Siebel Application Integration for Oracle Fusion Middleware Guide

Siebel Innovation Pack 2015

May 2015

ORACLE®

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information,
visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Contents

Chapter 1: What's New in This Release

Chapter 2: Overview of SAI for Oracle Fusion Middleware

Chapter 3: Installing, Configuring, and Upgrading SAI for Oracle Fusion Middleware

Chapter 4: Setting Up Security for SAI for Oracle Fusion Middleware on Oracle WebLogic Server

Chapter 5: Deploying Siebel Resource Adapters by Using SAI for Oracle Fusion Middleware

Chapter 6: Working with Web Services Using SAI for Oracle Fusion Middleware

Chapter 7: Working with Siebel RESTful Services

Chapter 8: Working with XSDs Using SAI for Oracle Fusion Middleware

Chapter 9: Migrating Siebel Web Services to Oracle WebLogic Server

Chapter 10: Troubleshooting SAI for Oracle Fusion Middleware

Glossary

Index

1 What’s New in This Release

What’s New in Siebel Application Integration for Oracle Fusion Middleware Guide, Siebel Innovation Pack 2015

No new features have been added to this guide for this release. This guide has been updated to reflect only product name changes.

NOTE: Siebel Innovation Pack 2015 is a continuation of the Siebel 8.1/8.2 release.

2 Overview of SAI for Oracle Fusion Middleware

This chapter provides an overview of Siebel Application Integration for Oracle Fusion Middleware (SAI for Oracle Fusion Middleware) and how this integration environment relates to Siebel Enterprise Application Integration (EAI). This chapter also describes Oracle WebLogic Server and Oracle JDeveloper and how to use them in relation to SAI for Oracle Fusion Middleware. It includes the following topics:

About SAI for Oracle Fusion Middleware

SAI for Oracle Fusion Middleware is an enterprise application integration infrastructure that allows you to create composite applications with a single view of data in the entire enterprise. A composite application is an application built by combining multiple existing functions into a new application drawn from several different sources. The components can be individual selected functions from within other applications, or entire data management systems whose outputs have been packaged as business functions, modules, or Web services.

Using SAI for Oracle Fusion Middleware, you can resolve inherent integration challenges because you can do the following:

- Present information from multiple applications in a composite application.
- Comply with evolving integration standards. Avoid long design and implementation cycles.
- Experience higher volumes and faster response times.
- Use your existing investment in Siebel Business Applications integration to accelerate the development of composite applications.
- Use Siebel RESTful Services to access data in Siebel Business Applications.
- Design and deploy XML Schema Definitions (XSDs) for use with Siebel RESTful services.

- Use service-oriented architecture (SOA) development and Siebel integration resources.
- Design and deploy Siebel inbound and outbound Web services.

SAI for Oracle Fusion Middleware is built on Oracle’s standards-compliant middleware. Session management and user impersonation provide a more linear progression when scaling for modern integration patterns, and you use new migration tools to move existing Siebel Enterprise Application Integration (EAI) Web services to the new architecture. And, because there is no dependence on Siebel integration resources, there is enhanced performance for SOA development.

Some of the benefits of using SAI for Oracle Fusion Middleware include:

- Faster time to market
- Enhanced performance and scalability
- Additional capability for managing and scaling integration independently
- Enhanced standards compliance with greater alignment to standards evolution
- New technologies that make designing and implementing SOA architectures and solutions easier
- Ability to migrate functionality in phases that suit an organization’s specific requirements

How SAI for Oracle Fusion Middleware Relates to Siebel EAI

The existing Siebel EAI infrastructure is best suited for business-to-consumer (B2C) integration solutions where every transaction is conducted in the framework of a specific user session. In comparison, the SAI for Oracle Fusion Middleware infrastructure is designed to accommodate business-to-business (B2B) as well as B2C integration scenarios.

With SAI for Oracle Fusion Middleware, you can achieve similar results as Siebel EAI but with a better balance of hardware components. The workload profile of the various server components using SAI for Oracle Fusion Middleware is less than that required for Siebel EAI.

How Does SAI for Oracle Fusion Middleware Work?

SAI for Oracle Fusion Middleware provides a mechanism for making Siebel Web services available through Oracle WebLogic Server. It contains the components shown in Figure 1.

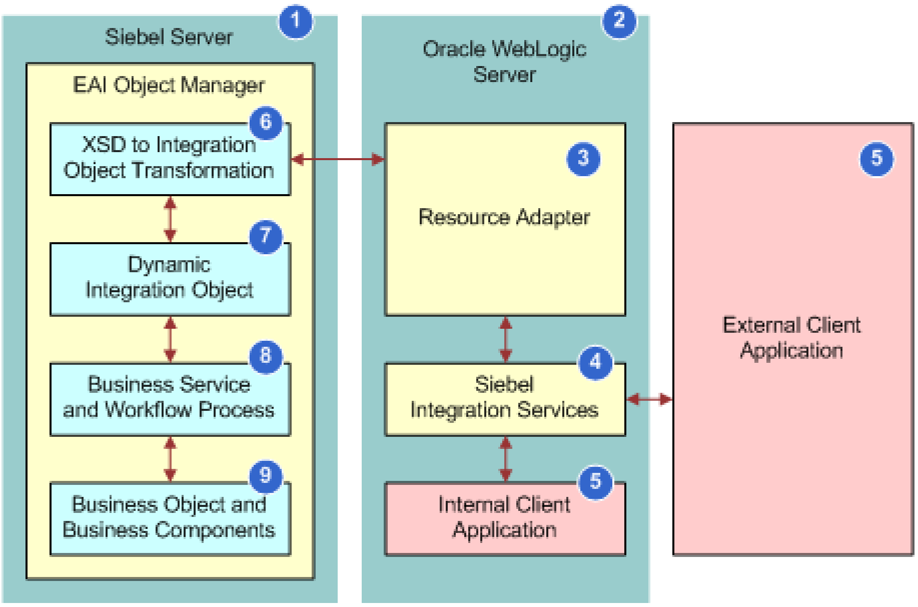
Siebel Server
1
EAI Object Manager
6
XSD to Integration Object Transformation
7
Dynamic Integration Object
8
Business Service and Workflow Process
9
Business Object and Business Components
Oracle WebLogic Server
2
3
Resource Adapter
4
Siebel Integration Services
5
Internal Client Application
5
External Client Application

Figure 1. SAI for Oracle Fusion Middleware Workflow

Figure 1 illustrates the functional components involved in processing an inbound Web service called using SAI for Oracle Fusion Middleware.

1 A client application (5) sends a request that is processed by the Web service (4) deployed on Oracle WebLogic Server (2).

2 This Web service deployment communicates through the Resource Adapter (3) to the Siebel Server (1) where the message is passed to the EAI Object Manager.

3 Within the EAI Object Manager, the inbound message is examined and any XML Schema Definition (XSD) referenced in the message is presented to the XSD to the integration object transformation section (6).

4 Next, a cache is consulted to see if an existing integration object is available, and if not, then a dynamic integration object (7) is created and loaded into the cache.

5 This dynamic integration object is then used to pass the data through the target business service and workflow process (8) and interacts with the Siebel business object layer (9).

SAI for Oracle Fusion Middleware Workflow

Figure 1 additionally illustrates the following workflow:

1. The Siebel Server executes one or more instances of the Siebel EAI Object Manager.
2. Oracle WebLogic Server executes one or more instances of the Resource Adapter and instances of the deployed Web services and other client applications.
3. The Resource Adapter provides the communications channel between the Siebel Server and Oracle WebLogic Server.
4. Siebel integration services are implemented as deployments within Oracle WebLogic Server architecture; these deployments include both specific SOAP-based services or generic Siebel RESTful Services.
5. Client applications can be either internal or external to Oracle WebLogic Server instance.
6. At runtime, the XSD referenced in any inbound message is examined, checked against a cache, and if required, converted to a dynamic IO.
7. At this stage, the underlying cache for dynamic integration objects is queried and, depending on the result, a new cache entry is created. The new cache is based on the message format.
8. Business services and workflow processes are the functional service components that provide access to Siebel CRM data through the Siebel Object Model.
9. Business object and business components are representative of the Siebel Object Model through which Siebel data is transferred to and from Oracle WebLogic Server.

About Oracle WebLogic Server

Oracle WebLogic Server is a scalable, enterprise-ready Java Platform, Enterprise Edition (Java EE) application server. The Oracle WebLogic Server infrastructure supports the deployment of many types of distributed applications and is an ideal foundation for building applications based on Service Oriented Architectures (SOA). SOA is a design methodology aimed at maximizing the reuse of application services.

Oracle WebLogic Server provides a standard set of application programming interfaces (APIs) based on the Java EE 5.0 specification for creating distributed Java applications that can access a wide variety of services, such as databases, messaging services, and connections to external enterprise information systems (EIS). Clients can access these applications using Web browser clients or Java clients. Diagnostic tools allow system administrators to monitor and tune the performance of deployed applications and the Oracle WebLogic Server environment. You can also configure Oracle WebLogic Server to monitor and tune application throughput automatically without human intervention. Security features protect access to services, keep enterprise data secure, and prevent malicious attacks.

About Oracle WebLogic Server Domains

Before you can develop and run a WebLogic application, you must first create a *domain*. A domain is an interrelated set of Oracle WebLogic Server resources that is managed as a unit. For information about creating domains, see "Oracle WebLogic Server Configuration Wizard" on page 17. A domain includes one or more Oracle WebLogic Server instances, which can form clusters, instances, or a combination of both. A domain can include multiple clusters. For more information about clusters, see "About Oracle WebLogic Server Clusters" on page 15.

A domain also contains the application components deployed in the domain and the resources and services required by those application components and the server instances in the domain. Examples of the resources and services used by applications and server instances include computer definitions, optional network channels, connectors, and startup classes. After you create a domain, you can start an Oracle WebLogic Server instance to run in the domain for developing, testing, and deploying applications.

As shown in Figure 2, an Oracle WebLogic domain consists of the following:

- One administration server
- One or more managed servers
- One or more clusters

An Oracle WebLogic domain is managed using the Oracle WebLogic Server System Administration Tools. For more information about these tools, see "About System Administration Tools for Oracle WebLogic Server" on page 16.

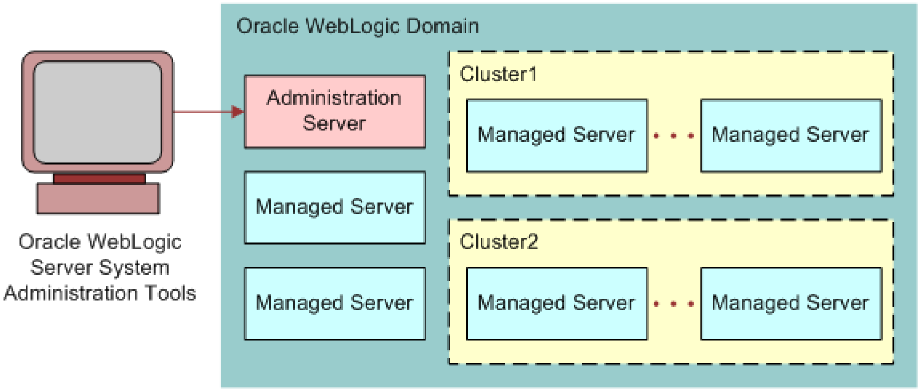

Figure 2. Structure of an Oracle WebLogic Domain

Figure 2 illustrates the structure of an Oracle WebLogic domain:

- One administration server that is managed using the Oracle WebLogic Server System Administration Tools
- One or more managed servers
- One or more clusters with one or more managed servers for each cluster

Components of an Oracle WebLogic Server Domain

An Oracle WebLogic Server administration domain is a logically related group of Oracle WebLogic Server resources. Domains include a special Oracle WebLogic Server instance called the administration server, which is the central point from which you configure and manage all resources in the domain. Usually, you configure a domain to include additional Oracle WebLogic Server instances called managed servers. You can optionally configure the managed servers into clusters. The components of a domain are described in Table 2.

Table 2. Components of an Oracle WebLogic Server Domain

Component	Description
Administration server	A domain includes one Oracle WebLogic Server instance that is configured as an administration server. You make all changes to the configuration and deployment of applications in the administration server. The administration server provides a central point for managing the domain and providing access to the Oracle WebLogic Server administration tools. These tools include the following: ■ **Administration Console.** A graphical user interface (GUI) to the administration server. For more information about the Administration Console, see "Oracle WebLogic Server Administration Console" on page 17. ■ **Node Manager.** A Java program that enables you to start and stop server instances remotely and to monitor and automatically restart them after an unexpected failure. For more information about Node Manager, see "Oracle WebLogic Server Node Manager" on page 19.

Table 2. Components of an Oracle WebLogic Server Domain

Component	Description
Managed servers	All Oracle WebLogic Server instances other than the administration server in a domain are called *managed servers.* Managed servers host application components and resources, which are also deployed and managed as part of the domain. In a domain with only a single Oracle WebLogic Server instance, that single server works as both the administration server and the managed server. **NOTE:** All managed servers in a domain must run the same version of Oracle WebLogic Server. The administration server can run either the same version as the managed servers in the domain or a later service pack.
Clusters	A domain might also include Oracle WebLogic Server clusters, which are groups of managed server instances that work together to provide scalability and high availability for applications. Clusters can improve performance and provide *failover* when a server instance becomes unavailable. In a clustered environment, failover is when one node fails but transactions continue to process on other nodes with no interruption of the end-user experience. The servers within a cluster can either run on the same computer or reside on different computers. For more information about clusters, see "About Oracle WebLogic Server Clusters" on page 15. For more information about failover, see *Siebel Self-Service Application Developer's Guide* on *Siebel Bookshelf*. **NOTE:** The *Siebel Bookshelf* is available on Oracle Technology Network (http://www.oracle.com/technetwork/indexes/documentation/index.html) and Oracle Software Delivery Cloud. It might also be installed locally on your intranet or on a network location.

About the Configuration of Oracle WebLogic Domains

In addition to the components, a domain defines the basic network configuration for the server instances that it contains. Specifically, a domain defines application deployments, supported application services (such as database and messaging services), security options, and physical host computers. Domain configuration information is stored in the configuration directories under the domain directory.

About Oracle WebLogic Server Clusters

In addition to the Java EE implementation, Oracle WebLogic Server allows you to deploy your own applications. You can configure clusters of Oracle WebLogic Server instances to distribute the load and provide extra capacity in case of hardware or other failures.

An Oracle WebLogic Server *cluster* consists of multiple Oracle WebLogic Server instances running simultaneously and working together to provide increased scalability and reliability. It is part of a particular Oracle WebLogic Server domain. It appears to clients as a single Oracle WebLogic Server instance. The server instances that constitute a cluster can run on the same computer or be on different computers. You can increase the capacity of a cluster by adding additional server instances to the cluster on an existing computer, or you can add computers to the cluster to host the additional server instances. Each server instance in a cluster must run the same version of Oracle WebLogic Server. For more information about domains, see "About Oracle WebLogic Server Domains" on page 13.

About System Administration Tools for Oracle WebLogic Server

Oracle WebLogic Server provides a variety of tools and utilities for system administrators. Some of these tools include the following:

- **Administration Console.** For more information, see "Oracle WebLogic Server Administration Console."
- **Configuration Wizard.** For more information, see "Oracle WebLogic Server Configuration Wizard" on page 17.
- **Domain Template Builder.** For more information, see "Oracle WebLogic Server Domain Template Builder" on page 18.
- **Domain Upgrade Wizard.** For more information, see "Oracle WebLogic Server Domain Upgrade Wizard" on page 18.
- **Node Manager.** For more information, see "Oracle WebLogic Server Node Manager" on page 19.
- **WebLogic Scripting Tool.** For more information, see "About the WebLogic Scripting Tool" on page 20.

Some of the server administration tasks you might perform in a SAI for Oracle Fusion Middleware environment include the following:

- Creating Oracle WebLogic Server domains
- Migrating domains from development environments to production environments
- Deploying applications
- Monitoring and configuring the performance of the Oracle WebLogic Server domain
- Diagnosing and troubleshooting problems

For detailed information about the tasks above, see the applicable Oracle WebLogic Server administration guide in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

Oracle WebLogic Server Administration Console

This topic provides a high-level description of the Oracle WebLogic Server Administration Console. For more information about this console and how to use it, see *Administration Console Online Help* in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

The Administration Console is a Web browser-based, graphical user interface that you use to manage an Oracle WebLogic Server domain. For more information about WebLogic Domains, see “About Oracle WebLogic Server Domains” on page 13.

You use the Administration Console to:

- Configure, start, and stop Oracle WebLogic Server instances
- Configure Oracle WebLogic Server clusters
- Configure Oracle WebLogic Server services, such as database connectivity (JDBC) and messaging (JMS)
- Configure security parameters, including managing users, groups, and role
- Configure and deploy your applications
- Monitor server and application performance
- View server and domain log files
- View application deployment descriptors
- Edit selected run-time application deployment descriptor elements

Related Topic

“Logging In to the Oracle WebLogic Server Administration Console” on page 39

Oracle WebLogic Server Configuration Wizard

This topic provides a high-level description of the Oracle WebLogic Server Configuration Wizard and how to use it. For more information about this wizard, see *Oracle® Fusion Middleware Creating Domains Using the Configuration Wizard 11g Release 1 (10.3.1)* in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library. You use the Configuration Wizard to create and extend domains. For more information about domains, see “About Oracle WebLogic Server Domains.”

You create and extend domains using the Configuration Wizard by selecting the product components you want included in the domain (or choose a template that best meets your requirements), and provide basic configuration information. The Configuration Wizard then creates or extends the domain by using the settings from the templates.

Before you can develop and run an Oracle WebLogic application, you must first create a domain. After you create a domain, you can start an Oracle WebLogic Server instance to run in the domain for developing, testing, and deploying applications.

Oracle WebLogic Server Domain Template Builder

This topic provides a high-level description of the Oracle WebLogic Server Domain Template Builder and how to use it. For more information about this wizard, see *Oracle® Fusion Middleware Creating Domain Templates Using the Domain Template Builder 11g Release 1 (10.3.1)* in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

The Domain Template Builder is a standalone Java application that guides you through the process of creating custom domain and extension templates. You create Oracle WebLogic domains by using domain and extension templates that contain the key attributes and files required for building or extending a domain. You can also use the Domain Template Builder to customize an existing template. For example, you might want to remove applications or add SQL scripts for additional databases.

The Domain Template Builder is designed to be used offline, in graphical mode only. This application is not supported in console mode. Therefore, the console attached to the computer on which you are using the Domain Template Builder must support Java-based GUIs. All consoles running on Windows operating systems support Java-based GUIs. Only a subset of UNIX-based consoles supports Java-based GUIs.

The Oracle WebLogic Server installation includes a set of predefined domain and extension templates. This set of templates includes the base Oracle WebLogic Server domain template and various extension templates that allow you to add product features and samples to the base domain.

Oracle WebLogic Server Domain Upgrade Wizard

This topic provides a high-level description of the Oracle WebLogic Server Domain Upgrade Wizard and how to use it. For more information about this wizard, see *Oracle® Fusion Middleware Upgrade Guide for Oracle WebLogic Server 11g Release 1 (10.3.1)* in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

You use the Domain Upgrade Wizard to streamline the upgrade of domains with a newer version of Oracle WebLogic Server. For example, you might want to use this wizard to upgrade a domain to version 11*g* that was created in version 10.3.x.

You must upgrade the domains on every computer in the domain. The wizard supports the following upgrade modes:

- **Graphical.** Use this mode to upgrade a domain interactively; that is, using a graphical user interface.
- **Silent.** Use this mode to upgrade an Oracle WebLogic Server domain by specifying the upgrade requirements in a file.

Tasks Performed by the Upgrade Wizard

During an Oracle WebLogic domain upgrade, you specify the domain that you want to upgrade and respond to a set of prompts. The wizard performs the following tasks:

1. (Optional) Backs up the original domain directory.

