransport Business Service step and call it *Receive Account Data*.

The Business Service is defined to receive the Account data from the file system.

The EAI File Transport Business Service receives the Account data in Siebel XML format and sends it to the EAI XML Converter Business Service.

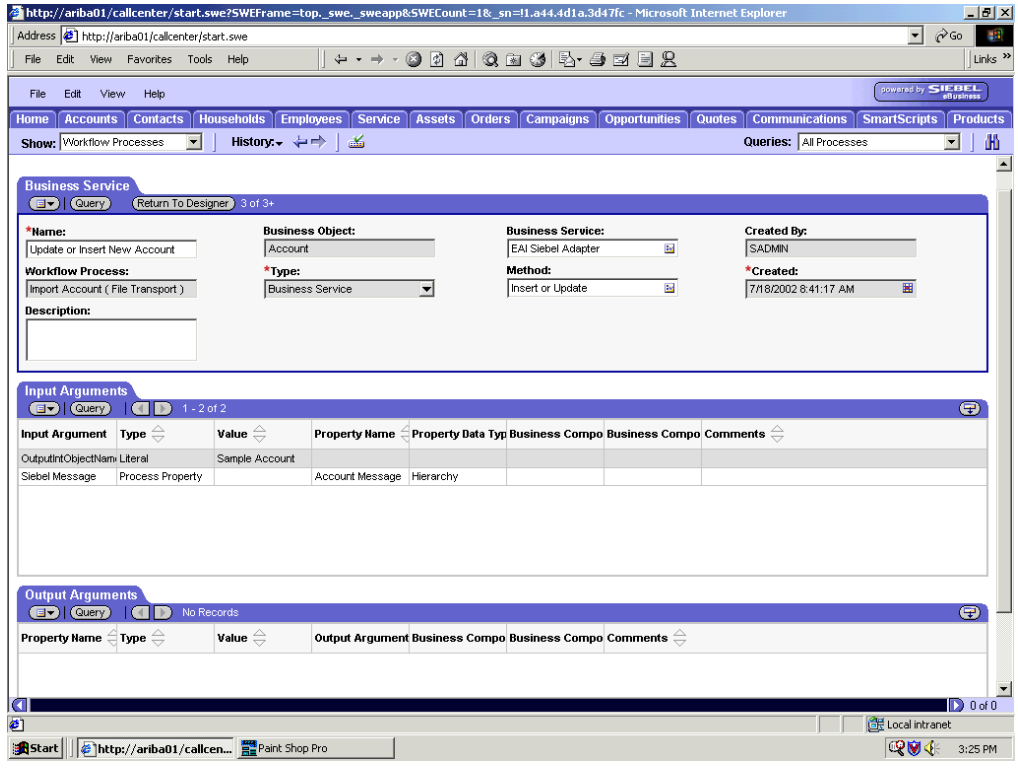
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Convert from XML Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



3. Define an EAI XML Converter Business Service step and call it *Convert from XML*.

   This Business Service is defined to receive the Account data from the EAI File Transport Business Service in XML format and convert it to hierarchical format.

   The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Update or Insert New Account Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



4. Define an EAI Siebel Adapter Business Service step and call it *Update or Insert New Account*.

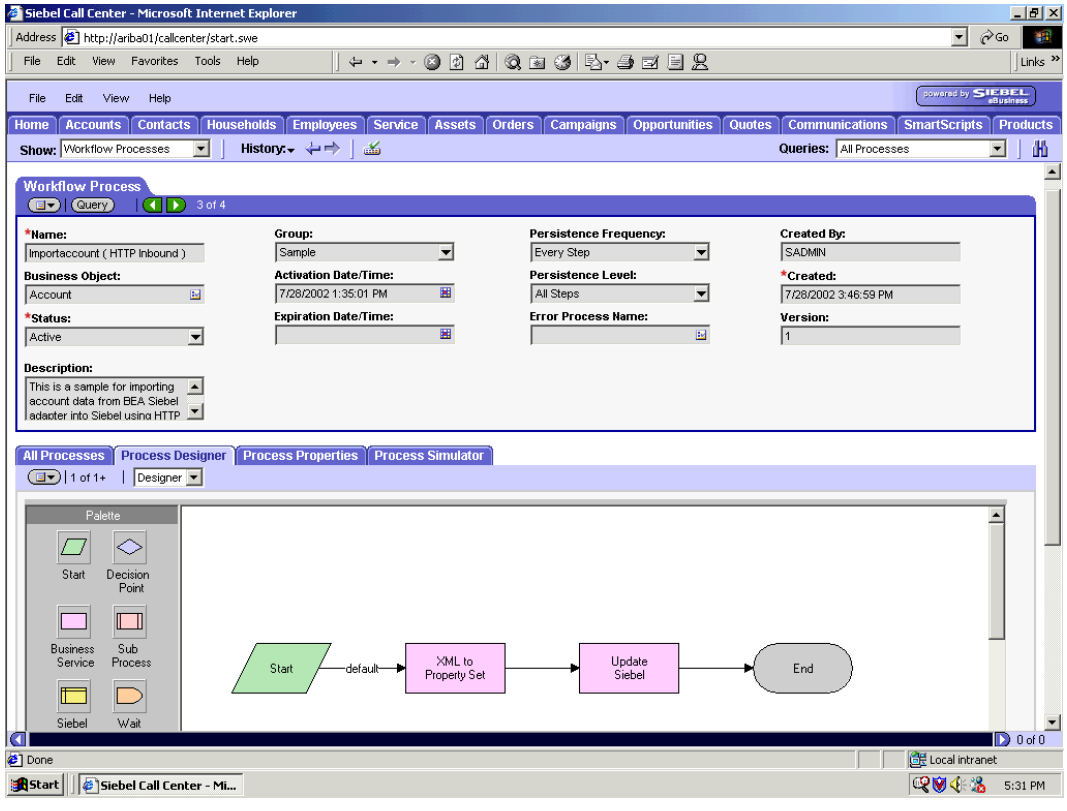
   This Business Service is defined to receive from the EAI XML Converter Business Service the instance of Account data in hierarchical format.