2. Recreates scripts, such as the startup and shutdown scripts, and renames any original scripts as orig-scriptname.bak, where *orig-scriptname* specifies the original script name and extension.
3. Restructures the original domain, creating a new directory structure and then moving the domain components to new locations.
4. Upgrades the persistent configuration information stored in the configuration file (config.xml) to the configuration directory.
5. Upgrades the persistent data, such as the JMS file stores, JMS JDBC stores, and transaction stores.
6. Saves the configuration.
7. Reports any errors that occurred during the domain upgrade.

   For information about troubleshooting errors, see "Troubleshooting SAI for Oracle Fusion Middleware" on page 117.

Oracle WebLogic Server Node Manager

This topic provides a high-level description of Oracle WebLogic Server Node Manager. For more information about Node Manager, see *Oracle® Fusion Middleware Node Manager Administrator's Guide for Oracle WebLogic Server 11g Release 1 (10.3.1)* in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

Node Manager is an optional Oracle WebLogic Server utility that allows you to manage server instances from a remote location. Using Node Manager, you can do the following:

- Start, shut down, and restart an administration server
- Start, shut down, suspend, and restart managed servers
- Restart the administration server and managed servers
- Monitor servers and view the log files

NOTE: It is recommended that you use Node Manager if your Oracle WebLogic Server environment hosts applications with high-availability requirements.

A node manager process is not associated with a specific Oracle WebLogic domain but with a computer. You can use the same Node Manager process to control server instances in any Oracle WebLogic Server domain, as long as the server instances reside on the same computer as the node manager process. Node Manager must run on each computer that hosts Oracle WebLogic Server instances that you want to control with Node Manager.

Oracle WebLogic Server supports the following two versions of Node Manager:

- **Java-based.** Java-based Node Manager runs within a Java Virtual Machine (JVM) process. It is recommended that you run it as a Windows service on a Windows operating system and as an operating system service on UNIX platforms, allowing it to restart automatically when the computer is rebooted. This version of Node Manager determines its configuration from the nodemanager.properties file.

- **Script-based.** Script-based Node Manager is for UNIX and Linux operating systems. This script is based on the UNIX shell scripts but uses Secure Shell (SSH) for increased security. SSH is a network protocol that allows data to be exchanged using a secure channel between two networked devices and implements ID-based security.

NOTE: Java-based Node Manager provides more security than the script-based version. However, the advantage of the script-based Node Manager is that it can remotely manage servers over a network that has been configured to use SSH. No additional server installation is required. You copy the scripts to the remote computer.

About the WebLogic Scripting Tool

This topic provides a high-level description of the WebLogic Scripting Tool. For more information about this tool, see *Oracle® Fusion Middleware Oracle WebLogic Scripting Tool 11g Release 1 (10.3.1)* in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

WebLogic Scripting Tool is a command-line scripting environment that you can use to create, manage, and monitor Oracle WebLogic Server instances and domains as well as monitor and manage server run-time events. WebLogic Scripting Tool is based on the Java scripting interpreter, Jython. You can use WebLogic Scripting Tool as the command-line equivalent to the Oracle WebLogic Server Administration Console (online) or as the command-line equivalent to the Configuration Wizard (offline).

You can use any of the following modes to call WebLogic Scripting Tool commands:

- **Interactive.** In interactive mode, you enter a command and view the response at a command-line prompt. This mode is useful for learning the tool, prototyping command syntax, and verifying the configuration options before building a script.
- **Script.** In script mode, you can call a sequence of WebLogic Scripting Tool commands without your input, much like a shell script. Scripts contain the WebLogic Scripting Tool commands in a text file with a .PY file extension, for example, filename.py. You use script files with the Jython commands for running scripts.
- **Embedded.** In embedded mode, you instantiate the WebLogic Scripting Tool interpreter in your Java code and use it to run WebLogic Scripting Tool commands and scripts. You can run in embedded mode the WebLogic Scripting Tool commands and variables that you use in interactive and script mode.

About Oracle JDeveloper

Oracle JDeveloper is an integrated development environment (IDE) for developing and deploying Java-based service-oriented architecture (SOA) applications and user interfaces. Oracle JDeveloper integrates development features for Java, SOA, Web 2.0, Database, XML, and Web services into a single development tool.

You use Oracle JDeveloper to deploy and use SAI for Oracle Fusion Middleware. For more information about Oracle JDeveloper, see the Oracle JDeveloper documentation in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

About the SAI for Oracle Fusion Middleware Wizards

The interface for using SAI for Oracle Fusion Middleware is by way of three different wizards in Oracle JDeveloper. Table 3 describes what you can accomplish with each wizard. For more information about Oracle JDeveloper, see "About Oracle JDeveloper" on page 20.

Table 3. SAI for Oracle Fusion Middleware Wizards

Wizard	Description
Deployment	Use this wizard to deploy Siebel Web services, Siebel RESTful Services, Enterprise Java Beans (EJBs), and XSDs to Oracle WebLogic Server.
Design	Use this wizard to generate business service and workflow process code which you can later use to deploy as inbound Web services. You can also use this wizard to design an XSD by selecting a base integration object that you can later use in a Siebel RESTful Service request.
Migration	Use this wizard to migrate existing Siebel inbound Web services.

For more information about these wizards and how to use them, see *Siebel Application Integration for Oracle Fusion Middleware: Oracle JDeveloper Online Help*.

3 Installing, Configuring, and Upgrading SAI for Oracle Fusion Middleware

This chapter describes how to install and configure SAI for Oracle Fusion Middleware for Siebel Business Applications. It includes the following topics:

About the SAI for Oracle Fusion Middleware Extension Bundle

SAI for Oracle Fusion Middleware is provided in Siebel Business Applications by way of an Oracle JDeveloper WebCenter extension bundle. You use Oracle JDeveloper to install and use SAI for Oracle Fusion Middleware. Oracle JDeveloper is an integrated development environment for developing and deploying Java applications and Web services. For more information about Oracle JDeveloper, see the Oracle JDeveloper documentation in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library. For information about installing SAI for Oracle Fusion Middleware, see “Installing SAI for Oracle Fusion Middleware” on page 28.

Roadmap for Installing and Configuring SAI for Oracle Fusion Middleware

Figure 3 provides a roadmap for installing and configuring SAI for Oracle Fusion Middleware. It is recommended that you print this roadmap to use as a checklist as you prepare to use SAI for Oracle Fusion Middleware.

Explanation of Callouts

In Figure 3, the following conventions are used:

- A green box indicates a task.
- A brown box indicates a subtask.
- A broken line indicates a first-time installation only task. The task is not applicable to upgrades.

- A yellow box indicates that information for performing this step is documented external to this guide.

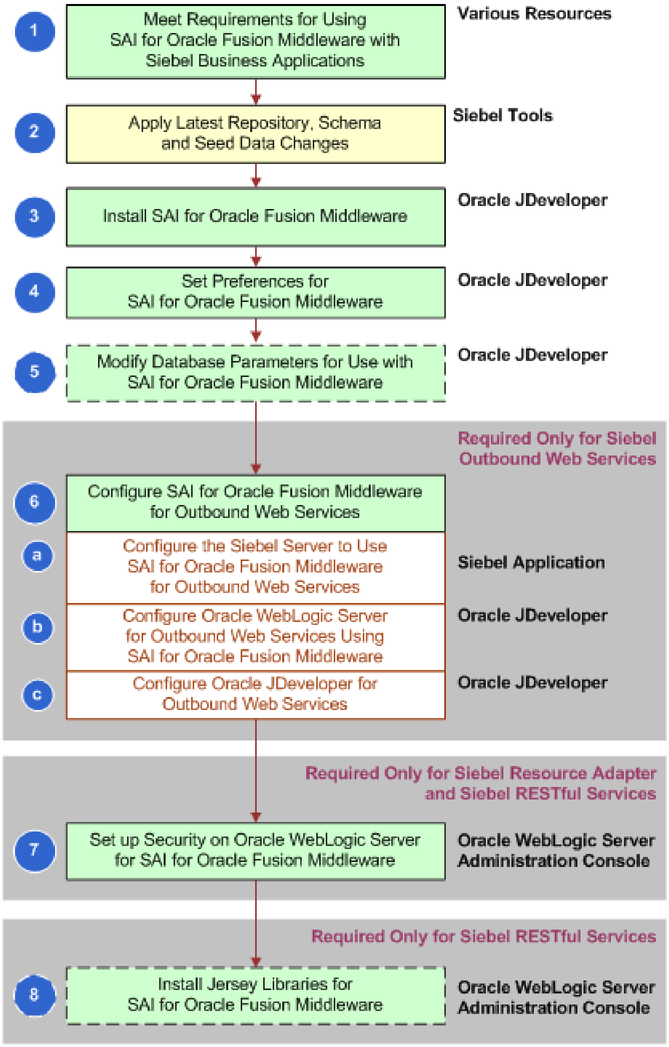
1
Meet Requirements for Using SAI for Oracle Fusion Middleware with Siebel Business Applications
Various Resources
2
Apply Latest Repository, Schema and Seed Data Changes
Siebel Tools
3
Install SAI for Oracle Fusion Middleware
Oracle JDeveloper
4
Set Preferences for SAI for Oracle Fusion Middleware
Oracle JDeveloper
5
Modify Database Parameters for Use with SAI for Oracle Fusion Middleware
Oracle JDeveloper
Required Only for Siebel Outbound Web Services
6
Configure SAI for Oracle Fusion Middleware for Outbound Web Services
a
Configure the Siebel Server to Use SAI for Oracle Fusion Middleware for Outbound Web Services
Siebel Application
b
Configure Oracle WebLogic Server for Outbound Web Services Using SAI for Oracle Fusion Middleware
Oracle JDeveloper
c
Configure Oracle JDeveloper for Outbound Web Services
Oracle JDeveloper
Required Only for Siebel Resource Adapter and Siebel RESTful Services
7
Set up Security on Oracle WebLogic Server for SAI for Oracle Fusion Middleware
Oracle WebLogic Server Administration Console
Required Only for Siebel RESTful Services
8
Install Jersey Libraries for SAI for Oracle Fusion Middleware
Oracle WebLogic Server Administration Console

Figure 3. Roadmap for Installing and Configuring SAI for Oracle Fusion Middleware

Figure 3 illustrates the high-level tasks you perform to install and configure SAI for Oracle Fusion Middleware.

1 "Requirements for Using SAI for Oracle Fusion Middleware with Siebel Business Applications" on page 26

2 "Applying the Latest Repository, Schema, and Seed Data Changes" on page 28

3 "Installing SAI for Oracle Fusion Middleware" on page 28

4 "Setting Preferences for SAI for Oracle Fusion Middleware" on page 29

5. (First-time installation only) "Modifying Database Parameters for Use with SAI for Oracle Fusion Middleware" on page 33

   NOTE: For upgrades, perform this step only if you change the database you want to use.

6. (Outbound Web Services Only) "Process of Configuring SAI for Oracle Fusion Middleware for Outbound Web Services" on page 33

   a. "Configuring the Siebel Server to Use SAI for Oracle Fusion Middleware for Outbound Web Services" on page 34

   b. "Configuring Oracle WebLogic Server for Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 37

   c. "Configuring Oracle JDeveloper for Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 37

7. (Siebel Resource Adapter and Siebel RESTful Services Only) "Setting Up Security on Oracle WebLogic Server for SAI for Oracle Fusion Middleware" on page 42

8. (First-time installation Only for Siebel RESTful Services) "Manually Installing Jersey Libraries for SAI for Oracle Fusion Middleware" on page 38

Requirements for Using SAI for Oracle Fusion Middleware with Siebel Business Applications

This task is a step in "Roadmap for Installing and Configuring SAI for Oracle Fusion Middleware" on page 24.

Before deploying SAI for Oracle Fusion Middleware, you must have met the following requirements.

1. Make sure the following products are installed and properly configured:

   - Siebel CRM Fix Pack 8.1.1.7 or later

     Some of the requirements include:

     - Siebel Gateway Server
     - Siebel Server (or servers)
     - Siebel Tools

     For information about installing fix packs, see the topic about Fix Pack 8.1.1.7 in the applicable *Siebel Maintenance Release Guide* on My Oracle Support. For information about installing Siebel Server and Siebel Tools, see *Siebel Installation Guide* for the operating system you are using.

- The latest Oracle WebLogic Server 11*g* Release 1. For information about which version is applicable to your deployment, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

  NOTE: For Siebel CRM product releases 8.1.1.9 and later and for 8.2.2.2 and later, the system requirements and supported platform certifications are available from the Certification tab on My Oracle Support. For information about the Certification application, see article 1492194.1 (Article ID) on My Oracle Support.

  Download it from Oracle Technology Network at

  http://www.oracle.com/

  NOTE: When installing this server, the basic Oracle WebLogic Server Domain is sufficient. Choose Sun JDK when you are prompted, which is a requirement for Siebel RESTful Services.

  For more information on installing the Oracle WebLogic Server, see *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server* on Oracle Technology Network.

- Oracle JDeveloper 11*g*

  Download it from Oracle Technology Network at

  http://www.oracle.com/technetwork/developer-tools/jdev/documentation/index.html

  For information about which release of Oracle JDeveloper to download, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network. For information about installing Oracle JDeveloper, see *Installation Guide for Oracle JDeveloper* in the Oracle Fusion Middleware Documentation Library 11*g* Release 1.

  NOTE: For Siebel CRM product releases 8.1.1.9 and later and for 8.2.2.2 and later, the system requirements and supported platform certifications are available from the Certification tab on My Oracle Support. For information about the Certification application, see article 1492194.1 (Article ID) on My Oracle Support.

2. Set the following system environment variables to the appropriate directories:

   - ANT_HOME to the Oracle JDeveloper installation directory

     For example, you might set it to the following:

     ```
     D:\Oracle\Middleware\jdeveloper\ant
     ```

   - JAVA_HOME to the JDK (Java Developer’s Kit) installation directory

     For example, you might set it to the following:

     ```
     D:\Oracle\Middleware\jdk160_11
     ```

   For information about the system environment variables, see *Siebel System Administration Guide*.

Applying the Latest Repository, Schema, and Seed Data Changes

Make sure you have applied the latest repository, schema, and seed data changes. For information, see instructions for ACR 750 in 880452.1 (Article ID) *Siebel Maintenance Release Guide* on My Oracle Support.

This task is a step in "Roadmap for Installing and Configuring SAI for Oracle Fusion Middleware" on page 24.

Installing SAI for Oracle Fusion Middleware

This topic provides instructions for installing the Oracle JDeveloper extension bundle for SAI for Oracle Fusion Middleware and other installation tasks. For more information about this extension bundle, see "About the SAI for Oracle Fusion Middleware Extension Bundle" on page 23.

After you install SAI for Oracle Fusion Middleware, you can use this extension bundle in Oracle JDeveloper to deploy resource adapters as well as to design and deploy Web services, Siebel RESTful Services, and XSDs. You can also use SAI for Oracle Fusion Middleware to migrate existing Siebel Web services by deploying them to Oracle WebLogic Server.

NOTE: If you plan to use existing Siebel Web services, then make sure you have migrated the services to Oracle WebLogic Server. For more information about migrating existing Siebel Web services to Oracle WebLogic Server, see Chapter 9, "Migrating Siebel Web Services to Oracle WebLogic Server."

For more information about:

- Deploying resource adapters by using SAI for Oracle Fusion Middleware, see Chapter 5, "Deploying Siebel Resource Adapters by Using SAI for Oracle Fusion Middleware."
- Deploying Web services by using SAI for Oracle Fusion Middleware, see "Working with Web Services Using SAI for Oracle Fusion Middleware" on page 59.
- Deploying Siebel RESTful Services, see Chapter 7, "Working with Siebel RESTful Services."

This task is a step in "Roadmap for Installing and Configuring SAI for Oracle Fusion Middleware" on page 24.

NOTE: Use the following procedure for upgrading as well as for first-time installations.

To install SAI for Oracle Fusion Middleware

1. Make sure you have met the requirements for deploying SAI for Oracle Fusion Middleware.

   For more information, see "Requirements for Using SAI for Oracle Fusion Middleware with Siebel Business Applications" on page 26.

2 Make sure the oracle-siebel-eai-fusion_bundle.zip file resides in the *SIEBSRVR_ROOT*\classes directory where your Siebel application is installed.

NOTE: This file was made available when you installed Siebel CRM Fix Pack 8.1.1.9.

3 Start Oracle JDeveloper from the Start menu.

4 Select the Default Role to enable all technologies.

5 Check for updates by doing the following:

a From the application-level menu, select Help, and then the Check for Updates menu item.

b In the Welcome window, click Next.

c In the Source window, select the Install From Local File option, and then browse to the directory where you downloaded the oracle-siebel-eai-fusion_bundle.zip file to select it.

d Click Next.

TIP: You use the Check for Updates menu item to update an existing SAI for Oracle Fusion Middleware installation as well as for a first-time deployment.

6 For a first-time installation, accept the License Agreements, and then click Next to review the Check for Updates summary.

NOTE: You can find the version number for your deployment from the Oracle JDeveloper application-level menu. Choose Help, the About menu item, and then click the Version tab.

7 Click Finish to close the wizard, and then click Yes when prompted to restart Oracle JDeveloper.

8 Verify SAI for Oracle Fusion Middleware successfully installed by doing the following:

- Select File, and then New to bring up the New Gallery.
- In the Business Tier category, verify SAI for Oracle Fusion Middlewarer Oracle Fusion Middleware appears. This is your access point for working with SAI for Oracle Fusion Middleware.

Related Topics

"Setting Preferences for SAI for Oracle Fusion Middleware" on page 29

Setting Preferences for SAI for Oracle Fusion Middleware

This topic describes how to set preferences for SAI for Oracle Fusion Middleware. You specify details about the database connection and other information relating to the behavior of SAI for Oracle Fusion Middleware.

NOTE: You must have installed the SAI JDeveloper extension bundle for the Siebel Application Integration for Oracle Fusion Middleware option to appear in the Tools Preferences menu. For information about installing the extension bundle, see "Installing SAI for Oracle Fusion Middleware" on page 28.

This task is a step in "Roadmap for Installing and Configuring SAI for Oracle Fusion Middleware" on page 24.

To set preferences for SAI for Oracle Fusion Middleware

1. Start Oracle JDeveloper.
2. From the application-level menu, select Tools, and then the Preferences menu item.
3. From the explorer, scroll down to select Siebel Application Integration for Oracle Fusion Middleware, and then fill in the required fields for the General tab. Do not click OK.

   Some fields are described in the following table.

Field	Description
Service Name	A unique name that identifies the database. **NOTE:** For Oracle databases, this is the actual SERVICE_NAME specified in tnsnames.ora.
Connection Type	The type of database connection that you want to use. You can choose from the following: Oracle, MSSQL, and DB2. By default, this value is set to Oracle. For MSSQL and DB2, you must perform more configuration tasks before you can work with SAI for Oracle Fusion Middleware. For information about the requirements for these databases, see "Modifying Database Parameters for Use with SAI for Oracle Fusion Middleware" on page 33.
Host Name	The Siebel Database Server name.
Port	The Siebel database port number.
Table Owner	The Siebel database table owner name.
Username	The user name used to connect to the Siebel database.
Password	The password used to connect to the Siebel database.
Test Connection	Select this option to test the database connection after you have entered your database settings. A confirmation dialog appears when the connection is successful. Continue setting the remaining parameters only if the connection is successful. If the connection is not successful, then check the database parameters again, or check the database connection.
XSD File Path	The path where Siebel business service code, Siebel workflow process code, and XSD files will be generated. By default, this path is set to one of the following: ■ `C:\JDeveloper\mywork\SFE` ■ *`default_JDeveloper_work_directory`*`\SFE`

Field	Description
Log File Directory	The path to the log files related to the design code for inbound Web services. By default, this path is set to one of the following: ■ `C:\JDeveloper\mywork\SFE` ■ *default_JDeveloper_work_directory*\SFE The information collected can range from error messages to detailed diagnostic log files. **NOTE:** The SiebelToolkitExtension.log file stores all log-related information for the Design Wizard in SAI for Oracle Fusion Middleware.
Log Level	The log level determines the amount of information that is written to the log file. The options are the same as the other log levels in SAI for Oracle Fusion Middleware: FATAL, ERROR, WARN, INFO, DETAIL, and DEBUG. By default, the log level is set to ERROR. For more information about log levels, see "Setting Log Levels for SAI for Oracle Fusion Middleware" on page 118.