   The Business Service applies the Account information into Siebel using the Insert or Update method.

Creating a Siebel Workflow for a Service Using HTTP Transport

The following procedure is an example of a Siebel workflow illustrated in the Siebel Workflow Administration window. The Workflow was designed for importing Siebel Account record information through the HTTP transport.

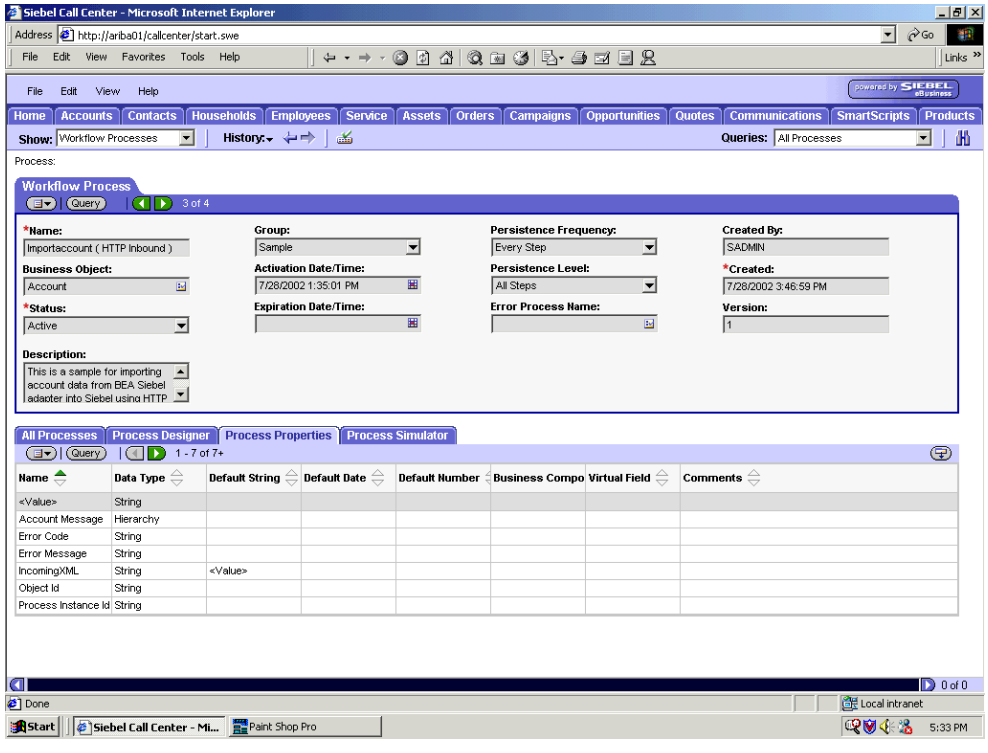
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The lower pane includes four tabs. The Process Designer tab is active and displays an illustration of a Workflow process. The Workflow process can be modified using the palette to the left of the diagram.



Procedure: How to Create a Siebel Workflow for a Service Using HTTP Transport

The following procedure describes how to create a Siebel Workflow that generates Siebel XML when an Account record is updated in the Siebel Call Center application and then places Siebel XML on the file system.

The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Workflow Process tab, which includes several fields and drop-down lists for defining Account record information for each Workflow. The lower pane includes four tabs. The Process Properties tab is active and includes a chart of Siebel Account data properties.



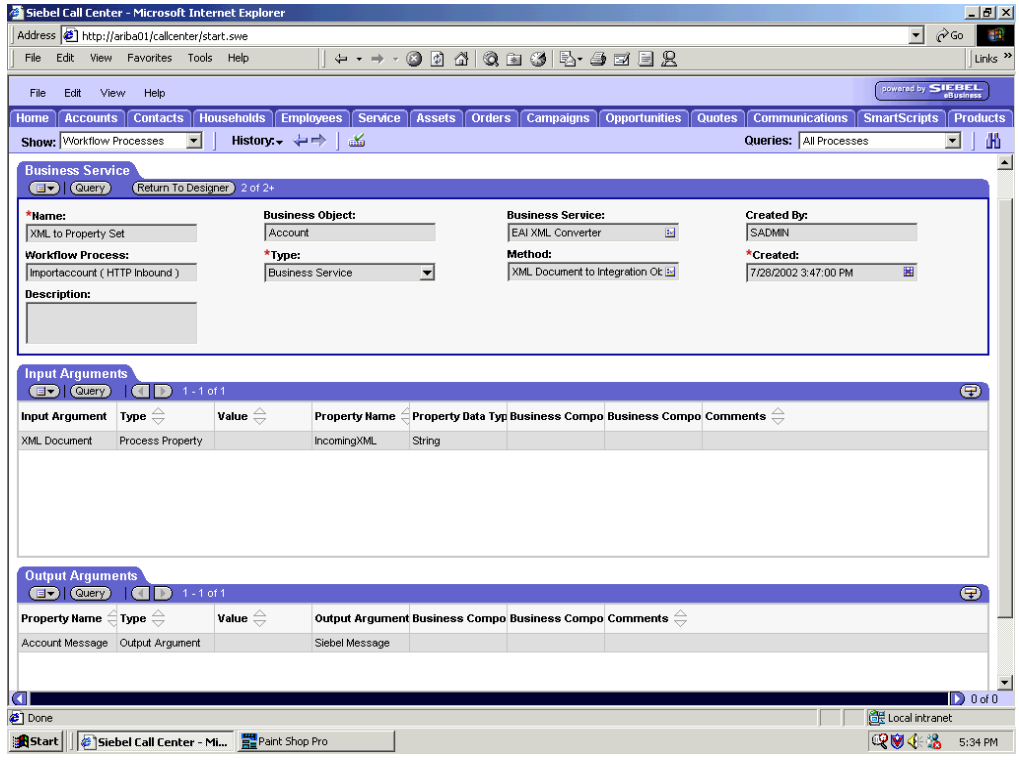
To create a Siebel Workflow:

1. In the Process Properties tab of the Workflow Process window, define the Account message and Account XML process properties.

   Account message contains the Siebel Account data in hierarchical format.

   Account XML specifies the Siebel Account data that the workflow converted to XML.

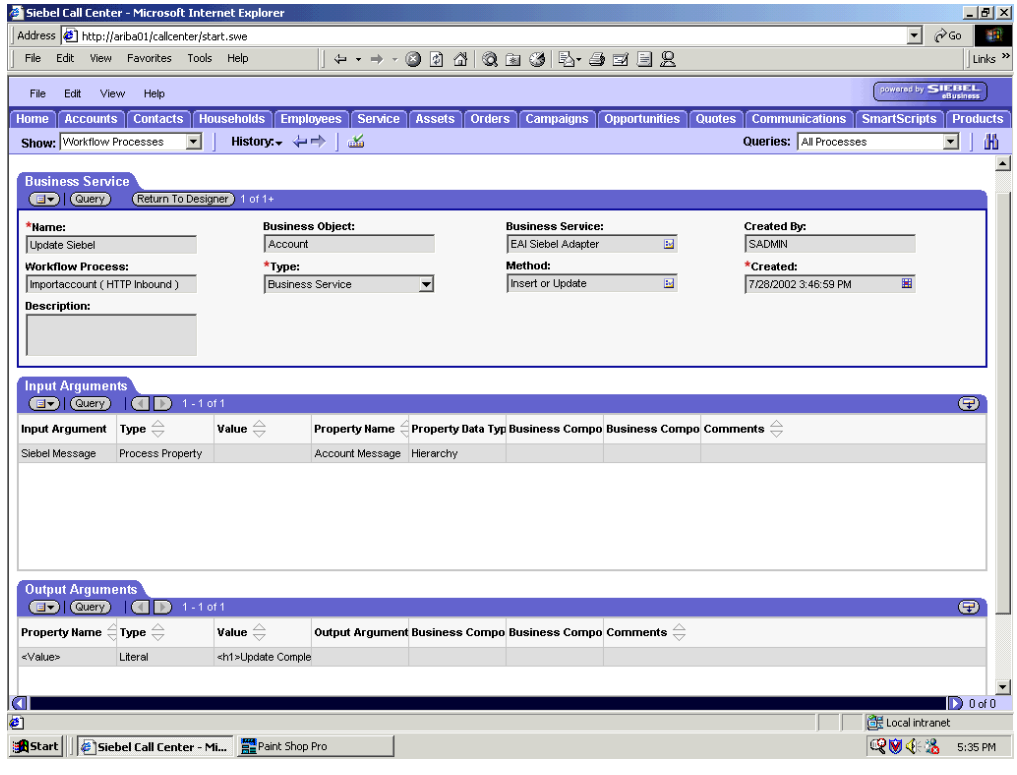
The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the XML to Property Set Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



2. Define an EAI XML Converter Business Service step and call it *XML to Property Set*.

   The Business Service is defined to receive the Account data from the EAI HTTP Transport Business Service in XML format and convert it to hierarchical format.