4 Click the Deployment tab, and then fill in the required fields.

Some fields are described in the following table.

Field	Description
Root Path of Deployment Archive	The path where the archive files are created during the deployment of SAI for Oracle Fusion Middleware. By default, this path is set to the Oracle JDeveloper IDE work directory (ide.work.dir) appended with mywork\SFE. For example: `C:\JDeveloper\mywork\SFE` or *default_JDeveloper_work_directory*\SFE **TIP:** To find the value for the IDE work directory, from the Oracle JDeveloper application-level menu, select Help, the About menu item, and then click the Properties tab.
Log File Directory	The path to the deployment log files of Siebel Application Integration for Oracle Fusion Middleware. By default, this path is set to: `C:\JDeveloper\mywork\SFE` or *default_JDeveloper_work_directory*\SFE **NOTE:** The SiebelDeployExtension.log file is the log file that stores all log-related information for SAI for Oracle Fusion Middleware deployment processes.
Log Level	The log level for deployment log files. The options are the same as the other log levels in SAI for Oracle Fusion Middleware: FATAL, ERROR, WARN, INFO, DETAIL, and DEBUG. By default, the log level is set to ERROR. For more information about log levels, see "Setting Log Levels for SAI for Oracle Fusion Middleware" on page 118.

Field	Description

Log File Directory of Outbound Web Service	The path to the Siebel outbound log directory within Oracle JDeveloper. This directory includes the log files related to any outbound Web service Java Remote Method Invocation calls. By default, this path is set to:
	`C:\JDeveloper\mywork\SFE` or *`default_JDeveloper_work_directory`*`\SFE`
	The syntax for the outbound log directory is:
	outbound_log_path\SFE\SFEOutboundWebService_*outbound_Web_service_name*.log
	For example, if the outbound log path specified in Preferences is `C:\JDeveloper\mywork\SFE`, and if the outbound Web service name is *GlobalWeather*, then the outbound log file name is:
	`C:\JDeveloper\mywork\SFE\SFEOutboundWebService_GlobalWeather.log`
	For outbound Web service Java Messaging Service (JMS) calls, the resource adapter log is updated. The location of the log file and the log level are configured when you deploy the resource adapter.
	NOTE: Make sure that a drive exists on the computer where Oracle WebLogic Server is configured. For security purposes, the run-time log files are created on Oracle WebLogic Server.

Outbound Log Level	The log level for the Siebel outbound log. The options are the same as the other log levels in SAI for Oracle Fusion Middleware: FATAL, ERROR, WARN, INFO, DETAIL, and DEBUG. By default, this value is set to ERROR. For more information about log levels, see "Setting Log Levels for SAI for Oracle Fusion Middleware" on page 118.
Siebel Language Pack	Lists the installed language packs. Select the appropriate language for your environment. By making this selection, the correct JAR file is used for the deployment. For example, if FRA is selected, then SiebelJI_fra.jar is used for the deployment.

5 Click OK.

Related Topics

"Installing SAI for Oracle Fusion Middleware" on page 28

Modifying Database Parameters for Use with SAI for Oracle Fusion Middleware

This topic describes how to modify database parameters for use with SAI for Oracle Fusion Middleware.

NOTE: If you are upgrading, then perform this step only if you are using a different database from your initial installation.

This task is a step in "Roadmap for Installing and Configuring SAI for Oracle Fusion Middleware" on page 24.

To modify database parameters for use with SAI for Oracle Fusion Middleware

1. Copy the driver JAR files to the Oracle JDeveloper home directory (`JDEV_HOME\jdeveloper\jdev\extensions\oracle.jdeveloper.siebel.eai.fusion`) by doing the following:

   - For MSSQL, copy the sqljdbc4.jar file from the MSSQL installation folder.

   - For DB2, copy the db2jcc.jar, db2jcc_license_cisuz.jar, and db2jcc_license_cu.jar files from the DB2 installation folder.

2. In Oracle JDeveloper, modify the connection type by doing the following:

   a. From the application-level menu, select the Tools menu, and then the Preferences menu item.

   b. Scroll down to select Siebel Application Integration for Oracle Fusion Middleware.

   c. In the General tab, select the appropriate connection type, and make sure the other fields are populated correctly.

      The fields are described in "Manually Installing Jersey Libraries for SAI for Oracle Fusion Middleware" on page 38.

   d. Click Test Connection.

      A message appears indicating the connection was successful or not.

   NOTE: For a DB2 connection, the service name is case sensitive.

Process of Configuring SAI for Oracle Fusion Middleware for Outbound Web Services

After you install SAI for Oracle Fusion Middleware, you must perform configuration tasks before you can use outbound Web services. This topic provides the process for configuring SAI for Oracle Fusion Middleware for outbound Web services.

This process is a step in "Roadmap for Installing and Configuring SAI for Oracle Fusion Middleware" on page 24.

NOTE: This process is only applicable if you are using outbound Web services in your deployment.

To configure SAI for Oracle Fusion Middleware for Siebel Business Applications, perform the following:

1. "Configuring the Siebel Server to Use SAI for Oracle Fusion Middleware for Outbound Web Services" on page 34
2. "Configuring Oracle WebLogic Server for Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 37
3. "Configuring Oracle JDeveloper for Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 37

After you complete these tasks, you can deploy new outbound Web services by using SAI for Oracle Fusion Middleware.

Configuring the Siebel Server to Use SAI for Oracle Fusion Middleware for Outbound Web Services

This topic describes how to configure the Siebel Server to use SAI for Oracle Fusion Middleware for outbound Web services. It describes how to set the CLASSPATH, VMOPTIONS, and DLL parameters for the EAIOutboundJvmSubsys JVM (Java Virtual Machine) outbound named subsystem.

This task is a step in "Process of Configuring SAI for Oracle Fusion Middleware for Outbound Web Services" on page 33.

NOTE: This task is only applicable if you are using outbound Web services in your deployment.

The parameters are described in the following table:

Parameter	Description
CLASSPATH	It sets the JAR file names in the classpath so that the Siebel Server JVM loads the required classes.
VMOPTIONS	It sets the EAIOutboundlogFile and EAIOutboundlogLevel parameters for the Siebel outbound Web service log file. **TIP:** You can also use VMOPTIONS to pass extra arguments that are related to the JVM, for example, heap size, stack size, and so on.
DLL	It sets the jvm.dll that handles the JVM on a Siebel Server. That is, the value in this parameter points to the DLL on a Windows computer that handles the JVM for the Siebel Server.

To configure Siebel Business Applications to use SAI for Oracle Fusion Middleware

1. Navigate to the Administration - Server Configuration screen, Enterprises, and then the Profile Configuration view.
2. In the Profile Configuration list, query the Alias field for:

   ```
   EAIOutboundJvmSubsys OR EAIOutboundSubsys
   ```

3 Provide values for the EAIOutboundJvmSubsys profile (named subsystem) parameters as shown in the following table.

Parameter	Value
Classpath	`SIEBEL_HOME\CLASSES\Siebel.jar;SIEBEL_HOME\CLASSES\jaxws-rt.jar;SIEBEL_HOME\CLASSES\wlfullclient.jar;SIEBEL_HOME\CLASSES\SiebelJI_language.jar;SIEBEL_HOME\CLASSES\xml.jar;SIEBEL_HOME\CLASSES\xmlparserv2.jar;SIEBEL_HOME\CLASSES\webserviceclient+ssl.jar` where: ■ *SIEBEL_HOME* is the installation directory for Siebel Business Applications software. ■ xml.jar and xmlparserv2.jar come from the Oracle WebLogic Server installation directory (*ORACLE_HOME*). ■ SiebelJI_*language*.jar is specific to your environment. ■ webserviceclients+ssl.jar is specific to SSL configuration. **TIP:** For Windows, use a semicolon (;) as the separator; for UNIX, use a colon (:).
DLL	`path_DLL_uses_to_set_load_jvm.dll` **NOTE:** This value is the path pointing to the jvm.dll that is loaded by the Siebel Server to handle the JVM of the Siebel Server.
VMOPTIONS	`-Xrs -Djava.compiler=NONE -DEAIOutboundlogFile=EAIOutbound -DEAIOutboundlogLevel=DEBUG`

4 Provide values for the EAIOutboundSubsys profile (named subsystem) parameters as shown in the following table.

Parameter	Value	Description
EAI Outbound Context Factory	`weblogic.jndi.WLInitial Context Factory`	An initial context factory class that obtains the reference to an Enterprise Java Bean deployed for each outbound proxy service on Oracle WebLogic Server.
EAI Outbound Provider URL	`t3:// weblogicserver:7101`	URL used to connect to Oracle WebLogic Server. The communication protocol is t3.
EAI Outbound Security Credentials	`********* (asterisks)`	The password used for authentication on Oracle WebLogic Server.
EAI Outbound Security Principal	`weblogic`	The user name used for authentication on Oracle WebLogic Server.
EAI Outbound ConnectionFactory	`Connection Factory JNDI Name`	If the transport type is JMS, then this parameter specifies the queue connection factory JNDI name.
EAI Outbound ReceiveQueue	`Receive Queue JNDI Name`	If the transport type is JMS, then this parameter specifies the receive queue JNDI name from which messages are read by the Siebel application.
EAI Outbound ReceiveTimeout	`3000`	If the transport type is JMS, then this parameter specifies the time the JMS service waits for the response to be received from Oracle WebLogic Server before the server times out.
EAI Outbound SendQueue	`Send Queue JNDI Name`	If the transport type is JMS, then this parameter specifies the send queue JNDI name on which messages will be sent by the Siebel application.
EAI Outbound Transport Type	`RMI or JMS`	This parameter specifies the communication type between the Siebel application and Oracle WebLogic Server. Transport types supported are RMI and JMS.
EAI Outbound Server Type	Weblogic	This parameter specifies the application server.

NOTE: The following parameters are specific to JMS: EAIOutboundConnectionFactory, EAIOutboundReceiveQueue, and EAIOutboundSendQueue.

Configuring Oracle WebLogic Server for Outbound Web Services Using SAI for Oracle Fusion Middleware

This topic describes how to create a new domain for Oracle WebLogic Server for the proxy computer to enable outbound Web services.

This task is a step in "Process of Configuring SAI for Oracle Fusion Middleware for Outbound Web Services" on page 33.

NOTE: This task is only applicable if you are using outbound Web services in your deployment.

To configure Oracle WebLogic Server for outbound Web services using SAI for Oracle Fusion Middleware

1. Create a new domain on Oracle WebLogic Server.

   For information about creating domains, see *Oracle Fusion Middleware Creating Domains Using the Configuration Wizard 11g Release 1 (10.3.5)* in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

2. Set up the Oracle WebLogic Serverr proxy for Oracle JDeveloper by doing the following:

   a. Navigate to the bin folder of that domain (for example, C:\oracle\Middleware\user_projects\domains*domain_name*\bin)

   b. In the startWebLogic.cmd file replace JAVA_OPTIONS=%SAVE_JAVA_OPTIONS% with:

      ```
      set JAVA_OPTIONS=%SAVE_JAVA_OPTIONS% -Dhttp.proxyHost=proxy_server_name -
      Dhttp.proxyPort=proxy_server_port
      ```

      For example:

      ```
      set JAVA_OPTIONS=%SAVE_JAVA_OPTIONS% -Dhttp.proxyHost=www-
      proxy.us.mycompany.com - Dhttp.proxyPort=80
      ```

3. Restart Oracle WebLogic Server.

Configuring Oracle JDeveloper for Outbound Web Services Using SAI for Oracle Fusion Middleware

This topic describes how to configure Oracle JDeveloper for outbound Web services using SAI for Oracle Fusion Middleware.

This task is a step in "Process of Configuring SAI for Oracle Fusion Middleware for Outbound Web Services" on page 33.

NOTE: This task is applicable only if you are using outbound Web services in your deployment.

The Log File Directory that you enter in SAI for Oracle Fusion Middleware Deployment preferences includes the log files related to any outbound Web service Remote Method Invocation (RMI) calls.

NOTE: For outbound Web service JMS calls, the resource adapter log file is updated. You enter the location and log level when you deploy a Siebel Resource Adapter. Validate the drive of the computer where Oracle WebLogic Server is hosted because these run-time log files are created for security reasons.

To configure Oracle JDeveloper for outbound Web services using SAI for Oracle Fusion Middleware

1. Log in to Oracle JDeveloper.
2. (URL-based WSDLs only) From the Tools menu, select the Preferences menu item.
3. (URL-based WSDLs only) Scroll down in the explorer to select Web Browser and Proxy, click the Use HTTP Proxy Server check box, and then fill in the following parameters:

   - The host name of the HTTP proxy server
   - The port number of the HTTP proxy server

   For information about the other Web browser and proxy parameters, see "Setting Preferences for SAI for Oracle Fusion Middleware" on page 29.

4. In the explorer, select Siebel Application Integration for Oracle Fusion Middleware.
5. Click the Deployment tab, and provide the value for the Log File Directory parameter.

   For information about this parameter, see Step 7 on page 56 in "Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware."

   NOTE: You might also want to set the log levels at this time, but you can change the levels at any time.

Manually Installing Jersey Libraries for SAI for Oracle Fusion Middleware

Jersey libraries are called by Siebel RESTful services at runtime. Before you can work with Siebel RESTful Services, you must deploy the Jersey libraries that are provided as part of the Oracle WebLogic Server installation. After you install the Jersey libraries, they are available for use in the Oracle WebLogic Server Administration Console. For information about Siebel RESTful Services, see Chapter 7, "Working with Siebel RESTful Services."

This task is a step in "Roadmap for Installing and Configuring SAI for Oracle Fusion Middleware" on page 24. It is applicable only to first-time installations; it is not applicable to upgrades. Perform this task for each Oracle WebLogic Server you install.

To manually install Jersey libraries for SAI for Oracle Fusion Middleware

1. Log in to the Oracle WebLogic Server Administration Console.
2. In the Domain Structure, select Deployments.

3. Click Lock & Edit in the Change Center.
4. Click Install, and then click Next.
5. Specify `*WLS_HOME*\wlserver_10.3\common\deployable-libraries` in the path.

   A list of the components that you can install appears.

6. For each of the files below, do the following:

   - jersey-bundle-1.1.5.1.war
   - jsr311-api-1.1.1.war

   a. Select the file, and then click Next
   b. Choose to install this as a library, and then click Next.
   c. Choose optional settings, and then click Finish.

Logging In to the Oracle WebLogic Server Administration Console

After you have installed Oracle WebLogic Server, make sure that you can log in to the Oracle WebLogic Server Administration Console. For more information about this console, see "Oracle WebLogic Server Administration Console" on page 17.

NOTE: Before performing the following procedure, make sure Oracle WebLogic Server is running.

To log in to the Oracle WebLogic Server Administration Console

1. From a browser window, enter the URL for your deployment in the Address field by using the following syntax:

   http://*hostname*:*port*/console/login/LoginForm.jsp

   For example:

   ```
   http://myhost.mycompany.com:7001/console/login/LoginForm.jsp
   ```

   NOTE: If you have configured Oracle WebLogic Server in SSL mode, then the URL is: https://hostname:SSL-port/console/login/LoginForm.jsp. For more information configuring Oracle WebLogic Server, see the documentation in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

2. Log in using the Oracle WebLogic Server user name and password.

   The Oracle WebLogic Server Administration Console home page appears.

Related Topic

"Oracle WebLogic Server Administration Console" on page 17

About Upgrading SAI for Oracle Fusion Middleware

If you have an existing SAI for Oracle Fusion Middleware environment and you want to use the latest features, then you must upgrade to the latest Siebel CRM Fix Pack 8.1.1.x and perform additional configuration tasks for both the Siebel application and SAI for Oracle Fusion Middleware. For instructions, see "Roadmap for Installing and Configuring SAI for Oracle Fusion Middleware" on page 24.

4 Setting Up Security for SAI for Oracle Fusion Middleware on Oracle WebLogic Server

This chapter describes how to set up security for Siebel Application Integration for Oracle Fusion Middleware (SAI for Oracle Fusion Middleware) on Oracle WebLogic Server. It includes the following topics:

About Oracle WebLogic Server Security Realms

A security realm comprises mechanisms for protecting Oracle WebLogic Server resources. Each security realm consists of a set of configured security providers, users, groups, security roles, and security policies. You must define a user in a security realm to access any WebLogic resources belonging to that realm. When a user attempts to access a particular WebLogic resource, Oracle WebLogic Server tries to authenticate and authorize the user by checking the security role assigned to the user in the relevant security realm and the security policy of the particular WebLogic resource.

You can configure multiple security realms in a domain, however, only one can be the default (active) security realm. Oracle WebLogic Server provides one default security realm: myrealm. For more information about security realms, see *Oracle® Fusion Middleware Understanding Security for Oracle WebLogic Server* 11*g* Release 1 available in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

About the Security Models for SAI for Oracle Fusion Middleware

The two security models available for SAI for Oracle Fusion Middleware are:

- Deployment Descriptor Only (DD Only)
- Custom Roles and Policies

You choose a security model when you deploy each Web application or Enterprise Java Bean (EJB), and your choice is immutable for the lifetime of the deployment. If you want to use a different model, then you must delete and redeploy the Web application or EJB. For more information about the security models, see *Oracle® Fusion Middleware Securing Resources Using Roles and Policies for Oracle WebLogic Server* available in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

Setting Up Security on Oracle WebLogic Server for SAI for Oracle Fusion Middleware

This topic describes how to set up security for SAI for Oracle Fusion Middleware on Oracle WebLogic Server.

To set up security for SAI for Oracle Fusion Middleware

1. Navigate to the Oracle WebLogic Server Administration Console.
2. Under Domain Structure, select Security Realms.
3. In the Realms table, drill down on myrealm or create a new realm and name it *SiebelRealm*.
4. Create a user group or groups in the security realm by doing the following:
   a. In the Users and Groups tab, click the Groups tab, and then click New.
   b. Define and name a group or groups, and then click OK.

      For example, you might create a new group called *SiebelResourceGroup.*
5. Create users in the security realm by doing the following:
   a. In the Users and Groups tab, click the Users tab, and then click New.

      For example, you might create two users: *tester* and *siebeladmin*.

   b. Add the new users as members of the new group you created in Step 4.

      For more information, see the topic about adding users to groups in *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Online Help*.
6. If you have not yet configured a resource adapter, then configure one now.

   For information about configuring resource adapters, see "Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware" on page 46.
7. Add a new policy that grants group access by doing the following:
   a. Under Domain Structure, click Deployment, and then drill down on the applicable deployment.
   b. Click Security, click Policies, and then add a new policy.
   c. Click Add Conditions, and then select group from the predicate list, and then click Next.
   d. Specify the user group you created in Step 4, click Add, and then wait for the group to appear in the text box.
   e. Click Finish, and then click Save.
8. Create credential maps between the Oracle WebLogic Server users and Siebel users by doing the following:
   a. Under Domain Structure, click Deployment, and then drill down on the applicable deployment.

      In this case, the deployment is SiebelResourceAdapter.

b Choose Security, then Credential Mapping, then drill down on SiebelResourceAdapter.

c Click New, then choose the resource adapter to create a new security credential map.

For example, eis/*JNDI_ra_name.*

where:

- *eis* is a required text string.
- *JNDI_ra_name* is a text string that you choose that describes your deployment.

The default is *eis/siebel/SiebelResourceAdapterConnFactory*.

NOTE: If the resource adapter is deployed with a JNDI name of eis/siebel/SiebelResourceAdapter, then in the REST URL where the JNDI name of the resource adapter is specified, you must enter siebel.SiebelResourceAdapter (note the period). The following variations do not work:

siebel/SiebelResourceAdapter
eis/Siebel.siebelResourceAdapter
eis.siebel.siebelresourceadapter

d Select the Configured User Name option, and then click Next.

e Choose an existing Oracle WebLogic user who is defined in the default security realms, and then click Next. For example, you might choose *tester*.

f Enter the EAI user name and password. For example, the user name might be *PERFUSER* and the password might be *PERFUSER*.

9 Repeat Step 8 to map additional users.

For example, you might want to map the Oracle WebLogic user *admin* to the Siebel resource user *SADMIN*, and so on.

10 (Optional) Enable and examine the troubleshooting information.

For more information, see “Enabling and Examining Troubleshooting Information on the Oracle WebLogic Server” on page 117.

5 Deploying Siebel Resource Adapters by Using SAI for Oracle Fusion Middleware

This chapter provides information about Siebel Resource Adapters and how to deploy them using SAI for Oracle Fusion Middleware. It includes the following topics:

About Resource Adapters

A *resource adapter* is a deployed adapter that communicates with an Enterprise Information System (EIS). The Siebel resource adapter is a system library specific to an EIS that provides connectivity to an EIS. It enables the external application to participate and function in one or more integration patterns without significant changes to the application. The primary integration patterns include:

- Data synchronization
- Online services
- Process automation

These three patterns are generic and cover most technologies, including mobile computing, E-Commerce, Web services, business process automation, supply chain automation, data integrity, and business intelligence. However, you can extend these patterns to include other integration patterns.

In SAI for Oracle Fusion Middleware, the Siebel Resource Adapter is the core Java EE component that provides the link between Siebel Business Applications and Oracle WebLogic Server and is compliant with the J2EE 1.5 Connector Architecture (JCA) specification.

For more information about:

- Resource adapters used in Oracle products, see *Oracle® Fusion Middleware Programming Resource Adapters for Oracle WebLogic Server 11g Release 1 (10.3.1)* in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.
- The Siebel Resource Adapter that is used for deploying SAI for Oracle Fusion Middleware and Siebel RESTful Services, see *Integration Platform Technologies: Siebel Enterprise Application Integration*.
- Deploying Siebel Resource Adapters, see "Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware" on page 46.

About Siebel Resource Adapter Naming Conventions

If the JNDI name is *siebel/SiebelResourceAdapter,* then *siebel* and *SiebelResourceAdapter* are two different names that map as different inputs by using different annotations. However, if the name is separated by a period (.), then the name is considered to be single, and it is mapped to a single input.

About Siebel Resource Adapter Log Files

Each Siebel Resource Adapter has its own log file. The log file name is the same name as the resource adapter name appended with .log; that is, ra_name.log. The location of the log file depends on the value that you specify in the Siebel Resource Adapter Log Directory logging parameter for the Siebel Server. For information about this parameter, see Step 7 on page 46 in "Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware."

Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware

The resource adapter is the core component that provides the link between Siebel Business Applications and the Oracle WebLogic Server for SAI for Oracle Fusion Middleware. You use the Deployment wizard in SAI for Oracle Fusion Middleware to deploy resource adapters. A many-to-many relationship exists between resource adapters and Siebel instances. For more information about resource adapters, see "About Resource Adapters" on page 45.

To deploy a resource adapter using SAI for Oracle Fusion Middleware, perform the following tasks:

1. Make sure the Oracle WebLogic Server is running.
2. Start Oracle JDeveloper if it is not already active.
3. Select File, and then click the New menu item.
4. In the New Gallery dialog box, perform the following:
   a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.
   b. Select the Deployment item, and then click OK.
5. At the Welcome to the Deployment Wizard screen, click Next.
6. Select Siebel Resource Adapter as the deployment type, and then click Next.
7. Enter the resource adapter parameters by doing the following:
   a. Enter the Siebel Resource Adapter Name, if you do not want to accept the default.

      By default, the resource adapter name is SiebelResourceAdapter.

b Click Edit ra.xml to enter the XML parameters.

Some settings are described in the following table.

NOTE: The Advanced Settings enable you to configure Siebel Server load balancing and Secure Sockets Layer (SSL). The Outbound Resource Adapter settings apply to outbound Web services with JMS and the transport type.

Parameter	Description
Siebel Server Parameters	
Siebel Server Host Name	The name of your Siebel Server. By default, this parameter is set to localhost.
Port	The listening port for the SCBroker server component. By default, this parameter is set to 2321. **NOTE:** This field is ignored when using load balancing on the Siebel Server.
User Name	The name of the Siebel user that is configured for default connections. Use this user name for creating the credential map. Credential mapping is the process whereby a legacy database is used to obtain an appropriate set of credentials to authenticate users with a target resource. Oracle WebLogic Server uses credential mapping to map credentials used by Oracle WebLogic Server users to credentials used in a legacy or remote system. Oracle WebLogic Server then uses the credential maps to log in to a remote system on behalf of a user who has already been authenticated. For more information on credential mapping, see "About Protecting User Names and Passwords for SAI for Oracle Fusion Middleware" on page 52. **NOTE:** It is recommended that you assign a user name with minimal credentials, for example, you might assign an anonymous user.
EAI OM Alias	The alias for the Enterprise Application Integration Object Manager. By default, this parameter is set to EAIObjMgr_enu. Keep this setting unless you have configured the object manager with a different alias.
Enterprise Name	The name of your Siebel Enterprise Server. By default, this parameter is set to siebel. Keep this setting unless you have a different name for your enterprise.
B2BMode	This mode enables a connection between Oracle WebLogic Server and the Siebel Server to operate under the auspices of a service user; that is, no impersonation occurs and transactions are run in the context of the service user. By default, B2B mode is not selected, which means requests are executed in B2C mode. For more information, see "About Protecting User Names and Passwords for SAI for Oracle Fusion Middleware" on page 52. You must specify a B2B User Name if this check box is selected.

Parameter	Description
B2B User Name	The service user name. Check the B2B Mode check box to enter the user name. Use this user name for creating the credential map. For more information about credential mapping, see “About Protecting User Names and Passwords for SAI for Oracle Fusion Middleware” on page 52.
Advanced Settings	
Session Timeout	The number of minutes a session can remain inactive before a user is logged out and the session is closed. By default, this parameter is set to 15 minutes.
Connections per EAI OM	Specifies the number of physical sockets that are created from the resource adapter to the Enterprise Application Integration Object Manager process.
Session Token Max Age	The total number of minutes that a session can remain open before a user is logged out and the session is closed. By default, this parameter is set to 2880 minutes.
Session Token Timeout	The number of minutes that a session can be idle before the session is locked. By default, this parameter is set to 15 minutes.
JNDI Cache Duration	The value of this parameter sets the cache duration for the Java Naming and Directory Interface (JNDI) object. The cached JNDI object is reused for subsequent inbound requests. By default, this parameter is set to 15 minutes.
Max Anonymous User	Specifies the maximum number of Web service sessions for anonymous users that are permitted. By default, this parameter is set to 10.
Transport	Specifies the type of transport protocol between Oracle WebLogic Server and the Siebel Server. The choices are: TCP and SSL. By default, this parameter is set to TCP. If you choose SSL, then a Parameters option appears that you can click to specify the SSL-related parameters. For more information about setting these parameters, see “Configuring Siebel Server Load Balancing and SSL for SAI for Oracle Fusion Middleware” on page 55.
Encryption	Specifies whether encryption is used for data transmission from Oracle WebLogic Server and the Siebel Server. The choices are: None and RSA. By default, this parameter is set to None.
Compression	Specifies whether compression is used for data transmission from Oracle WebLogic Server and the Siebel Server. The choices are: None and zlib. By default, this parameter is set to None.
Virtual Hosts	Lists the available virtual servers. Use this parameter when you have more than one Siebel Server. The following is a sample value that you might enter: `VirtualServer1=1:localhost:3320,2:sdchs20i183:3320;VirtualServer2=1:xyz.corp.oracle.com:2321,2:ABC.corp.oracle.com:2321`

Parameter	Description
Logging Parameters	
Siebel Resource Adapter Log Directory	The directory where the Siebel Resource Adapter log files are stored. By default, this parameter is set to: `C:\JDeveloper\mywork\SFE` or *default_JDeveloper_work_directory*\SFE where the default log file name is ResourceName.log. The ResourceName.log file contains the run-time logging files during the execution of inbound Web service requests, inbound Enterprise Java Bean requests, and Siebel RESTful Services requests. Use this log file to troubleshoot any run-time issues. ■ If you deploy Siebel Resource Adapter by using message-driven beans, then two more log files are stored in the preceding location: ■ JCAReceiver.log ■ JCASender.log **NOTE:** If your deployment is on a remote computer, then make sure Oracle WebLogic Server can access this directory. **TIP:** It is recommended that you also check the Oracle WebLogic Server Administration Console for additional, log file information.
Log Level	The log level determines the amount of information that is written to the log file. The options are the same as the other log levels in SAI for Oracle Fusion Middleware: FATAL, ERROR, WARN, INFO, DETAIL, and DEBUG. By default, the log level is set to ERROR. For more information about log levels, see "Setting Log Levels for SAI for Oracle Fusion Middleware" on page 118.
Outbound Resource Adapter Configuration	
Queue Connection Factory	The JNDI name for the connection factory queue created on the Oracle WebLogic Server Administration Console.
Receive Queue	Specify the JNDI name for the configured queue on Oracle WebLogic Server. **NOTE:** You use the ReceiveQueue and SendQueue parameters to pass the JNDI name to Oracle WebLogic Server. The JNDI name for ReceiveQueue is the same as SendQueue in the Siebel Server named subsystem.
Send Queue	Specify the JNDI name for the configured queue on Oracle WebLogic Server. **NOTE:** You use the SendQueue and ReceiveQueue parameters to pass the JNDI name to Oracle WebLogic Server. The JNDI name for SendQueue is the same as ReceiveQueue in the Siebel Server named subsystem.

Parameter	Description
MDB Parameters	
Initial Free Pool Size	The initial number of beans in the free pool. By default, this value is set to 5.
Maximum Free Pool Size	The maximum number of beans in the free pool. When an Enterprise Java Bean has been in the free pool for the number of seconds specified in Idle Timeout, and the total number of beans in the free pool approaches the maximum number of beans in the free pool specified in this field, then the idle beans are removed from the free pool. By default, this value is set to 10.

c Click Edit weblogic-ra.xml to verify the Java Naming and Directory Interface (JNDI) names, and then click OK.

Some settings are described in the following table.

Parameter	Description
Resource Adapter JNDI Name	This parameter specifies the JNDI name for the resource adapter. The resource adapter bean is registered with this name in the JNDI tree. By default, this parameter is set to: eis/siebel/SiebelResourceAdapter
Connection Factory JNDI Name	This parameter is the JNDI name for the connection factory that the resource adapter uses. Use this JNDI name for deploying Siebel RESTful Services. By default, this parameter is set to: eis/siebel/SiebelResourceAdapterConnFactory

d Choose one of the following security models, and then click Next.

- DDOnly
- CustomRolesAndPolicies

NOTE: By default, the model is set to DDOnly. If you want to use Oracle WebLogic Server security, then you must choose CustomRolesandPolicies, and also perform some additional configuration. For more information, see Chapter 4, “Setting Up Security for SAI for Oracle Fusion Middleware on Oracle WebLogic Server.”

8 Enter the Oracle WebLogic Server parameters for your environment, and then click Next.

Some of the parameters are described in the following table.

Oracle WebLogic Server Parameter	Description
Host Name	The name of the server where Oracle WebLogic Server is running. By default, this parameter is set to localhost.
Port	The port number on which Oracle WebLogic Server is configured. By default, this parameter is set to 7001.
SSL Port	The Secure Sockets Layer (SSL) port number on which Oracle WebLogic Server is configured. By default, this parameter is set to 7002. This setting is available only if you have chosen the Use SSL for this Deployment option.
SSL Parameters	Click to specify the truststore parameters. This setting is available only if you have chosen the Use SSL for this Deployment option.
Oracle WebLogic Administration Server	The name of the Oracle WebLogic Administration Server. By default, this parameter is set to AdminServer. **TIP:** To find the name of your Oracle WebLogic Server, use the Oracle WebLogic Server Administration Console to select Environment, Deployments, and then Servers.
Username	The administrator user name for Oracle WebLogic Server.
Password	The administrator password for Oracle WebLogic Server.

9 Verify the summary and read the applicable notes, and then click Finish to deploy the resource adapter.

Information about successful and failed resource adapter deployments appears.

10 Follow the procedure in "Configuring Credential Mapping for SAI for Oracle Fusion Middleware on Oracle WebLogic Server" on page 53.

11 Set the deployment order of the resource adapter to a value that is less than 100 by doing the following:

a Drill down on the resource adapter.

b In the Overview tab, find Deployment Order, enter a number less than 100, and then click Save.

NOTE: The deployment order is an integer value that indicates when this unit is deployed during startup, relative to other units on a server. The units with lower values are deployed before those with higher values. If the value is the same for two units, then the alphabetical order sorted by the deployment name is applied. Changes take effect after you redeploy the module or restart the server.

CAUTION: The order of deployment matters because if an inbound Web service or Siebel RESTful service becomes active without an active resource adapter, then the deployment fails.

About Protecting User Names and Passwords for SAI for Oracle Fusion Middleware

For SAI for Oracle Fusion Middleware, you use credential mapping to protect Enterprise Information System (EIS) user names and passwords that make the initial connection pool to the Siebel Server. You configure credential mapping for SAI for Oracle Fusion Middleware by way of the Resource Adapter Descriptor Parameters in the Deployment wizard. If your deployment supports one of the following options, then do the following:

- **B2C (Business-to-Customer).** Provide the user name under the Siebel Server Parameters tab.
- **B2B (Business-to-Business).** Provide the B2B user name under the Siebel Server Parameters tab.

The names that you provide are used in the credential map to create the initial connection to the Siebel Server. The same credential map is also used to create new connections if the existing connections are busy. Oracle WebLogic Server refers to the ra.xml file and searches for the appropriate user name to create the new connections to the Siebel Server. For information on configuring credential mapping for SAI for Oracle Fusion Middleware, see “Process of Configuring Credential Mapping for SAI for Oracle Fusion Middleware” on page 52.

NOTE: By default, Oracle WebLogic Server deployment parameters are set to zero initial connections. Because you can create credential mapping only after a resource adapter is deployed, you must increase this initial connection capacity to greater than zero. That way, Oracle WebLogic Server uses the Siebel Server user name and password settings from the credential map to create the connections.

Related Topics

“Process of Configuring Credential Mapping for SAI for Oracle Fusion Middleware” on page 52

“Configuring Credential Mapping for SAI for Oracle Fusion Middleware on Oracle WebLogic Server” on page 53

“Reconfiguring the Connection Pool on Oracle WebLogic Server for SAI for Oracle Fusion Middleware” on page 54

Process of Configuring Credential Mapping for SAI for Oracle Fusion Middleware

SAI for Oracle Fusion Middleware supports credential mapping to protect Enterprise Information System (EIS) user names and passwords. For more information about this protection, see “About Protecting User Names and Passwords for SAI for Oracle Fusion Middleware” on page 52.

To configure credential mapping for SAI for Oracle Fusion Middleware, perform the following tasks:

1 Make sure you have configured a Siebel Resource Adapter with the appropriate user name and passwords for your deployment.

   For information about configuring Siebel Resource Adapters, see "Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware" on page 46.

2 Create credential mapping on Oracle WebLogic Server to establish connections to the Siebel Server.

   For information, see "Configuring Credential Mapping for SAI for Oracle Fusion Middleware on Oracle WebLogic Server" on page 53.

3 Reconfigure the connection pool on Oracle WebLogic Server.

   For information, see "Reconfiguring the Connection Pool on Oracle WebLogic Server for SAI for Oracle Fusion Middleware" on page 54.

Related Topics

"About Protecting User Names and Passwords for SAI for Oracle Fusion Middleware" on page 52

"Configuring Credential Mapping for SAI for Oracle Fusion Middleware on Oracle WebLogic Server" on page 53

"Reconfiguring the Connection Pool on Oracle WebLogic Server for SAI for Oracle Fusion Middleware" on page 54

Configuring Credential Mapping for SAI for Oracle Fusion Middleware on Oracle WebLogic Server

Use the following procedure to configure credential mapping for SAI for Oracle Fusion Middleware on Oracle WebLogic Server.

This task is a step in "Process of Configuring Credential Mapping for SAI for Oracle Fusion Middleware" on page 52.

To configure credential mapping for SAI for Oracle Fusion Middleware on Oracle WebLogic Server

1 Log in to the Oracle WebLogic Server Administration Console.

2 In the Deployment domain, select Deployments.

3 Select the Siebel Resource Adapter for which you want to configure credential mapping.

4 Click the Security tab, and then click the Credential Mappings tab.

5 Select New to create a new mapping.

6 Select the appropriate Outbound Connection Pool, and then click Next.

   For example, if you chose SiebelResourceAdapter above, then you might want to choose eis/siebel/SiebelResourceAdapterConnFactory.

7 Select Default User, and then click Next.

8 Enter the EIS user name and password to create the initial new connections, and then Click Finish.

NOTE: Enter the same user name and password that you specified in the Resource Adapter Descriptor parameters during the Siebel Resource Adapter deployment. For information about the Resource Adapter Descriptor parameters, see Step 7 on page 46 in "Configuring Siebel Server Load Balancing and SSL for SAI for Oracle Fusion Middleware."

A message appears indicating the new security map entry for the resource adapter was successfully created.

Related Topics

"About Protecting User Names and Passwords for SAI for Oracle Fusion Middleware" on page 52

"Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware" on page 46

"Reconfiguring the Connection Pool on Oracle WebLogic Server for SAI for Oracle Fusion Middleware" on page 54

Reconfiguring the Connection Pool on Oracle WebLogic Server for SAI for Oracle Fusion Middleware

Use the following procedure to reconfigure the connection pool on Oracle WebLogic Server for SAI for Oracle Fusion Middleware.

This task is a step in "Process of Configuring Credential Mapping for SAI for Oracle Fusion Middleware" on page 52.

To reconfigure the connection pool on Oracle WebLogic Server for SAI for Oracle Fusion Middleware

1 Log in to the Oracle WebLogic Server Administration Console.

2 In the Deployment domain, select Deployments.

3 Select the Siebel Resource Adapter for which you want to configure credential mapping.

For example, you might choose SiebelResourceAdapter.

4 Click the Configuration tab, the Outbound Connections Pools tab, and then drill down on the outbound connection pool instance for this resource adapter.

5 Check the properties and compare them with the information you entered for the Resource Adapter Descriptor Parameters for the Siebel Server (the ra.xml file).

NOTE: Make sure the same user name and password appears that you specified in the Resource Adapter Descriptor parameters during the Siebel Resource Adapter deployment. For information about the Resource Adapter Descriptor parameters, see Step 7 on page 46 in "Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware."

6 Click the Connection Pool tab, click Lock & Edit in the Change Center, and then enter the Initial Capacity and Max Capacity values that you want.

7 Save the changes, and click OK to apply them.

Oracle WebLogic Server creates the connection pool by using the credential mappings values that you entered.

8 If the task is successful, then you can verify the connections by using Siebel Server Manager (srvrmgr) and running the following command:

```
List active task for comp Siebel_object_manager_language
```

Related Topics

"About Protecting User Names and Passwords for SAI for Oracle Fusion Middleware" on page 52

"Process of Configuring Credential Mapping for SAI for Oracle Fusion Middleware" on page 52

"Reconfiguring the Connection Pool on Oracle WebLogic Server for SAI for Oracle Fusion Middleware" on page 54

Configuring Siebel Server Load Balancing and SSL for SAI for Oracle Fusion Middleware

This topic describes how to configure the XML Advanced Settings parameters for Siebel Server load balancing and Secure Sockets Layer (SSL) encryption and authentication for inbound Web services by using SAI for Oracle Fusion Middleware. For more information about Siebel Server load balancing, see *Siebel Deployment Planning Guide*. For more information about SSL, see *Siebel Security Guide*.