   The following image shows the Siebel Workflow Administration window. The window includes fourteen tabs across the top, a Show drop-down list in the upper left, and a Queries drop-down list in the upper right. The upper pane shows the Business Service tab, which includes several fields and drop-down lists for defining the Update Siebel Business Service step. The middle pane shows the Input Arguments tab, which includes a chart of Input Arguments. The lower pane shows the Output Arguments tab, which includes a chart of Output Argument properties.



3. Define an EAI Siebel Adapter Business Service step and call it *Update Siebel*.

   The Business Service is defined to receive from the EAI XML Converter Business Service the instance of Account data in hierarchical format.

   The Business Service applies the Account information into Siebel using the Insert or Update method.

APPENDIX C

Best Practices and Performance Considerations

Topics:

- Overview of the BEA WebLogic Adapter for Siebel
- Supported Tools and Technologies
- Understanding Web Services and Java Connector Architecture Functionality
- Understanding JCA Managed and Non-managed Modes
- Application Design
- Configuration Tuning

This topic provides best practices and performance considerations for the BEA WebLogic Adapter for Siebel when deployed to the BEA WebLogic Server.

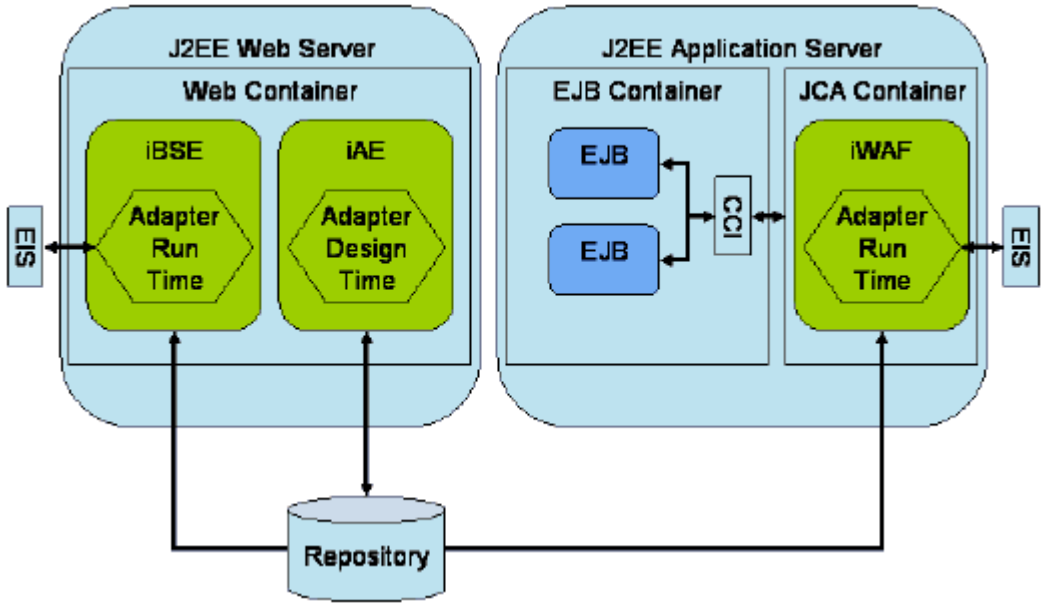
Overview of the BEA WebLogic Adapter for Siebel

The BEA WebLogic Adapter for Siebel provides a means to exchange real-time business data between Siebel systems and other application, database, or external business partner systems. The adapter enables external applications for inbound and outbound processing with Siebel.

The adapter uses XML messages to enable non-Siebel applications to communicate and exchange transactions with Siebel using services and events.

Adapter Framework

The following diagram illustrates the BEA WebLogic Adapter for Siebel framework.



The Integration Business Services Engine (iBSE) and Application Explorer (iAE) are deployed to the J2EE Web container. The Connector for JCA is deployed to the JCA container in the J2EE server. The repository is written at design time and read at run time. The container (iAE, iWAF, or iBSE) is responsible for the connection to the repository as shown.

Integrating With Siebel

You can use the BEA WebLogic Adapter for Siebel to invoke a Siebel business process, such as add/update account, or you can use the adapter as part of an integration effort to connect Siebel and non-Siebel systems. The BEA WebLogic Adapter for Siebel is bidirectional and can detect an event from Siebel by receiving a Siebel XML document emitted by Siebel.

When integrating with Siebel using Siebel XML documents, the adapter application developer can use existing Siebel Integration Objects or create new Siebel Integration Objects to use within a Siebel Workflow. The Workflow processes inbound or outbound Siebel XML and uses various transports such as MQSeries, File, and HTTP to exchange transactions with external systems. The Siebel Workflow is usually created by the Siebel administrator or developer using Siebel Workflow Administration screens.