NOTE: SAI for Oracle Fusion Middleware supports SSL for outbound Web services as well, but load balancing is not supported.

Typically, you perform the following procedure as part of deploying a Siebel Resource Adapter. However, you can reset these parameters.

This task is a step in "Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware" on page 46.

To configure Siebel Server load balancing and SSL for SAI for Oracle Fusion Middleware

1 Make sure the Oracle WebLogic Server is running.

2 Start Oracle JDeveloper if it is not already active.

3 Select File, and then click the New menu item.

4 In the New Gallery dialog box, perform the following:

   a In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.

   b Select the Deployment item, and then click OK.

5 At the Welcome to the Deployment Wizard screen, click Next.

6. Select Siebel Resource Adapter as the deployment type, and then click Next.

7. Enter the resource adapter parameters by doing the following:

   a. Enter a resource adapter name if you do not want to accept the default. By default, the resource adapter name is *SiebelResourceAdapter*.

      NOTE: Each resource adapter has its own log file. The log file name is the same name as the resource adapter name appended with .log (that is, *JNDI_ra_name*.log). The location of the log file depends on the value you specify in the Deployment Log Directory logging parameter in the SAI for Oracle Fusion Middleware preferences.

      For information about the naming conventions for resource adapters, see "About Siebel Resource Adapter Naming Conventions."

   b. Click Edit ra.xml, click Advanced Settings, and then edit or fill in the necessary settings.

      For information about the Advanced Settings parameters, see Step 7 in "Configuring Siebel Server Load Balancing and SSL for SAI for Oracle Fusion Middleware" on page 55.

8. (Optional) For SSL transport, configure the Keystore and Truststore parameters by doing the following:

   a. In the Transport field, select SSL, and then click Parameters.

   b. Fill in the necessary settings, click OK, and then click OK again to exit the Resource Adapters Descriptors Parameters dialog box.

      Some parameters are described in the following table. For more information about SLL transport, see *Siebel Security Guide* on the *Siebel Bookshelf*.

Parameter	Description
Keystore Name	The name of the Java Key Store that stores the certificate. Specify the path of the Java Key Store on Oracle WebLogic Server.
Keystore Password	The password to the keystore file.
Keystore Alias	The name of the alias of the certificate in the keystore.
Trust Store Name	The name of the Java Key Store that contains the trusted certificate authorizations. Specify the path of the Java Key Store truststore on Oracle WebLogic Server.
Trust Store Password	The password to access the truststore.

About Testing Your Siebel Resource Adapter Deployment

After you have deployed SAI for Oracle Fusion Middleware and applied the security configurations, test your deployment. If you have Siebel RESTful Services deployed, then you might want to test that deployment. For information about Siebel RESTful Services, see Chapter 7, “Working with Siebel RESTful Services.”

If you have not deployed Siebel RESTful Services, then try deploying another service type, and check its connectivity by using the Oracle WebLogic Server tools or an external Web service client.

6 Working with Web Services Using SAI for Oracle Fusion Middleware

This chapter describes how to deploy and test Web services using Siebel Application Integration for Oracle Fusion Middleware (SAI for Oracle Fusion Middleware). It includes the following topics:

About Oracle WebLogic Web Services

Oracle WebLogic Web services are implemented according to the Java EE 1.2 specification, which defines the standard Java Platform, Enterprise Edition (Java EE) runtime architecture for implementing Web services in Java. This specification also describes a standard Java EE Web service packaging format, deployment model, and runtime services, all of which are implemented by Oracle WebLogic Web services.

Compliance Standards for SAI for Oracle Fusion Middleware Web Services

This topic provides various standards with which Siebel Application Integration for Oracle Fusion Middleware (SAI for Oracle Fusion Middleware) is compliant.

Table 4 provides a comparison of Oracle Fusion Middleware technology artifacts with the Siebel Business Applications equivalents.

Table 4. Artifacts Comparison Between Oracle Fusion Middleware and Siebel Business Applications

Artifact	Oracle Fusion Middlware Technology	Siebel Business Applications Equivalent
SOAP version	1.1 and 1.2	1.1
JCA version	1.5	1.0
WS-Security version	1.1	1.0 (partial)
Soap binding	JAX-WS 2.1	DOC_LITERAL RPC_LITERAL RPC_ENCODED
Attachments	MTOM - XOP	Base64Binary
Programming model	WebService for Java EE 1.2	Proprietary
Asynchronous communication	WS-Addressing 1.0	N/A
Advertisement (registration and discovery)	UDDI 2.0 Java API for XML Registries (JAX-R) 1.0	N/A
Enterprise Java Beans version	3.0	N/A
JAX-RS for RESTful (Representational State Transfer) services	Supported	N/A
Data binding	Java Architecture for XML Binding (JAXB) 2.1	String data types

Table 5 provides a comparison of SAI for Oracle Fusion Middleware features with Siebel EAI equivalents.

Table 5. Feature Comparison of SAI for Oracle Fusion Middleware with Siebel EAI Equivalents

Features	Siebel EAI	SAI for Oracle Fusion Middleware	Customer Benefits
Integration development	Siebel Tools, C++	JDeveloper, Java	■ Integration based on standards and decoupled from Siebel-specific technologies ■ Reduced integration cost
Service design and extensibility	Repository, SRF dependent	Dynamic	■ Zero downtime for the Siebel enterprise ■ Ease of development and deployment
Service versioning	No in-built versioning	Fusion Middleware-based versioning	■ Compatible across releases ■ Low upgrade cost and risk
Service interoperability	Limited, no standard compliance	WS Basic Profile 1.1	■ Distributed coexistence ■ Platform, language, and application independent
Server-managed session pooling	None (client manages sessions)	Managed JCA connection pool based on impersonation	■ No reliability or performance concerns due to suboptimal client session handling ■ Not vulnerable to Denial of Service (DOS) attacks because sessions are managed by Oracle WebLogic Server rather than the client
Monitoring and management	■ Minimal monitoring supported using Server manager ■ No versioning capability	■ Oracle WebLogic Administration Console provides detailed user interface-based monitoring for all connections, requests, and parameters ■ Ability to switch between versions	■ Preconfigured diagnostics and management ■ Oracle Enterprise Manager compliant

About Working with Web Services Using SAI for Oracle Fusion Middleware

Siebel Application Integration for Oracle Fusion Middleware (SAI for Oracle Fusion Middleware) allows you to separate service-oriented architecture (SOA) integration development completely from your Siebel Business Applications implementation.

You can build Java classes or Enterprise Java Beans, expose them as Web services, automatically deploy them to an instance of Oracle WebLogic Server, and immediately test the running Web service. Alternatively, you can use Oracle JDeveloper to drive the creation of Web services from WSDL descriptions. Because Oracle JDeveloper also is Ant-aware, you can use this tool to build and run Ant scripts for assembling the client and for assembling and deploying the service.

About Base Integration Objects

Traditionally, Siebel EAI integrations specify an integration object as the input, the output, or both. With SAI for Oracle Fusion Middleware, the core requirement for an integration object is extended to accommodate a logical one-to-one relationship between a business object and an integration object. Each integration object in SAI for Oracle Fusion Middleware provides a representation of an underlying business object and its relationships, hiding the Siebel-specific ID fields and displaying the constituent fields of the underlying key to maintain the relationship. Such integration objects are called *base integration objects*.

There is a one-to-one mapping between business object and integration object; for each business object, there is one base integration object. XSDs are generated from the base integration object. You enforce the one-to-one mapping with a column in the business object repository that contains the base integration object name.

NOTE: Base integration objects are seed integration objects that are provided and maintained by Oracle.

There is one base integration object for each business object that contains all the fields that are exposed in that business object. These base integration objects are the primary integration tool, and the required subset of fields demanded by the business requirements are specified by the use of XSD (XML Schema Definition) files. At run time, these XSD files are correlated back to the base integration object and a dynamic integration object is created that the underlying Siebel infrastructure can use transparently. These dynamic integration object definitions are then cached in order to avoid any performance overhead.

Alternatively, you can continue to use your existing integration object definitions by explicitly selecting to Query Integration Objects in the SAI for Oracle Fusion Middleware deployment wizard for inbound Web services in Oracle JDeveloper, and then entering a search term. For information about using this feature, see "Creating Inbound Web Services Based on Siebel Business Services" on page 65.

About Binding Options and Settings for Controlling the Structure of SOAP Messages

This topic describes the binding options and settings for controlling the structure of SOAP messages that are transmitted to and from a Siebel Web service.

SOAP Message Formats Supported in SAI for Oracle Fusion Middleware

SAI for Oracle Fusion Middleware supports both SOAP 1.1 and SOAP 1.2. SOAP 1.1 is based on XML 1.0 and can only use HTTP POST headers to transmit SOAP messages. Therefore, SOAP 1.1 is often not suitable for wide-scale applications. SOAP 1.2 is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. SOAP 1.2 uses XML technologies to define an extensible messaging framework providing a message construct that you exchange over a variety of underlying protocols. The framework is designed to be independent of any particular programming model and other implementation-specific semantics.

SOAP Message Binding Options Supported in SAI for Oracle Fusion Middleware

A Web Services Description Language (WSDL) binding style can be either RPC or document. A WSDL describes a Web service by defining its interface, the operations on that interface, their binding to the transport protocol, and the network address where they are available. A WSDL binding describes how a Web service is bound to a messaging protocol, that is, what is the format or structure of the messages being exchanged with a service, and the transport protocol over which this message is sent. The most popular messaging protocol in Web services today is the SOAP messaging protocol. (The other option is to use REST-based messages.) The binding options supported in SAI for Oracle Fusion Middleware include Document/Literal and RPC/Literal.

About Using the JMS Connector with SAI for Oracle Fusion Middleware

The Java Message Server Transport (JMS) connector is a mechanism you can use to send outbound Web service calls using the new features available in SAI for Oracle Fusion Middleware. For more information about JMS, see *Transports and Interfaces: Siebel Enterprise Application Integration* on *Siebel Bookshelf* and *Oracle® Fusion Middleware Configuring and Managing JMS for Oracle WebLogic Server 11g Release 1 (10.3.5)* in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

You can also use Remote Method Invocation (RMI) to send outbound Web service calls, which is available in the framework provided by SAI for Oracle Fusion Middleware. RMI is the standard for distributed object computing in Java. RMI enables an application to obtain a reference to an object that exists elsewhere in the network, and then invoke methods on that object as though it existed locally in the client's virtual machine. RMI specifies how distributed Java applications must operate over multiple Java virtual machines. For more information about RMI, see *Oracle® Fusion Middleware Programming RMI for Oracle WebLogic Server 11g Release 1 (10.3.5)* in the Oracle Fusion Middleware 11*g* Release 1 Documentation Library.

Process of Deploying Inbound Web Services Using SAI for Oracle Fusion Middleware

You use Oracle JDeveloper to deploy inbound Web services using SAI for Oracle Fusion Middleware. You can create inbound Web services based on either a Siebel business service or a Siebel workflow process, and you can deploy inbound Web services using either Siebel Web services or Enterprise Java Beans.

To deploy an inbound Web service using SAI for Oracle Fusion Middleware, perform the following tasks:

1. Create an inbound Web service by performing one of the following:
   - "Creating Inbound Web Services Based on Siebel Business Services" on page 65
   - "Creating Inbound Web Services Based on Siebel Workflow Processes" on page 69
2. "Deploying Siebel Inbound Web Services Using SAI for Oracle Fusion Middleware" on page 72
3. (Optional) "Deploying Enterprise Java Beans Using SAI for Oracle Fusion Middleware" on page 74
4. "Accessing WSDLs for Deployed Inbound Web Services in the Oracle WebLogic Server Administration Console" on page 77

Creating Inbound Web Services Based on Siebel Business Services

This topic describes how to create inbound Web services based on a Siebel business service. For general information about working with Web services using SAI for Oracle Fusion Middleware, see "About Working with Web Services Using SAI for Oracle Fusion Middleware" on page 62.

This task is a step in "Process of Deploying Inbound Web Services Using SAI for Oracle Fusion Middleware" on page 64.

To create an inbound Web service based on a Siebel business service

1. Start Oracle JDeveloper.
2. Make sure you have specified the database parameters specific to your environment in the General tab in the SAI for Oracle Fusion Middleware Preferences.

   For information about specifying the database parameters, see "Setting Preferences for SAI for Oracle Fusion Middleware" on page 29.

   NOTE: By setting preferences, you have access to Siebel business services when using the Design Wizard.
3. From the application-level menu, select File, and then click the New menu item.
4. In the New Gallery dialog box, perform the following:
   a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.
   b. Select the Design item, and then click OK.
5. At the Welcome to the Design Wizard screen, click Next.
6. Select Siebel Business Service for the artifact type, and then click Next.
7. Specify the binding option and the settings that control the structure of the SOAP message that is transmitted to and from the Siebel Web service, and then click Next.
   a. Select either SOAP 1.1 or 1.2 binding.
   b. Select the SOAP message format:
      - ❑ **Document/Literal.** If you choose this option, then the following annotation is added to your Web service:

        ```
         @SOAPBinding(style=SOAPBinding.Style.DOCUMENT, use=SOAPBinding.Use.LITERAL,
        parameterStyle=SOAPBinding.ParameterStyle.WRAPPED)
        ```

- **RPC/Literal.** If you choose this option, then the following annotation is added to your Web service:

  ```
    @SOAPBinding(style=SOAPBinding.Style.RPC, use=SOAPBinding.Use.LITERAL,
  parameterStyle=SOAPBinding.ParameterStyle.WRAPPED)
  ```

A list of available Siebel business services appears. Only those business services indicated as External user in Siebel Tools appears.

For more information about SOAP message formats using SAI for Oracle Fusion Middleware, see "About Binding Options and Settings for Controlling the Structure of SOAP Messages" on page 63.

8 Select the business service you want to use, and then click Next.

TIP: Alternatively, you can query for a specific business service by typing in the name of the business service, and then clicking Query. You can use wildcards to find the business service that you want using standard Siebel CRM wildcards, for example, %Account%.

The details for the business service and its associated business service methods and arguments appear.

9 Select the business service methods for which you want to generate code, and then click Next. You must choose at least one business service method to proceed.

CAUTION: Make sure not to choose a Siebel business service method that has arguments of the data type Strongly Type Integration Obj, because that data type is not supported.

10 Select how this Web service will use XSDs, and then click Next.

NOTE: This step applies only if you have chosen a business service that has at least one method with argument type Integration Object. Otherwise, skip to Step 14 on page 68.

- **Create New XSD**. Select this option to create a new XSD file by selecting a base integration object.

  NOTE: To create new XSDs, you must have chosen a business service that has at least one argument with the data type of Integration Object defined.

- **Use Existing XSDs**. Select this option to choose an existing XSD file for code generation.

- **Do Not Use XSD**. Select this option to use Siebel integration object definitions configured in the Siebel Repository to generate code.

  NOTE: You can use the same integration object at runtime for the inbound Web service execution.

The resultant dialog box depends on the XSD type you choose.

11 Do one of the following for code generation:

- If you are creating a new XSD, then proceed to Step 12.
- If you are using an existing XSD, then proceed to Step 13.
- If you are not using an XSD, then proceed to Step 14.

12 If you are creating a new XSD to support Web service deployments, then select a base integration object to design an XSD using that integration object by doing the following:

NOTE: If no methods for the chosen business service have the Integration Object data type defined, then the Design XSD wizard page does not appear.

a Click Query Integration Objects, and in the Enter Query dialog box, query for potential integration objects (using the percent sign (%) wildcard), and then click OK.

TIP: Alternatively, you can select an integration object from the prepopulated Base Integration Object drop-down list without performing a query.

The Base Integration Object field populates with all the integration objects for that query. For example, if you enter %Account% for your query, then all the integration objects that have Account as part of their name become available for selection.

b (Optional) If applicable, select a different method name and method argument, otherwise accept the defaults.

c From the Base Integration Object drop-down list, select the integration object you want to use, and then click Load.

The load operation queries the integration object selected and populates the Available XSD tree with the integration object and its user properties as well as the integration component, its fields, user properties, and user keys that are configured for that integration object. However, only those integration components indicated as Active in Siebel Tools appears.

NOTE: Loading the integration object definition might take some time.

d Use the arrows to move the integration object and its user properties and the integration component fields, user properties, and user keys that are configured for that integration object from the Available XSD tree to the Selected XSD tree.

CAUTION: A requirement of XSD generation is that at least one user key is defined for an integration component. If this requirement is not met, then an error occurs when you generate the XSDs. To define user keys in the Selected XSD tree, see Step e.

e (Optional) If you need to create new or update user properties or create new integration component keys or integration component key fields, then do the following:

- **New User Property.** Right-click the integration component or integration component field for which you want to create new user properties, enter the details, and then click OK to return to the Design XSD wizard page.

- **Update User Property.** Right-click the integration component user property you want to update, replace the user property value, and then click OK to return to the Design XSD wizard page.

- **New Integration Component Key**. Right-click the integration components or integration component fields for which you want to create new integration component keys, enter the details, and then click OK to return to the Design XSD wizard page.

- **New Integration Component Key Field.** Right-click the integration component key for which you want a new integration component key field, select the integration component key field you want to use from the autopopulated list, and then click OK to return to the Design XSD wizard page.

  The integration component user key is updated with the new field. If the field already exists in the user key, then no changes occur.

f If there are multiple methods with arguments of type Integration Object, then the Use same XSD for all the methods option is enabled. You can choose to use this option or not.

NOTE: If the Use same XSD for all methods option is not enabled, then you must repeat Step c through Step e on page 67 for all the methods listed in the Method Name drop-down list.

g Click Next to continue.

A message appears upon successful completion stating that the XSD file was successfully generated. The file is saved in the XSD file path that you defined in the General tab in the SAI for Oracle Fusion Middleware Preferences.

NOTE: If an error occurs generating the XSD file, then check the SiebelToolkitExtension.log file for errors and troubleshooting information.

h Click OK to exit the message and continue to the next wizard page.

i Review the XSD files (and the path) for which code will be generated, and then click Next.

j Proceed to Step 14 to review the code generation summary.

13 If you are using an existing XSD for code generation, then do the following, otherwise skip this step:

a Browse to choose an XSD file, and then click Open to return to the Choose an Existing XSD wizard page.

b If there are multiple methods with arguments of type Integration Object, then the Use same XSD for all the methods option is enabled. Select this option if you want to use the first XSD built for all methods. Do not select this option if you want to provide different XSDs for each method.

NOTE: If you do not choose the Use the same XSD for all methods option, then you must repeat Step c through Step e in Step 12 on page 67 for all the methods listed in the Method Name drop-down list.

c Click Next to continue.

14 Review the code generation summary, and then click Finish to start generation.

If there is an existing archive folder structure, then a warning appears asking if you want to overwrite it. One of the following happens:

- If you click Yes, then the folder structure is overwritten.
- If you click No, then a dialog box appears in which you can save the new folder structure in a directory of your choice. By default, the path is set to the XSD file path you defined in the General tab in the SAI for Oracle Fusion Middleware Preferences.

Creating Inbound Web Services Based on Siebel Workflow Processes

This topic describes how to create inbound Web services based on a Siebel workflow process. For more information about working with Web services using SAI for Oracle Fusion Middleware, see "About Working with Web Services Using SAI for Oracle Fusion Middleware" on page 62.

This task is a step in "Process of Deploying Inbound Web Services Using SAI for Oracle Fusion Middleware" on page 64.