When integrating with Siebel directly using the Java™ Data Bean or COM Data Interface, the BEA WebLogic Adapter for Siebel does not require a Siebel Integration Object or Siebel Workflow. Instead, it executes Siebel Business Services and Siebel Business Components directly.

Supported Tools and Technologies

The BEA WebLogic Adapter for Siebel works in conjunction with the following components:

- Application Explorer
- Integration Business Services Engine (iBSE)
- Enterprise Connector for J2EE™ Connector Architecture (JCA)

Application Explorer

Application Explorer uses an explorer metaphor to browse the Siebel system for metadata. The explorer enables you to create XML schemas and Web services for the associated object. In addition, you can create ports and channels to listen for events in your Siebel application. External applications that access Siebel through the BEA WebLogic Adapter for Siebel use either XML schemas or Web services to pass data between the external application and the adapter.

Integration Business Services Engine

The Integration Business Services Engine (iBSE) exposes—as Web services—enterprise assets that are accessible from adapters regardless of the programming language or the particular operating system.

iBSE simplifies the creation and execution of Web services when running:

- Custom and legacy applications

- Database queries and stored procedures
- Packaged applications
- Terminal emulation and screen-based systems
- Transactional systems

Web services is a distributed programming architecture that solves Enterprise Application Integration (EAI) hurdles that other programming models cannot. It enables programs to communicate with one another using a text-based, platform- and language-independent message format called XML.

Coupled with a platform and language independent messaging protocol called SOAP (Simple Object Access Protocol), XML enables application development and integration by assembling previously built components from multiple Web services.

Enterprise Connector for J2EE Connector Architecture (JCA)

The Enterprise Connector for J2EE Connector Architecture (JCA) enables developers of JCA-compliant applications to deploy adapters as JCA resources. The connector is supported on J2EE-compliant application servers, such as your application server.

The Connector for JCA is distributed as a standard Resource Adapter Archive (RAR) for deployment to the application server. Thus, the connector can be used in systems that are non-compliant, although services such as pooled connections are not available.

Understanding Web Services and Java Connector Architecture Functionality

The following section describes how the BEA WebLogic Adapter for Siebel can incorporate Web services and Java Connector Architecture technology.

Web Services

Web services allow Siebel application calls to be made across the Internet or an intranet, using specialized versions of the XML language that allow a developer to specify the parameters, connections methods, and remote calls and store them for reference in a repository. At runtime, a person, an interface, or another function, can read this repository and automatically invoke the service. Web services currently do not have industry standards for transactional behavior. Web services are useful when your function calls must be made across firewall boundaries. Using Web services, you can use functions provided by external providers, as long as you know the function interface.

Java Connector Architecture

Java Connector Architecture (JCA) provides a reusable component model to build and deploy multi-tier applications that are platform and vendor-independent. JCA acts as a type of envelope or "container" that will allow the adapter to run inside the application server and connect to Siebel and immediately return the results. JCA is useful when your Siebel application system resides within a local intranet or is accesed directly. JCA implements JAVA Connection and Transaction models. JCA requires a resource adapter to be physically deployed on the host application server to access the remote EIS system.

Using combinations of JCA and Web services is possible. For example, a JCA application can be invoked via a Web service or a Web service may be implemented inside a JCA container. The standards and protocols are still evolving.

Understanding JCA Managed and Non-managed Modes

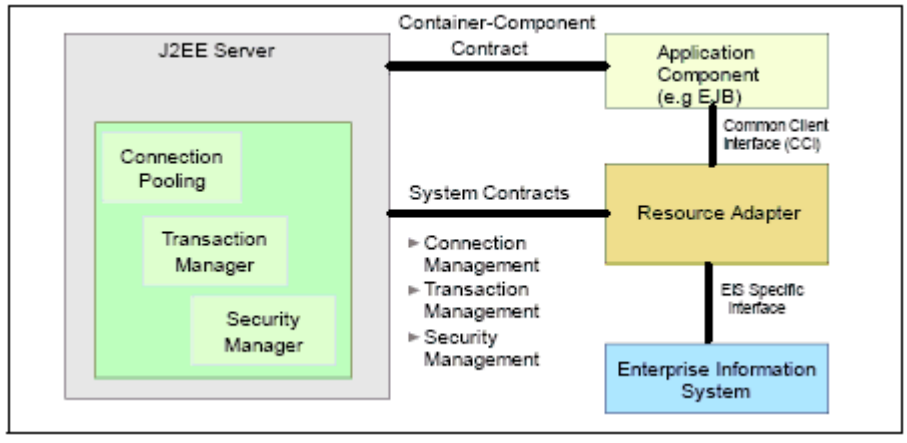
The J2EE Connector Architecture (JCA) provides a specification that standardizes access to Enterprise Information Systems (EIS). When a connection to an EIS is made, this connection can exist in a managed or non-managed mode, depending on your environment and deployment method.

The following section provides an overview of JCA managed and non-managed modes and their key differences. In addition, information relating to the deployment of the Connector for JCA in a managed or non-managed mode is provided.

JCA Managed Mode

By default, the Connector for JCA runs in managed mode when deployed to an application server, for example, BEA WebLogic. In this case, the BEA WebLogic server manages the connections, transactions, and security. As a result, the Connector for JCA provides features such as connection pooling. Connection pooling allows the BEA WebLogic server to pool connections to a back-end EIS, which results in efficient performance and increased scalability.

The following diagram illustrates the Connector for JCA (Resource Adapter) being used in managed mode. In this scenario, the application is running inside the J2EE application server.



In order to make connections to an EIS in managed mode, the Connector for JCA obtains an `IWAFConnectionFactory` from the BEA WebLogic server. In managed mode, the `IWAFConnectionFactory` is accessed from the JNDI and its properties will be configured by the BEA WebLogic server.

The following connection properties can be found in the weblogic-ra.xml file, which is packaged in the iwafjca.rar file.

```
<connection-instance>
        <jndi-name>eis/IWAFConnectionFactory</jndi-name>
         <connection-properties>
            <pool-params>
               <initial-capacity>0</initial-capacity>
               <max-capacity>10</max-capacity>
            </pool-params>
         </connection-properties>
</connection-instance>
```

The following connection properties can be found in the web.xml file, which is used by the JCA IVP test tool to run in managed mode.

```
<context-param>
    <description>JNDI name for the IWAF JCA Resource Adapter.
      If not provided, the application will create a new one based on
      iway.home, iway.config and iway.loglevel.
    </description>
    <param-name>iway.jndi</param-name>
    <param-value>eis/IWAFConnectionFactory</param-value>
</context-param>
```

The iway.jndi parameter is the connection factory name for the Connector for JCA. The connection factory name under BEA WebLogic is eis/IWAFConnectionFactory. The JCA IVP test tool attempts to connect to the adapter via JNDI if JNDI is defined (managed mode). If JNDI is undefined, iway.home, iway.config, and iway.loglevel are used instead (non-managed-mode).

The default location of the web.xml file on Windows is:

```
C:\Program Files\iWay55\bea\iwfjcaivp\WEB-INF\web.xml
```

JCA Non-managed Mode

You can use the Connector for JCA in non-managed mode by embedding it in an application instead of deploying it to an application server. In this case, the application must manage connections, transactions, and security itself. To enable non-managed mode for the the Connector for JCA, you must specify the iWay55\lib connector library in the classpath. No RAR file or ra.xml descriptor is needed. In non-managed mode, the default connection manager is used.

Performance may be reduced when deploying the Connector for JCA in non-managed mode since connection pooling is not available.

Application Design

This section addresses best practice principles that can be used during application design stages when using the BEA WebLogic Adapter for Siebel.

Web Services and JCA Usage Considerations

The following four factors explain the differences between deploying Web services and deploying the JCA option. Understanding the factors can help in selecting a deployment option.

1. Web services is the preferred deployment option because JCA:

- Can be deployed in a separate instance of BEA WebLogic server.
- Provides better distribution of load.
- Provides better isolation from any errors from third-party libraries.
- Provides better capability to isolate issues for debugging purposes.
- Conforms more closely to SOA model for building applications.

2. JCA provides slightly better performance

   JCA does provide slightly better performance than the Web services option; however, the difference decreases as the transaction rate increases.

3. The Web services and JCA options both provide identity propagation at run time.

The Web services option provides the capability to pass identity using the SOAP header. For the JCA option, user name and password can be passed using the connection spec of the CCI.

4. Transactions

   Because no adapters currently being resold by BEA support XA transactions, transactions are not a consideration when choosing between JCA and Web services.

Configuring a Clustered Environment

Events can be configured in a clustered BEA WebLogic environment. You can deploy iBSE or JCA to this environment.

A cluster consists of multiple server instances running simultaneously, yet appears to clients to be a single server instance. The server instances that contain a cluster can be run on one machine, but are usually run on multiple machines.

Service requests are processed through the HTTP router and routed to an available managed server.

Events are server-specific and are not processed through the HTTP router. You must configure each server separately.

A WebLogic cluster is a group of servers that, from the application point-of-view, operate as a single server. A cluster provides:

- **Scalability.** Additional servers can be added to the cluster dynamically to increase capacity.
- **High-availability.** Redundancy of multiple servers insulates applications from failures.

The following operations performed on one managed server will be replicated on all other managed servers:

- Create port and channel. Creates the channel and port under all available servers.
- Delete port and channel. Deletes the port and channel under all available servers.

The following operations must be performed on each server:

- Start channel. Starts the channel for the specific server.
- Stop channel. Stops the channel for the specific server.