To create an inbound Web service based on a Siebel workflow process

1 Start Oracle JDeveloper.

2 Make sure you have specified the database parameters specific to your environment in the General tab in the SAI for Oracle Fusion Middleware Preferences.

For information about specifying database parameters, see "Setting Preferences for SAI for Oracle Fusion Middleware" on page 29.

NOTE: By setting preferences, you have access to Siebel workflow processes when using the Design Wizard.

3 From the application-level menu, select File, and then click the New menu item.

4 In the New Gallery dialog box, perform the following:

a In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.

b Select the Design item, and then click OK.

5 At the Welcome to the Design Wizard screen, click Next.

6 Select Siebel Workflow Process for the artifact type, and then click Next.

A list of available Siebel workflow processes, with a status of complete and that are active, appears.

7 Specify the binding option and the settings that control the structure of the SOAP message that is transmitted to and from the Siebel Web service, and then click Next.

a Select either SOAP 1.1 or 1.2 binding.

b Select the SOAP message format:

❑ **Document/Literal.** If you choose this option, then the following annotation is added to your Web service:

```
 @SOAPBinding(style=SOAPBinding.Style.DOCUMENT, use=SOAPBinding.Use.LITERAL,
parameterStyle=SOAPBinding.ParameterStyle.WRAPPED)
```

- **RPC/Literal.** If you choose this option, then the following annotation is added to your Web service:

  ```
    @SOAPBinding(style=SOAPBinding.Style.RPC, use=SOAPBinding.Use.LITERAL,
  parameterStyle=SOAPBinding.ParameterStyle.WRAPPED)
  ```

For more information about SOAP message formats using SAI for Oracle Fusion Middleware, see "About Binding Options and Settings for Controlling the Structure of SOAP Messages" on page 63.

8 Select the Siebel workflow process you want to use, and then click Next.

TIP: Alternatively, you can query for a specific workflow process by typing in the name of the workflow process, and then clicking Query. You can use wildcards to find the workflow process that you want using standard Siebel CRM wildcards, for example, %Account%.

The details for the workflow process and its associated workflow process properties appear.

9 Review the Siebel workflow process properties, and then click Next to continue.

10 Select how this Web service will use XSDs, and then click Next.

- **Create New XSD**. Select this option to create a new XSD file by selecting a base integration object.

  NOTE: To create new XSDs, you must choose a workflow process that has at least one argument with the data type of Integration Object defined.

- **Use Existing XSD**. Select this option to choose an existing XSD file for code generation.
- **Do Not Use XSD**. Select this option to use Siebel integration object definitions configured in the Siebel Repository to generate code.

  NOTE: You can use the same integration object at runtime for the inbound Web service execution.

The resultant dialog box depends on the XSD type you choose.

11 Do one of the following for code generation:

- If you are creating a new XSD, then proceed to Step 12.
- If you are using an existing XSD, then proceed to Step 13 on page 72.
- If you are not using an XSD, then proceed to Step 14 on page 72.

12 If you are creating a new XSD to support Web service deployments, then select a base integration object to design an XSD using the integration object by doing the following:

a Click Query Integration Objects, and in the Enter Query dialog box, query for potential integration objects (using the percent sign (%) wildcard), and then click OK.

TIP: Alternatively, you can select an integration object from the prepopulated Base Integration Object drop-down list without performing a query.

The Base Integration Object field populates with all the integration objects for that query. For example, if you enter %Account% for your query, then all the integration objects that have Account as part of their name become available for selection.

b (Optional) If applicable, then select a different method name and method argument, otherwise accept the defaults.

c From the Base Integration Object drop-down list, select the integration object you want to use, and then Load.

The load operation queries the integration object selected and populates the Available XSD tree with the integration object and its user properties as well as the integration component, its fields, user properties, and user keys that are configured for that integration object. However, only those integration components indicated as Active in Siebel Tools appears.

NOTE: Loading the integration object definition might take some time.

d Use the arrows to move the integration object and its user properties and the integration component fields, user properties, and user keys that are configured for that integration object from the Available XSD tree to the Selected XSD tree.

CAUTION: A requirement of XSD generation is that at least one user key is defined for an integration component. If this requirement is not met, then an error occurs when you generate the XSDs. To define user keys in the Selected XSD tree, see Step e.

e (Optional) If you need to create new or update user properties or create new integration component keys or integration component key fields, then do the following:

- **Update User Property.** Right click the integration component user property you want to update, replace the user property value, and then click OK to return to the Design XSD wizard page.
- **New Integration Component Key**. Right-click the integration components or integration component fields for which you want to create new integration component keys, enter the details, and then click OK to return to the Design XSD wizard page.
- **New Integration Component Key Field.** Right click the integration component key for which you want a new integration component key field, select the integration component key field you want to use from the autopopulated list, and then click OK to return to the Design XSD wizard page.

  The integration component user key is updated with the new field. If the field already exists in the user key, then no changes occur.

f If there are multiple methods with arguments of type Integration Object, then the Use same XSD for all the methods option is enabled. You can choose to use this option or not.

NOTE: If you do not keep the Use same XSD for all methods option enabled, then you must repeat Step c through Step e on page 71 for all the methods listed in the Method Name drop-down list.

g Click Next to continue.

A message appears upon successful completion stating that the XSD file was successfully generated. The file is saved in the XSD file path that you defined in the General tab in the SAI for Oracle Fusion Middleware Preferences.

NOTE: If an error occurs generating the XSD file, then check the SiebelToolkitExtension.log file for errors and troubleshooting information.

h Click OK exit the message and continue on to the next wizard page.

i Review the XSD files (and the path) for which code will be generated, and then click Next.

   j. Proceed to Step 14 to review the code generation summary.

13. If you are using an existing XSD for code generation, then do the following, otherwise skip this step:

   a. Browse to choose an XSD file, then click Open to return to the Choose an Existing XSD wizard page.

   b. If there are multiple methods with arguments of type Integration Object, then the Use same XSD for all the methods option is enabled. Select this option if you want to use the first XSD built for all methods. Do not select this option if you want to provide different XSDs for each method.

      NOTE: If you do not choose the Use the same XSD for all methods option, then you must repeat Step c through Step e in Step 12 on page 70 for all the methods listed in the Method Name drop-down list.

   c. Click Next to continue.

14. Review the code generation summary, and then click Finish to start generation.

    If there is an existing archive folder structure, then a warning appears asking if you want to overwrite it. One of the following happens:

    - If you click Yes, then the folder structure is overwritten.
    - If you click No, then a dialog box appears in which you can save the new folder structure in a directory of your choice. By default, the path is set to the XSD file path you defined in the General tab in the SAI for Oracle Fusion Middleware Preferences.

Deploying Siebel Inbound Web Services Using SAI for Oracle Fusion Middleware

This topic describes how to deploy Siebel inbound Web services using SAI for Oracle Fusion Middleware.

This task is a step in “Process of Deploying Inbound Web Services Using SAI for Oracle Fusion Middleware” on page 64.

NOTE: Before you perform the following procedure, make sure you have deployed an appropriate Siebel Resource Adapter. For information about deploying Siebel Resource Adapters, see “Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware” on page 46.

To deploy a Siebel inbound Web service using SAI for Oracle Fusion Middleware

1. Start Oracle JDeveloper.
2. From the application-level menu, select File, and then click the New menu item.
3. In the New Gallery dialog box, perform the following:

   a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.

   b. Select the Deployment item, and then click OK.

4 In the Welcome to the Deployment Wizard screen, click Next.

5 Select Siebel Inbound Services as the deployment type, and then click Next.

6 Select one or both of the following options for the inbound deployment type you want to deploy, and then click Next.

- **Inbound Web Service Deployment.** This option allows you to deploy the code generated from the Siebel business services or Siebel workflow processes you design as SOAP Web services to Oracle WebLogic Server by way of a WAR file.
- **EJB Deployment.** This option allows you to deploy a Siebel business service or a Siebel workflow process as an inbound Enterprise Java Bean.

7 Select the Siebel inbound services you want to deploy by moving the services from the Available section to the Selected section, and then click Next.

NOTE: The path for the generated code is the XSD File Path you specified in the General tab in the SAI for Oracle Fusion Middleware Preferences.

8 Enter the inbound deployment parameters by doing the following:

a Enter the Web service version number that you want to use for the deployment. By default, this parameter is set to 1.0.0.0.0.

b Enter the appropriate Siebel Resource Adapter name that you want to use for this deployment. By default, the value from the last Siebel Resource Adapter deployed on Oracle WebLogic Server appears.

c Enter the Siebel Resource Adapter Connection Factory JNDI Name that you want to use for this deployment. By default, the value from the last Siebel Resource Adapter deployed on Oracle WebLogic Server appears.

d If you are deploying Enterprise Java Beans, then follow the instructions in Step 9 on page 76 in the "Deploying Enterprise Java Beans Using SAI for Oracle Fusion Middleware" procedure.

e Click Next to continue.

9 Enter the Oracle WebLogic Server parameters for the service you are deploying, and then click Next. For a description of the parameters, see *Siebel Application Integration for Oracle Fusion Middleware: Oracle JDeveloper Online Help*.

10 Review the summary information, and then click Finish to start the deployment.

NOTE: If there is an existing archive folder structure with the same name, then a warning appears asking if you want to overwrite it. If you click Yes, then the existing folder structure is overwritten. If you click No, then a dialog box appears in which you can name a new folder structure and save that structure in a directory of your choice.

Information about successful and failed deployments appears. Click OK to exit.

NOTE: If the deployment fails, then check the SiebelDeployExtension.log file. By default, this file is found in `C:\JDeveloper\mywork\SFE`.

The deployed service is now available in the Oracle WebLogic Server Administration Console. From there, you can access the WSDL and import it into a client application to send and receive messages. For information about logging in to Administration Console, see "Logging In to the Oracle WebLogic Server Administration Console" on page 39. For information about accessing and using the WSDL, see "Accessing WSDLs for Deployed Inbound Web Services in the Oracle WebLogic Server Administration Console" on page 77.

Deploying Enterprise Java Beans Using SAI for Oracle Fusion Middleware

This topic describes how to deploy Enterprise Java Beans using SAI for Oracle Fusion Middleware.

This task is a step in "Process of Deploying Inbound Web Services Using SAI for Oracle Fusion Middleware" on page 64.

To deploy an Enterprise Java Bean using SAI for Oracle Fusion Middleware

1. Start Oracle JDeveloper.
2. From the application-level menu, select File, and then click the New menu item.
3. In the New Gallery dialog box, perform the following:
    a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.
    b. Select the Deployment item, and then click OK.
4. In the Welcome to the Deployment Wizard screen, click Next.
5. Select the Siebel Inbound Services deployment type, and then click Next.
6. Select EJB Deployment, and then click Next.

    NOTE: You can select Inbound Web Service or EJB Deployment or both.

7. Select one or more Siebel inbound services that you want to deploy by moving the services from the Available section to the Selected section, and then click Next.

    NOTE: The path for the generated code is the XSD File Path you specified in the General tab in the SAI for Oracle Fusion Middleware Preferences.

8 Enter the inbound deployment parameters for the Enterprise Java Bean.

Some parameters are described in the following table.

Inbound Deployment Parameters	Description
Version	This field controls the version of the inbound Web service that is being deployed on Oracle WebLogic Server. For example, if you deployed a Siebel account with version 1.0.0.0, and you want to deploy another Siebel account, then use this field to increment the version (such as 1.0.0.1) to make sure there is no conflict. **NOTE:** Typically, the Oracle WebLogic Server Administration Console shows older versions as retired.
Siebel Resource Adapter Name	Specifies the name for the resource adapter. You can change this value to point to a different resource adapter. By default, the name of the resource adapter that was last deployed appears.
Siebel Resource Adapter JNDI Name	Specifies the JNDI name for the connection factory the resource adapter uses. You can change this value to point to a different resource adapter. By default, the name of the resource adapter that was last deployed appears.

9 Enter the Enterprise Java Bean parameters by doing the following:

a Click Edit, and then click the plus sign (+) to enter the parameters.

The parameters are described in the following table.

EJB Parameter	Description
Business Service Name	The Siebel business services and Siebel workflow processes that have been selected for deployment. **TIP:** To select a business service, click in the Business Service Name field, and then choose from the drop-down list.
JNDI Name	Enter the Java Naming and Directory Interface name for the Java Enterprise Bean. By default, this parameter is not specified.
Transaction Timeout	The transaction timeout value, in seconds. By default, this parameter is set to 600.
Initial Pool Size	The initial number of beans in the free pool. By default, this parameter is set to 5.
Maximum Pool Size	The maximum number of beans in the free pool. When an Enterprise Java Bean has been in the free pool for the number of seconds specified in Idle Timeout, and the total number of beans in the free pool approaches the maximum beans in the free pool specified in this field, then the idle beans are removed from the free pool. By default, this parameter is set to 20.

b Click Save (or Cancel) to exit the EJB Parameters dialog box, and then click Next again.

NOTE: Click X if you want to remove your selection without existing the EJB Parameters dialog box.

10 Enter the Oracle WebLogic Server parameters for the service you are deploying, and then click Next.

For a description of the parameters, see *Siebel Application Integration for Oracle Fusion Middleware: Oracle JDeveloper Online Help*.

11 Review the summary information, and then click Finish to deploy.

NOTE: If there is an existing archive folder structure with the same name, then a warning appears asking if you want to overwrite it. If you click Yes, then the existing folder structure is overwritten. If you click No, then a dialog box appears in which you can name a new folder structure and save that structure in a directory of your choice.

Information about successful and failed deployments appears. Click OK to exit.

NOTE: If the deployment fails, then check the SiebelDeployExtension.log file. By default, this file is found in `C:\JDeveloper\mywork\SFE`.

Accessing WSDLs for Deployed Inbound Web Services in the Oracle WebLogic Server Administration Console

This topic describes how to access WSDLs for deployed inbound Web services in the Oracle WebLogic Server Administration Console.

This task is one step in "Process of Deploying Inbound Web Services Using SAI for Oracle Fusion Middleware" on page 64.

To access a WSDL for a deployed inbound Web service in the Oracle WebLogic Server Administration Console

1. Log in to the Oracle WebLogic Server Administration Console.
2. From the domain structure, click Deployments, and then select an inbound Web service that was deployed using the SAI for Oracle Fusion Middleware Deployment wizard.
3. Drill down on the Web service.
4. Select the Web service for the deployed Web application.
5. Click the Testing tab, and then expand the Web service.
6. In the Test Point column, click WSDL.

   The WSDL document opens in a browser.

Roadmap for Deploying Outbound Web Services Using SAI for Oracle Fusion Middleware

You use the Oracle WebLogic Server Administration Console (Administration Console) to create new outbound Web services using SAI for Oracle Fusion Middleware, and then you use Oracle JDeveloper to deploy these newly created outbound Web services.

To deploy an outbound Web service using SAI for Oracle Fusion Middleware, perform the following tasks:

1. Configure the Siebel Server, Oracle WebLogic Server, and Oracle JDeveloper for use in deploying outbound Web services.

   For information, see "Process of Configuring SAI for Oracle Fusion Middleware for Outbound Web Services" on page 33.
2. "Referencing Oracle WebLogic Server Outbound Web Services in Siebel Business Applications" on page 78
3. "Deploying Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 79
4. "Configuring JMS Resource Adapters for Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 80
5. "Testing Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 82

Referencing Oracle WebLogic Server Outbound Web Services in Siebel Business Applications

This topic describes how to reference Oracle WebLogic Server outbound Web services in Siebel Business Applications. For general information about working with Web services using SAI for Oracle Fusion Middleware, see "About Working with Web Services Using SAI for Oracle Fusion Middleware" on page 62.

This task is a step in "Roadmap for Deploying Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 77.

To reference an Oracle WebLogic Server Web service in Siebel Business Applications

1. Start Oracle JDeveloper.
2. From the application-level menu, select File, and then click the New menu item.
3. In the New Gallery dialog box, perform the following:
   a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.
   b. Select the Deployment item, and then click OK.
4. At the Welcome to the Deployment Wizard screen, click Next.
5. Select the Siebel Outbound Services deployment type, and then click Next.
6. Specify the following outbound Web service parameters, and then click Next.

   The parameters are described in the following table.

Setting	Description
WSDL Document URL	The URL of the Web service description that you want to use. The URL can be either a file that you locate using the Browse button or an HTTP URL.
Outbound Web Service Name	The outbound Web service name on Oracle WebLogic Server. By default, this value is set to the service name that appears in the selected WSDL.
EAR Version	The outbound Web service version.
Outbound Web Service JNDI Name	Identifies a specific outbound Web service client deployed on Oracle WebLogic Server for each request by performing a lookup. By default, this value is set to: eis/siebel/outbound/*outbound_Web_service_name*
Transaction Timeout	The transaction timeout value in seconds. By default, the value is set to 600 seconds.

7. The Web service name and JNDI Name fields auto-populate based on the WSDL name, but are editable.

   NOTE: This is an outbound call from the Siebel application, therefore, if after a specified time there is no response, the Siebel application receives a timed out message.

8. Click OK to check the values.

Deploying Outbound Web Services Using SAI for Oracle Fusion Middleware

This topic describes how to deploy outbound Web services using SAI for Oracle Fusion Middleware.

This task is a step in "Roadmap for Deploying Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 77.

To deploy an outbound Web service using SAI for Oracle Fusion Middleware

1. Start Oracle JDeveloper.
2. From the application-level menu, select File, and then click the New menu item.
3. In the New Gallery dialog box, perform the following:
   a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.
   b. Select the Deployment item, and then click OK.
4. In the Welcome to the Deployment Wizard screen, click Next.
5. Select Siebel Outbound Services for the type of deployment, and then click Next.
6. Enter or browse for the WSDL (Web Services Description Language) file or enter the URI (Uniform Resource Identifier) for the WSDL, and then click Next.

   NOTE: The other fields in this screen automatically populate.

7. Enter the Oracle WebLogic Server parameters, and then click Next.

   For information about the Oracle WebLogic Server parameters, see "Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware" on page 46.

8. Verify the summary and read the applicable notes, and then click Finish to deploy the outbound Web service.
9. (Optional) If code was previously created for this service, then a dialog box appears asking you to confirm overwriting the existing code. Click Yes to continue, if appropriate.

Configuring JMS Resource Adapters for Outbound Web Services Using SAI for Oracle Fusion Middleware

This topic describes how to configure JMS resource adapters for outbound Web services using SAI for Oracle Fusion Middleware.

This task is a step in "Roadmap for Deploying Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 77.

TIP: The following procedure is similar to the resource adapter configuration described in "Configuring Siebel Server Load Balancing and SSL for SAI for Oracle Fusion Middleware" on page 55, except that you define outbound configurations for the adapter rather than inbound.

To configure a JMS resource adapter for outbound Web services using SAI for Oracle Fusion Middleware

1 Make sure the Oracle WebLogic Server is up and running.

2 Start Oracle JDeveloper, if not already active.

3 Select File, and then click the New menu item.

4 In the New Gallery dialog box, perform the following:

   a In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.

   b Select the Deployment item, and then click OK.

5 At the Welcome to the Deployment Wizard screen, click Next.

6 Select Siebel Resource Adapter as the deployment type, and then click Next.

7 Enter the resource adapter parameters by doing the following:

   a Enter a resource adapter name, if you do not want to accept the default. By default, the resource adapter name is SiebelResourceAdapter.

   NOTE: Each resource adapter has its own log file. The log file name is the same name as the resource adapter name appended with .log (that is, *JNDI_ra_name*.log). The location of the log file depends on the value you specify in the Deployment Log Directory logging parameter (see description in the following Step b on page 81).

b Click Edit ra.xml, click the Outbound RA Configuration tab, select the Enable JMS Configuration option, and then click Next.

Some parameters are described in the following table.