For more information on configuring BEA WebLogic for deployment in a clustered environment, see the *Deploying WebLogic Integration Solutions* documentation.

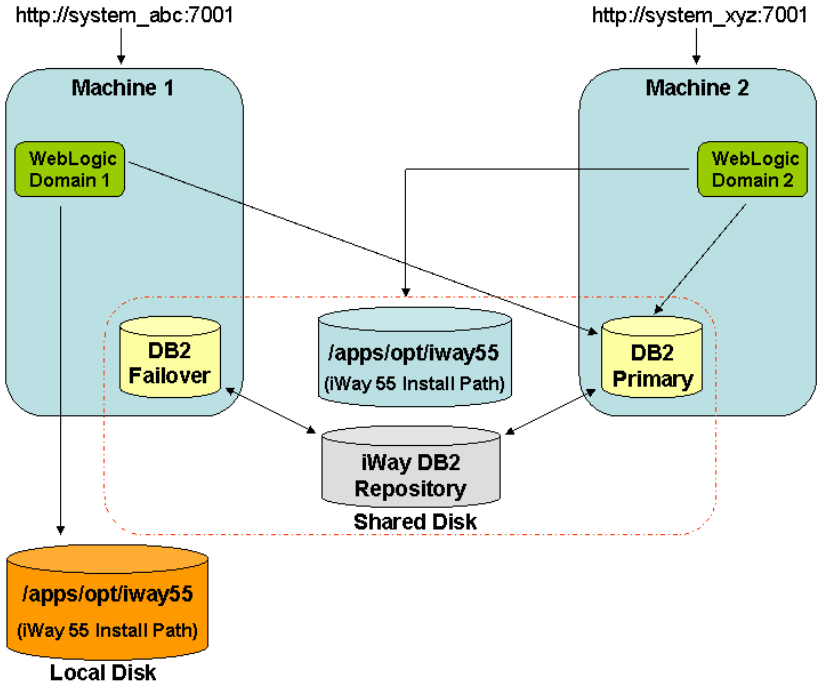
Load Balancing Support with WebLogic Server

The WebLogic Administration Console provides you with the tools to configure WebLogic for load balancing. Load balancing is one of the primary tools employed in making systems scalable. BEA WebLogic Adapter for Siebel takes advantage of the sophisticated load balancing features of WebLogic clusters.

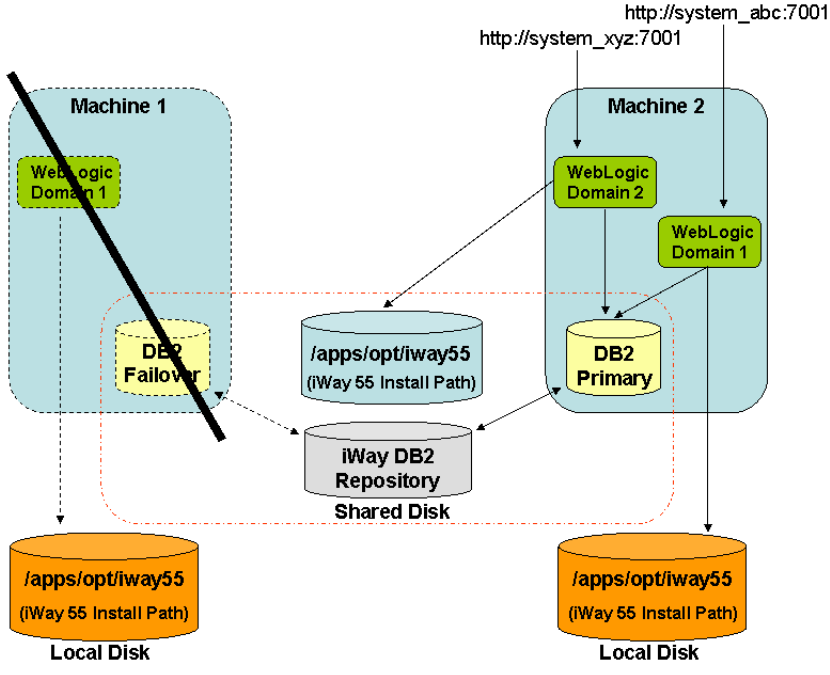
Failover Support with WebLogic Server

Failover is the ability of a system to detect a service failure and automatically retry the operation using another Gateway. This includes the ability to recover in the event of a server failure and the ability to find another instance of a service on an available server. Under the best of conditions the results of a service are idempotent, meaning that multiple invocations of the service, with the same input, produce the same output with no side effects. An example of an idempotent service would be a currency translation service. The service produces the same output each time it is run for a given input value and it has no side effects. Under these conditions it is safe to failover a request for any type of failure.

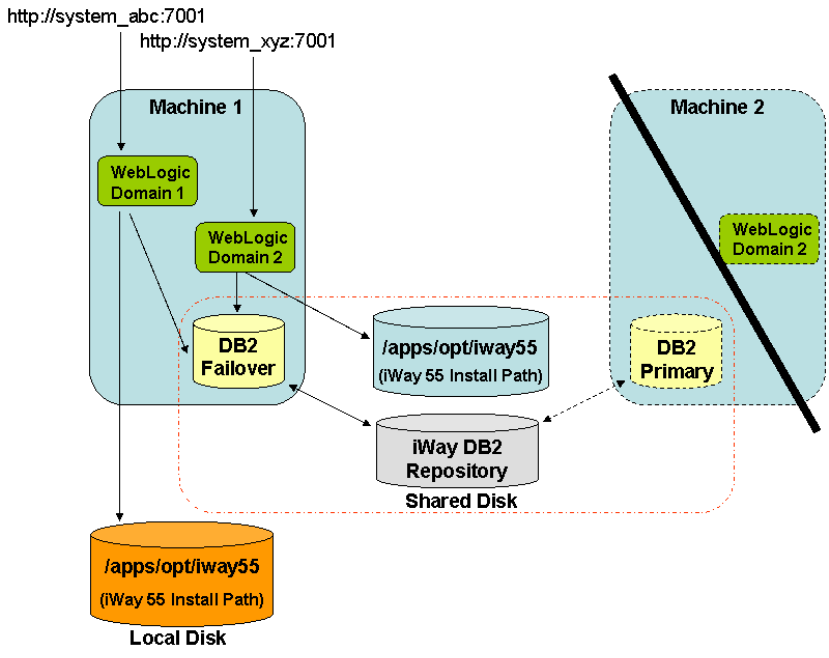
The following diagram illustrates a sample configuration where two WebLogic domains are configured on two separate machines. DB2 Failover and DB2 Primary represent a scenario where DB2 is configured for each WebLogic domain respectively. In the event of a failover condition, the DB2 Primary database instance would be switched over to the DB2 Failover instance.


http://system_abc:7001
http://system_xyz:7001
Machine 1
Machine 2
WebLogic Domain 1
WebLogic Domain 2
DB2 Failover
/apps/opt/iway55
(iWay 55 Install Path)
DB2 Primary
iWay DB2 Repository
Shared Disk
/apps/opt/iway55
(iWay 55 Install Path)
Local Disk

The following diagram illustrates a failover situation when the first machine becomes unavailable. As you can see, Machine 2 takes over in the clustered environment.



The following diagram illustrates a failover situation when the second machine becomes unavailable. In this case, Machine 1 takes over in the clustered environment.



Configuration Tuning

This section explores tuning considerations during design time and run time.

Repository Migration Using iBSE Administrative Services

iBSE Administrative Services allow a client application to export, import, and modify an existing iBSE repository, regardless of the underlying format. When suitably configured, the iBSE Administrative Services represent a best practice methodology for maintaining separate development, QA, and production environments.

For more information on using iBSE administrative services, see Chapter 4, *Management and Monitoring*.

JCA Troubleshooting

JCA provides the ability to set multiple log levels. The log level is set in the META-INF\ra.xml file.

To change defaults, you must:

1. Extract the META-INF\ra.xml file from the iwafjca.rar archive.

   a. Open a command prompt and navigate to the directory containing the connector, for example:

   ```
   iWay55\etc\setup
   ```

   where:

   iWay55

   Is the full path to your iWay 5.5 installation. The default is C:\Program Files\iWay55.

   b. Issue the following command:

   ```
   jar xvf iwafjca.rar META-INF/ra.xml
   ```

   The JAR command is located in the Java SDK bin directory which might not be in your search path. If you receive an error, execute the jar command using its full path. This path varies depending on which version of Java is installed, for example:

   ```
   C:\j2sdk1.4.1_03\bin\jar -xvf iwafjca.rar META-INF/ra.xml
   ```

   Note: Ensure you use the JAR command and not Winzip. Winzip does not properly extract Java related archives.

2. Open the extracted ra.xml file in a text editor.
3. Modify the contents of the <param-value> tags to change defaults.

   - **LogLevel.** Trace setting. This can be set to DEBUG, INFO, or ERROR.

   ```
   <context-param>
   <config-property>
       <config-property-name>LogLevel</config-property-name>
       <config-property-type>java.lang.String</config-property-type>
       <config-property-value></config-property-value>
   </config-property>
   ```

   For example:

   ```
   <config-property-value>DEBUG</config-property-value>
   ```

   Leave the remainder of this file unchanged.

4. Save the file and exit the editor.

5. Use the JAR command to return the ra.xml file to the META-INF directory within the archive. To do this:

   a. Ensure that you are in the following directory that contains the connector:

      ```
      iWay55\etc\setup
      ```

      where:

      iWay55

      Is the full path of your iWay 5.5 installation directory. The default is C:\Program Files\iWay55.

   b. Issue the following command:

      ```
      jar -uvf iwafjca.rar META-INF/ra.xml
      ```

6. Redeploy the connector.

   The trace information is written to the BEA WebLogic server log file.