Parameter	Description
Queue Connection Factory	The JNDI name for the connection factory queue created on the Oracle WebLogic Server Administration Console.
Receive Queue	Specify the JNDI name for the configured queue on Oracle WebLogic Server. **NOTE:** You use the ReceiveQueue and SendQueue parameters to pass the JNDI name to Oracle WebLogic Server. The JNDI name for ReceiveQueue is the same as SendQueue in the Siebel Server named subsystem.
Send Queue	Specify the JNDI name for the configured queue on Oracle WebLogic Server. **NOTE:** You use the SendQueue and ReceiveQueue parameters to pass the JNDI name to Oracle WebLogic Server. The JNDI name for SendQueue is the same as ReceiveQueue in the Siebel Server named subsystem.
MDB Parameters	
Initial Free Pool Size	The initial number of beans in the free pool. By default, this value is set to 5.
Maximum Free Pool Size	The maximum number of beans in the free pool. When an Enterprise Java Bean has been in the free pool for the number of seconds specified in Idle Timeout, and the total number of beans in the free pool approaches the maximum beans in the free pool specified in this field, then the idle beans are removed from the free pool. By default, this value is set to 10.

For descriptions of the other XML parameters, see Step 7 on page 46 in “Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware” on page 46.

8 Follow the prompts until you complete the resource adapter deployment.

For more information about deploying resource adapters, see “Configuring Siebel Server Load Balancing and SSL for SAI for Oracle Fusion Middleware” on page 55.

9 Click Finish.

Two new entries are created for Oracle WebLogic Server.

Testing Outbound Web Services Using SAI for Oracle Fusion Middleware

This task is a step in "Roadmap for Deploying Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 77.

To test outbound Web services using SAI for Oracle Fusion Middleware

1. Navigate to the Administration - Business Service screen, then the Simulator view.
2. Create a new business service with the following values:
   - Service Name is EAI Outbound Proxy Service.
   - Method Name is CallExternalWebSvc.
3. Import an XML file by doing the following:
   a. In the Inputs Arguments list, create a new record.
   b. In the Property Name field in the new record, create the following new properties, and then click OK.

Property	Value
RemoteJNDIName	JNDI name of the outbound WSDL deployed
WebServiceName	Service Name
WebMethodName	Method name
EndPointName	Ports for a service name
Input	Value for input

The following is sample input created for the GlobalWeather Web service:

```
<?xml version="1.0" encoding="UTF-8"?><?Siebel-Property-Set
EscapeNames="true"?><GetWeather
CountryName="INDIA"
CityName="BOMBAY"
RemoteJNDIName="eis.siebel.outbound.globalweather"
WebServiceName="GlobalWeather"
WebMethodName="GetWeather"
EndPointName="GlobalWeatherSoap"></GetWeather
>
```

4. Click Run.
5. The XML response appears in Output Arguments.

7 Working with Siebel RESTful Services

This chapter describes Siebel Representational State Transfer (RESTful) Services and how to use them. It includes the following topics:

Overview of Siebel RESTful Services

Siebel RESTful Services are a set of Java Platform, Enterprise Edition (Java EE) services that make available a Siebel Repository instance or a Siebel enterprise instance to a connected application or system. It provides a Web service architecture that defines how you use Web standards, such as HTTP and URIs (Uniform Resource Identifier). Many Siebel Business Applications use SOAP (Simple Object Access Protocol) Web services for integration and custom application development that require a wide range of Siebel business objects and business services. Siebel RESTful Services provide an alternative to SOAP where custom development and the user experience requires maximum scalability and stateless operation. While SOAP Web services provide a substantial set of architectural and application server demands, Siebel RESTful Services require no more complexity than what is required to deliver Web pages over HTTP.

REST (Representational State Transfer) is not a standard. It is a collection of resources that includes the following components:

- The URI for a Web service, for example:

  http://example.com/resources/cars

- The Multipurpose Internet Mail Extensions (MIME) type of the data that a Web service supports. Although this type is typically JavaScript Object Notation (JSON), eXtensible Markup Language (XML), or Yet Another Multicolumn Layout (YAML), it can be any valid MIME type.
- The set of operations that a Web service that uses HTTP methods supports, such as GET, PUT, POST, AND DELETE.

You can use Siebel RESTful Services to do the following:

- Access Siebel Repository metadata even if you are not familiar with other Siebel development environments, such as Siebel Tools or Siebel administrative interfaces.
- Generate applications that interact with Siebel CRM user data.

Benefits of Using Siebel RESTful Services

One of the benefits of using RESTful services is that a session is not necessary so the overhead is lower and the performance may be better. When an application does not need to maintain a state between steps such as a shopping cart application would, and the interaction is supposed to be a small and quick one such as updating, creating or deleting a single record, RESTful services make sense. For more complicated interactions with the Siebel application web services are the preferred method.

Architecture of Siebel RESTful Services

Figure 4 illustrates the architecture of Siebel RESTful Services.

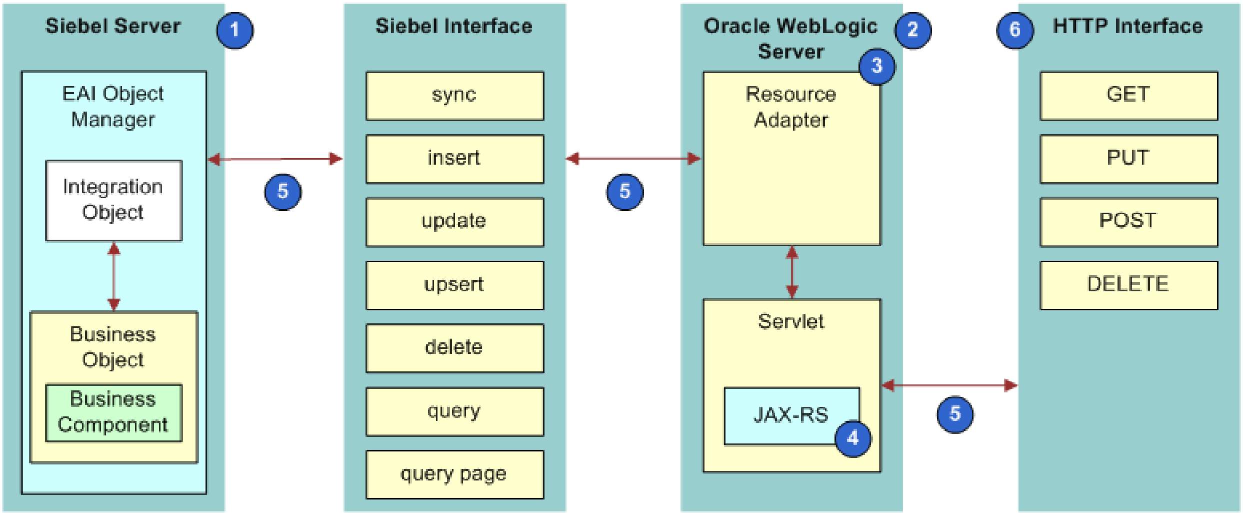

Figure 4. Architecture of Siebel RESTful Services

As shown in Figure 4, Siebel RESTful Services makes available the methods and operations that the EAI Object Manager business service provides through an interface that complies with the principles of REST HTTP services. It includes the following components:

1 **Siebel Server.** Executes one or more instances of an EAI Object Manager. Integration objects abstract required data from the business objects.

2 **Oracle WebLogic Server.** Executes one or more instances of the Siebel RESTful Services. This service is a Java EE Web application and a Java EE resource adapter. They are packaged together in a Java EE enterprise application. The Java EE Web application is a servlet that uses the JAX-RS standards. The Java EE resource adapter wraps the Siebel JDB EAIObjMgrSvc class in a Java EE Connector Architecture (JCA) interface.

3 **Resource Adapter.** To support Siebel RESTful Services, the resource adapter maps standard Siebel EAI operations to HTTP methods. Example REST operations include synchronization, insert, update, and so on. Example HTTP methods include: PUT, GET, DELETE, POST.

4 **JAX-RS Servlet.** This servlet does the following:

- Converts HTTP resource requests to requests that are compatible with Siebel EAI.
- Converts request and response objects between the Siebel EAI format and the Siebel RESTful Services format.

5 **Message objects.** Transferred as part of a Siebel RESTful Services request or response.

6 **HTTP Interface.** Siebel RESTful Services use the HTTP protocol to use the standard GET, PUT, POST, AND DELETE operations.

For more information, see "How Siebel RESTful Services Map HTTP to the EAI Object Manager" on page 85.

How Siebel RESTful Services Map HTTP to the EAI Object Manager

Siebel EAI uses the EAI Siebel Adapter business service and its own internal message structure and methods to insert, update, and delete records in the Siebel database. Siebel RESTful Services act as a translator between Web request structures and language and the EAI Siebel Adapter structures and language. Because of this, you can access the methods of the EAI Siebel Adapter business service from external applications without having to know all the intricacies of the service. For more information about the EAI Siebel Adapter, see *Integration Platform Technologies: Siebel Enterprise Application Integration*.

You can create simple XML and send it to a URI. The syntax and structures used in this chapter show how to create simple XML messages and construct URIs. Each of the standard PUT, GET, DELETE, and POST HTTP methods are constructed differently.

Siebel RESTful Services map semantics for HTTP methods and URIs into semantics for the EAI Object Manager. When a response object is returned, the default format is a Siebel RESTful Services message object. Siebel RESTful Services use the Attachment business components to access the file system. For an example, see "Example of Using Code to Access Data from the Siebel File System" on page 100.

Table 6 describes the mapping between HTTP resource requests and requests that are compatible with Siebel EAI.

Table 6. Mapping Between HTTP Resource Requests and Requests That Are Compatible with Siebel EAI

HTTP Resource Requests	Requests That Are Compatible with Siebel EAI
GET, PUT, POST, and DELETE	The following requests are compatible with Siebel EAI: ■ insert ■ fetch ■ query ■ querypage ■ update ■ upsert ■ delete ■ execute

For examples of how to use some of these methods by making calls to Siebel RESTful Services, see "Sample URIs for Siebel RESTful Services" on page 93.

HTTP URI Pattern for Siebel RESTful Services

Siebel RESTful Services execute resource requests that match the following HTTP URI pattern:

```
hostname:port/Web_context_root/rest/siebel/JNDI_ra_name/IO_name/modifier
```

where:

- *hostname:port* is the DNS name and port number of the server that hosts the Java EE application server. If the server is an instance of an Oracle WebLogic Server, then this host is typically similar to `myhost.mycompany.com:port`.
- *Web_context_root* is defined in the Web application deployment plan. The default value is `oracle-crm/api`.

  NOTE: The context root is a name that gets mapped to the document root of a Web client and is the starting place from which Oracle WebLogic Server attempts to find servlets or other programs that are to be executed.

- *rest* represents the default mapping for the JAX-RS servlet. It is defined in the web.xml deployment descriptor for the Web application.

- *siebel/JNDI_ra_name/IO_name* is interpreted by the JAX-RS resource handlers where:
  - *JNDI_ra_name* identifies the appropriate Java EE Connector Architecture (JCA) resource adapter.

    This adapter contains a JNDI (Java Naming and Directory Interface) name that is equal to eis/*JNDI_ra_name*ConnFactory, where *eis* is *enterprise information systems*. Siebel RESTful Services use it to execute the request.
  - *IO_name* identifies the requested Siebel resource on which to apply the request.
- *modifier* is an additional parameter that can modify the behavior of the request. This parameter can include a URI path parameter or URI query parameter.

  For example, the following URI queries a Siebel repository:

  http://*DNS_name*:*port*/oracle-crm/api/rest/siebel/siebel.*JNDI_ra_name*/PhoenixIO

  where:

  - *DNS_name* is the host.
  - *port* is the port number.
  - *oracle-crm/api* is the Web context root.
  - *rest/siebel* is automatically part of the URI for Siebel RESTful Services.
  - *siebel.JNDI_ra_name* is the resource adapter name.
  - *PhoenixIO* is the integration object name.

  This URI returns the integration objects in the repository as XML.

TIP: To search for particular integration objects, you use a modifier. For example, if you want to find only those integration objects that start with the letter A, then you can use the following URI:

```
http://myserver.mycompany.com:port/oracle-crm/api/rest/siebel/siebel.JNDI_ra_name/
Account_EMR/searchexpr=[IntCompName.IntCompFieldName] LIKE 'A*'
```

About URI Parameters

When constructing a URI, there are a number of parameters available to achieve the desired results. The parameters are optional, but depending on the desired outcome, might be required. Some parameters are described in Table 7.

Table 7. URI Parameters

Parameter	Description
pagesize	Integer that tells the EAI Siebel Adapter business service how many records to return. If you query for all the Siebel contacts whose last name starts with the letter A and you do not want to get too many records (for performance reasons), then you can restrict the number of records returned. You restrict the number of records returned by setting the pagesize parameter to a reasonable number. All records that match the search criteria are returned. For example, pagesize=20 returns only twenty contact records, even if more exist in the Siebel database. If fewer records exist that match that search criteria, then all records are returned (but no more than twenty). **NOTE:** It is recommended that you retrieve the lowest number of records required for any one call. The more records that are returned, the larger the message and the slower the response.
startrow	Used when there is a need to start returning records at a specific row. For example, startrow=100 starts at row 100 of the record set. The first number in a record set is zero, therefore, this request starts at record 99 (given you start counting from one for the first record). This parameter is useful for paging through a record set N records at a time. For example, if there are 100 records in a record set, but you want to retrieve only ten at a time, then enter startrow=0 and pagesize=10 on the first call, then startrow=10 on the next call, then startrow=21 on the next call, and so on.
viewmode	Maps to the different Siebel view modes. This parameter restricts record access similarly to how views are restricted in Siebel Business Applications. These restrictions include: ■ SalesRep ■ Manager ■ Personal ■ All ■ Organization ■ SubOrgainziation ■ Group ■ Catalog

Table 7. URI Parameters

Parameter	Description
searchexpr	Specifies search criteria. The syntax is similar to the syntax used in Siebel Workflow. For example, to search the Account integration component for accounts whose names begin with the letter A, construct a search expression searchexpr=[Account.Name] LIKE 'A*' using the following syntax: [Integration Component Name.Integration Component Field] LIKE 'SomeValue'. You can also search for more than one criterion searchexpr=[Account.Name] LIKE 'A*' AND [Account.Location] = 'Downtown' uses two integration component fields (Name and Location).
sortexpr	Specifies how to sort the records and is equivalent to use with business components. The syntax is sortexpr=Integration Component Field, Another Integration Component Field, and so on. For example, the expression: sortexprt=Last Name (DESCENDING), First Name sorts first by the Last Name with the alphabetically last one first, then by the First Name field in ascending order.
updatemode	Tells the EAI Siebel Adapter the type of operation to execute. The options are: ■ **sync.** This mode supports update and insert. Records that do not match the incoming record are deleted. For example, consider an instance of one parent and three children in the Siebel database. If a sync call is made where the message matches only two of the three records in the Siebel database, then the sync operation deletes the third, unmatched child record from the database. ■ **upsert.** This mode supports update and insert and is similar to sync, however, records are not deleted. ■ **update.** This mode supports update. Records that do not match the incoming record are deleted. ■ **insert.** This mode supports insert. Records that do not match the incoming record are deleted. ■ **mixed.** This mode supports delete, update, and insert. You determine the actual operation from the *operation* element in the message body. For more information about EAI Siebel Adapter operations, see *Integration Platform Technologies: Siebel Enterprise Application Integration*. **TIP:** The updatemode parameter is specific to Siebel RESTful Services.

The syntax for using URI parameters is lower case letters followed by an equal sign (=) with the value of the parameter, and each parameter is separated from other parameters by an ampersand (&). For example, if you want to set the pagesize parameter to 20 and the startrow parameter to 0 (zero), then you enter:

```
Pagesize=20&startrow=0
```

The following is an example of an HTTP GET call that combines most of the parameters described above:

```
http://localhost:port/oracle-crm/api/rest/siebel/JNDI_ra_name/Internal Contact
Interface?sortexpr=First Name(DESCENDING)&pagesize=2&startrow=2&viewmode=All&
searchexpr=[Contact.Last Name] LIKE 'C*'
```

Deploying Siebel RESTful Services Using SAI for Oracle Fusion Middleware

To deploy Siebel RESTful Services, you must first deploy SAI for Oracle Fusion Middleware. You must also deploy the Jersey libraries that are packaged as part of the Oracle WebLogic Server installer. For more information about Siebel RESTful Services, see "Overview of Siebel RESTful Services" on page 83.

To deploy Siebel RESTful Services using SAI for Oracle Fusion Middleware

1. Make sure you have deployed the Jersey libraries so that they are available in the Oracle WebLogic Server Administration Console.

   For information about the Jersey libraries and how to deploy them, see "Manually Installing Jersey Libraries for SAI for Oracle Fusion Middleware" on page 38.

2. Start Oracle JDeveloper, if not already active.

3. Select File, and then click the New menu item.

4. In the New Gallery dialog box, perform the following:

   a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.

   b. Select the Deployment item, and then click OK.

5. In the Welcome to the Deployment Wizard screen, click Next, and then do the following:

   a. Select REST as the deployment type, and then click Next.

   b. Choose one of the following security models, and then click Next.

      - DDOnly

      - CustomRolesAndPolicies

      NOTE: By default, the model is set to DDOnly. If you want to use Oracle WebLogic Server security, you must choose CustomRolesandPolicies, and also perform some additional configuration. For more information, see Chapter 4, "Setting Up Security for SAI for Oracle Fusion Middleware on Oracle WebLogic Server."

   c. Enter the details of the Oracle WebLogic Server parameters on which the Siebel RESTful Services will be deployed, and then click Next.

   d. Click Finish to deploy Siebel RESTful Services.

6. Make sure an appropriate Siebel Resource Adapter has been deployed to the applicable Oracle WebLogic Server and that the connection is established between Oracle WebLogic Server and the Siebel application.

7. Test your deployment by calling Siebel RESTful Services using the following URI syntax:

   ```
   http://hostname:port/oracle-crm/api/rest/siebel/siebel.JNDI_ra_name/
   IO_name?searchexpr=[IntCompName.IntCompFieldName] LIKE 'A*'
   ```

   where:

   - *hostname* is the fully qualified domain name of your Oracle WebLogic Server.
   - *port* is the configured port on Oracle WebLogic Server that processes requests for the Siebel RESTful Services Web application.
   - *JNDI_ra_name* is the JNDI name for the connection factory of the resource adapter that the Siebel RESTful Services use. The JNDI is specified during the resource adapter deployment and is also available through the Oracle WebLogic Server Administration Console.
   - *IO_name* is the integration object name.
   - *IC* is the integration component.
   - *IC_field_name* is the integration component name.

   For example, the following URI:

   ```
   http://localhost:7001/oracle-crm/api/rest/siebel/
   siebel.SiebelResourceAdapterConnFactory/EAI%20Account?searchexpr=[Account.Name]
   LIKE 'A*'
   ```

   Returns the following XML code:

   ```
   <?xml version="1.0" encoding="utf8" ?>

   <SiebelREST>

      <NumOutputObjects>1</NumOutputObjects>

      <LastPage>true</LastPage>

      <ErrorCode>0x0</ErrorCode>

         <ListOfAccount>

            <Account>

               <Location>Charlotte</Location>

               <Name>AB Company</Name>

            </Account>

         </ListOfAccount>

   </SiebelREST>
   ```

   Which, in turn, displays all the account records with names that start with A.

Related Topics

"Manually Installing Jersey Libraries for SAI for Oracle Fusion Middleware" on page 38

Architecture of Siebel RESTful Services Messages

Generally, you can accomplish interaction with Siebel RESTful Services by providing parameters to a URI. This task requires only a browser. More complicated interactions, however, such as querying a record by example, or inserting or updating a new record, need more than just the URI. The more complicated interactions (including any interaction more complicated than queries with row IDs or simple search strings) require you to also post a message body, not just a message header. The message *header* is what you can pass in a browser's address or location bar. For example, the following is a request header:

```
http://localhost:7101/oracle-crm/api/rest/siebel/MyRest/Internal Contact
Interface?updatemode=insert
```

It has no apparent information, because the pertinent details are contained in the message body, (not in the header) as is typical with a query.

A message *body* contains XML that holds all the data for the fields necessary to delete, update, or insert the correct record data. It has the following structure:

```
<XML header>

<SiebelREST>

    <IntegrationComponentWrapperName>

        <IntegrationComponentXMLName>

        <operation></operation>

           <IntegrationComponentFieldName1>DataForField1</
        IntegrationComponentFieldName1>

           …

           <IntegrationComponentFieldNameN>DataForFieldN</
        IntegrationComponentFieldNameN>

        </IntegrationComponentXMLName>

    <IntegrationComponentWrapperName>

</SiebelREST>
```

NOTE: SiebelREST is the root node.

Using this structure, you can insert a new contact into the Internal Contact Interface integration object providing values to produce the following message body for the message header above.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<SiebelREST>
   <ListOfInternal_0x20_Contact_0x20_Interface>
      <Contact>
         <operation>insert</operation>
         <Last_0x20_Name>MyLastName</Last_0x20_Name>
         <First_0x20_Name>MyFirstName</First_0x20_Name>
         <Middle_0x20_Name>MyMiddleName</Middle_0x20_Name>
         …
      </Contact>
   </ListOfInternal_0x20_Contact_0x20_Interface>
</SiebelREST>
```

Use the same structure for updates and deletes, but change the operation element's value and the update mode in the header as necessary.

Sample URIs for Siebel RESTful Services

Table 8 provides sample Uniform Resource Identifiers (URIs) that you can use for working with Siebel RESTful Services.

Table 8. Sample URIs

Task	HTTP Request Type	Sample URI
Determine REST server is up	GET	http://*weblogic_server_IP*:*port*/oracle-crm/api/rest/siebel/heartbeat The response you want is: The REST server is alive.

Table 8. Sample URIs

Task	HTTP Request Type	Sample URI
To invoke query operations	GET	To query by row ID: http://*weblogic_server_IP*:*port*/oracle-crm/api/rest/siebel/*JNDI_ra_name*/*IO_name*/*ID* The response is XML output for all of the fields that are defined and active in the integration component for that particular record. **NOTE:** The query by row ID HTTP request is the same as that used for the delete operation, however, the HTTP request type for the delete operation is DELETE, not GET.

	POST	To query by example:
		http://*weblogic_server_IP*:*port*/oracle-crm/api/rest/siebel/*JNDI_ra_name*/*IO_name*/query
		Sample request XML:
		`<?xml version = '1.0' encoding = 'utf-8'?>`
		`<SiebelREST>`
		`<ListOfAccount_EMR>`
		`<Account>`
		`<Name>A*</Name>`
		`</Account>`
		`</ListOfAccount_EMR>`
		`</SiebelREST>`
		NOTE: All requests with POST HTTP request type are either insert or upsert operations.

To invoke query operations (CONTINUED)	GET	To query by search spec: http://*weblogic_server_IP*:*port*/oracle-crm/api/rest/siebel/*JNDI_ra_name*/*IO_name*?searchexpr=[*IntCompName.IntCompFieldName*] LIKE 'A*'

Table 8. Sample URIs

Task	HTTP Request Type	Sample URI

To invoke delete operation	DELETE	To delete by row ID:
		http://*weblogic_server_IP*:*port*/oracle-crm/api/rest/siebel/*JNDI_ra_name*/*IO_name*/<ID>
		Sample response XML:
		`<?xml version = '1.0' encoding = 'utf-8'?>`
		`<SiebelREST>`
		`<ErrorCode>0x0</ErrorCode>`
		`    <ListOfAccountAttOrderAtt>`
		`        <Account/>`
		`    </ListOfAccountAttOrderAtt>`
		`</SiebelREST>`
		TIP: ErrorCode 0x0 indicates that the deletion operation was successful.
		NOTE: The delete HTTP request is the same as that used for the query by row ID operation, however, the HTTP request type for the query by row ID operation is GET, not DELETE. Additionally, the XML returned by the DELETE operation is similar to that of an update-insert-upsert operation.

Table 8. Sample URIs

Task	HTTP Request Type	Sample URI

To invoke insert operation	POST	http://*weblogic_server_IP*:*port*/oracle-crm/api/rest/siebel/*JNDI_ra_name*/*IO_name*?updatemode=insert
		Sample request XML:
		`<?xml version="1.0" encoding="ISO-8859-1"?>`
		`<SiebelREST>`
		`<ListOfAccountAttOrderAtt>`
		`<Account>`
		`<Name>AttachAbhishek36</Name>`
		`<Location>HeadQuarter</Location>`
		`<Currency_0x20_Code>BRL</Currency_0x20_Code>`
		`<Primary_0x20_Organization>Default Organization</Primary_0x20_Organization>`
		`</Account>`
		`</ListOfAccountAttOrderAtt>`
		`</SiebelREST>`
		After performing this insert operation, the resultant XML specifies the new row ID that was created as a result of the insert operation.
		Sample response XML:
		`<?xml version = '1.0' encoding = 'utf-8'?>`
		`<SiebelREST> <ErrorCode>0x0</ErrorCode>`
		`<PrimaryRowId>42-JIMUV</PrimaryRowId>`
		`<ListOfAccountAttOrderAtt>`
		`<Account> <Name>AttachAbhishek36</Name>`
		`<Location>HeadQuarter</Location>`
		`<Currency_0x20_Code>BRL</Currency_0x20_Code>`
		`<Primary_0x20_Organization>Default Organization</Primary_0x20_Organization>`
		`</Account> </ListOfAccountAttOrderAtt>`
		`</SiebelREST>`

Table 8. Sample URIs

Task	HTTP Request Type	Sample URI

To invoke upsert operation	POST	http://*weblogic_server_IP*:*port*/oracle-crm/api/rest/siebel/*JNDI_ra_name*/*IO_name*?updatemode=upsert
		Sample request XML:
		`<?xml version="1.0" encoding="ISO-8859-1"?>`
		`<SiebelREST>`
		`<ListOfAccountAttOrderAtt>`
		`<Account>`
		`<Name>InsertAccount1</Name>`
		`<Location>HeadQuarter</Location>`
		`<Currency_0x20_Code>USD</Currency_0x20_Code>`
		`<Primary_0x20_Organization>Default Organization</Primary_0x20_Organization>`
		`</Account>`
		`</ListOfAccountAttOrderAtt>`
		`</SiebelREST>`

Table 8. Sample URIs

Task	HTTP Request Type	Sample URI
To fetch schema of a repository object	GET	http://*weblogic_server_IP*:*port*/oracle-crm/api/rest/siebel/*JNDI_ra_name*/*IO_name*/schema

To retrieve attachments	POST	http://*weblogic_server_IP*:*port*/oracle-crm/api/rest/siebel/*JNDI_ra_name*/*IO_name*?updatemode=insert
		Make sure that the integration object has an attachment integration component using the following code structure:
		NOTE: The following code is part of a larger message. For the complete message structure, see "Sample XML for Uploading Attachments" on page 98.
		`<Attachments>`
		`<Document>`
		`<Attachment_0x20_Data>QXR0YWNobWVudHMgZm9yIFJFU1QgQVBJLg==</Attachment_0x20_Data>`
		`<TimedOut>false</TimedOut>`
		`<AttachmentIsTextData>false</AttachmentIsTextData>`
		`<ContentId>1234</ContentId>`
		`<Extension>txt</Extension>`
		`</Document>`
		`</Attachments>`
		NOTE: When the value is set to true (`<AttachmentIsTextData>false</AttachmentIsTextData>`), the attachment data is not decoded. The created attachment on the Siebel Server side is the same as that provided in the `<Attachment_0x20_Data>` tag.

Sample XML for Uploading Attachments

The following provides sample XML for uploading attachments. For a sample URI, see the POST call for retrieving attachments in Table 8 on page 93.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<SiebelREST>

    <ListOfAccountAttOrderAtt>
```

```
       <Account>

          <Name>Insert_Account2</Name>

          <Location>HeadQuarter</Location>

          <Currency_0x20_Code>BRL</Currency_0x20_Code>

          <Primary_0x20_Organization>Default Organization</
       Primary_0x20_Organization>

             <ListOfAccount_0x20_Attachment>

                <Account_0x20_Attachment>

                   <AccntFileSrcType>FILE</AccntFileSrcType>

                   <AccntFileName>att</AccntFileName>

                   <AccntFileExt>txt</AccntFileExt>

                    <Accnt_0x20_Attachment_0x20_Id>1234</
       Accnt_0x20_Attachment_0x20_Id>

                </Account_0x20_Attachment>

             </ListOfAccount_0x20_Attachment>

       </Account>

   </ListOfAccountAttOrderAtt>

   <Attachments>

       <Document>

          <Attachment_0x20_Data>QXR0YWNobWVudHMgZm9yIFJFU1QgQVBJLg==</
       Attachment_0x20_Data>

          <TimedOut>false</TimedOut>

          <EndOfData>true</EndOfData>

          <AttachmentIsTextData>false</AttachmentIsTextData>

          <ContentId>1234</ContentId>

          <Extension>txt</Extension>

       </Document>

   </Attachments>

</SiebelREST>
```

Example of Using Code to Access Data from the Siebel File System

This topic provides one example of how you can use code to access data from the Siebel File System. You might use code to access data differently, depending on your business model.

If your deployment must include attachment data for any business component, then you must define an integration object that allows you to retrieve the attachment data from the Siebel File System. You use Siebel Tools to create an integration object that includes the required configuration. In this example, you access attachments for the Account business component.

To access data from the Siebel File System

1. Log in to Siebel Tools.
2. From the View menu, choose the Options menu item, and then click the Object Explorer tab.
3. In the Object Explorer Hierarchy window, make sure the Integration Object tree check box is selected, and then click OK.
4. In the Object Explorer, click Integration Object, and then create a new record in the Integration Objects list using the values from the following table.

Property	Value
Name	Account Attach IO
Base Object Type	Siebel Business Object
Business Object	Account
Project	Any project that is locked

5. In the Object Explorer, expand the Integration Object tree, and then click Integration Component.
6. In the Integration Components list, create a new record using the values from the following table.

Property	Value
External Name Context	Account
Name	Account
Parent Integration Component	Leave this blank
External Name	Account
External Sequence	1
Cardinality	Zero or More

7. In the Object Explorer, expand the Integration Components tree, and then click Integration Component Field.

8 In the Integration Component Fields list, add four new records using the values from the following table.

Name	Data Type	Length
Integration Id	DTYPE_TEXT	30
Location	DTYPE_TEXT	50
Name	DTYPE_TEXT	100
Primary Organization	DTYPE_TEXT	100

NOTE: You can add any other fields from the Account business component. For brevity, only a small subset are included in this example.

9 To create a new Account Attachment integration object, repeat Step 6. Use the values from the following table.

Property	Value
External Name Context	Account Attachment
Name	Account Attachment
Parent Integration Component	Account
External Name	Account Attachment
External Sequence	2
Cardinality	Zero or More

10 Repeat Step 8 and Step 9, but this time add the integration component fields to the Account Attachment integration component.

Include all the fields that you require that are defined in the Account Attachment business component. For the Data Type and Length properties, use the same values that are defined for each business component field.

NOTE: Add only the integration component fields that your require. Make sure that the Account Attachment integration component is last in sequence. For more information on creating attachment integration objects, see *Integration Platform Technologies: Siebel Enterprise Application Integration*.

11 Activate the new integration objects.

For more information about activating integration objects, see *Integration Platform Technologies: Siebel Enterprise Application Integration*.

You can now access data that resides in the Siebel File System for the Account Attachment business component.

Related Topics

“Example of Using Code to Insert Data into the Siebel File System” on page 102

Example of Using Code to Insert Data into the Siebel File System

This topic is one example of using code to insert data into the Siebel File System. In this example you insert data for the Account Attachment business component. You might use this feature differently, depending on your business model. For more information about file attachments, see *Integration Platform Technologies: Siebel Enterprise Application Integration*.

Assume you attach a sample file for an account that contains row ID 1-5GZO. The content of this file is stored in Base64 encoding as follows:

```
VGhpcyBpcyBhIHNhbXBsZSB0ZXh0IGZpbGUNClRoaXMgaXMgYSBzYW1wbGUgdGV4dCBmaWxlLg==
```

You can use any Base64 decoder to decode the file. After decoding, the content is as follows:

```
This is a sample text file

This is a sample text file
```

Use the following URL to access the data from the Siebel File System:

http://localhost:7101/oracle-crm/api/rest/siebel/<your adapter name>/Account%20Attach%20IO/1-5GZO

The XML returned is:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<SiebelREST>

   <NumOutputObjects>1</NumOutputObjects>

   <ErrorCode>0x0</ErrorCode>

   <ListOfAccount_0x20_Attach_0x20_IO>

      <Account>

         <Name>AG Edwards & Sons, Inc</Name>

         <Location>San Francisco</Location>

         <Primary_0x20_Organization>Millennium Healthcare HC ENU

         </Primary_0x20_Organization>

            <ListOfAccount_0x20_Attachment>

               <Account_0x20_Attachment>

               <AccntFileSize>98</AccntFileSize>
               <AccntFileName>TextFile</AccntFileName>
               <Account_0x20_Id>1-5GZO</Account_0x20_Id>
               <AccntFileDate>12/14/2009 16:09:41</AccntFileDate>
               <AccntFileDockStatFlg>E</AccntFileDockStatFlg>
               <AccntFileSrcType>FILE</AccntFileSrcType>
               <AccntFileAutoUpdFlg>Y</AccntFileAutoUpdFlg>
               <AccntFileDockReqFlg>N</AccntFileDockReqFlg>
               <AccntFileExt>txt</AccntFileExt>
               <AccntFileDeferFlg>R</AccntFileDeferFlg>
               <Accnt_0x20_Attachment_0x20_Id>88-1NKG
               </Accnt_0x20_Attachment_0x20_Id>
            </Account_0x20_Attachment>
         </ListOfAccount_0x20_Attachment>
      </Account>
   </ListOfAccount_0x20_Attach_0x20_IO>
   <Attachments>
      <Document>
         <TimedOut>false</TimedOut>
         <EndOfData>true</EndOfData>
         <AttachmentIsTextData>false</AttachmentIsTextData>
         <ContentId>88-1NKG</ContentId>
         <Extension>txt</Extension>
      <Document>
      VGhpcyBpcyBhIHNhbXBsZSB0ZXh0IGZpbGUNClRoaXMgaXMgYSBzYW1wbGUgdGV4dCBmaWxlLg==
      </Document></Document>
   </Attachments>
</SiebelREST>
```

When the string in the XML is returned, your application that is written to access and process this data, searches for the Document tag and processes the data found between the Document tags.

NOTE: Use localhost only if Oracle WebLogic Server is running on the computer you are using.

Related Topics

“Example of Using Code to Access Data from the Siebel File System” on page 100

8 Working with XSDs Using SAI for Oracle Fusion Middleware

This chapter describes how to design and deploy XML Schema Definitions (XSDs) that are built by selecting base integration objects using Siebel Application Integration for Oracle Fusion Middleware (SAI for Oracle Fusion Middleware). It includes the following topics:

- About Working with XSDs Using SAI for Oracle Fusion Middleware on page 105
- Process of Designing and Deploying XSDs Using SAI for Oracle Fusion Middleware on page 106
- About Using XSDs with Siebel RESTful Services on page 110
- Undeploying XSDs That Were Created Using SAI for Oracle Fusion Middleware on page 110

About Working with XSDs Using SAI for Oracle Fusion Middleware

SAI for Oracle Fusion Middleware provides a dynamic integration object feature that allows for a one-to-one mapping between a business object and an integration object for external Siebel entities. The Siebel Repository has one base integration object for each the existing business objects.

The advantages of using the dynamic integration objects include:

- You can design your own integration object XSDs and use them to invoke Web services.
- The WSDL that is generated during a business service deployment contains only the schema reference and not the complete inline schema definition.
- The XSDs are available at run time as URIs.
- The XSDs allow for custom attributes at different levels to support Siebel properties, such as integration component user properties, integration component keys, and so on. The properties configured on the base integration object are inherited.

Related Topics

"Process of Designing and Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 106

"Designing XSDs Using SAI for Oracle Fusion Middleware" on page 106

"Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 109

"About Using XSDs with Siebel RESTful Services" on page 110

"Undeploying XSDs That Were Created Using SAI for Oracle Fusion Middleware" on page 110

Process of Designing and Deploying XSDs Using SAI for Oracle Fusion Middleware

You use Oracle JDeveloper to design and deploy XML Schema Definitions (XSDs) using SAI for Oracle Fusion Middleware to support various Web service deployments. You can create XSDs for immediate use or save them for future deployment.

To design and deploy XSDs using SAI for Oracle Fusion Middleware, perform the following tasks:

1. "Designing XSDs Using SAI for Oracle Fusion Middleware" on page 106
2. "Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 109

Related Topics

"About Working with XSDs Using SAI for Oracle Fusion Middleware" on page 105

"About Using XSDs with Siebel RESTful Services" on page 110

"Undeploying XSDs That Were Created Using SAI for Oracle Fusion Middleware" on page 110

Designing XSDs Using SAI for Oracle Fusion Middleware

This topic describes how to design XML Schema Definitions (XSDs) using SAI for Oracle Fusion Middleware. For general information about working with XSDs, see "About Working with XSDs Using SAI for Oracle Fusion Middleware" on page 105.

This task is a step in "Process of Designing and Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 106.

To design an XSD using SAI for Oracle Fusion Middleware

1. Start Oracle JDeveloper.
2. Make sure you have specified the database parameters specific to your environment in the General tab in the SAI for Oracle Fusion Middleware Preferences.

   For information about specifying database parameters, see "Setting Preferences for SAI for Oracle Fusion Middleware" on page 29.
3. From the application-level menu, select File, and then click the New menu item.
4. In the New Gallery dialog box, perform the following:
   a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.
   b. Select the Design item, and then click OK.
5. At the Welcome to the Design Wizard screen, click Next.
6. Select Siebel XSD Design for the artifact type, and then click Next.

7 Select the base integration object you want to use to design the XSD by doing the following:

a Click Query Integration Objects, and in the Enter Query dialog box, query for potential integration objects (using the percent sign (%) wildcard), and then click OK.

TIP: Alternatively, you can select an integration object from the prepopulated Base Integration Object drop-down list without performing a query.

The Base Integration Object field populates with all the integration objects for that query. For example, if you enter %Account% for your query, then all the Account integration objects become available for selection.

b From the Base Integration Object drop-down list, select the integration object you want to use, and then click Load.

The load operation queries the integration object selected and populates the Available XSD tree with the integration object and its user properties as well as the integration component, its fields, user properties, and user keys that are configured for that integration object. However, only those integration components indicated as Active in Siebel Tools appears.

NOTE: Loading the integration object definition might take some time.

8 Use the arrows to move the integration object and its user properties and the integration component fields, user properties, and user keys that are configured for that integration object from the Available XSD tree to the Selected XSD tree.

CAUTION: A requirement of XSD generation is that at least one user key is defined for an integration component. If this requirement is not met, then an error occurs when you generate the XSDs. To define user keys in the Selected XSD tree, see Step 9.

9 (Optional) If you need to create new or update user properties or create new integration component keys or integration component key fields, then perform one or more of the following, otherwise skip to Step 10.

- **New User Property.** Right-click the integration component or integration component field for which you want to create new user properties, enter the details, and then click OK to return to the Design XSD wizard page.

- **Update User Property.** Right-click the integration component user property you want to update, replace the user property value, and then click OK to return to the Design XSD wizard page.

- **New Integration Component Key**. Right-click the integration components or integration component fields for which you want to create new integration component keys, enter the details, and then click OK to return to the Design XSD wizard page.

- **New Integration Component Key Field.** Right-click the integration component key for which you want a new integration component key field, select the integration component key field you want to use from the autopopulated list, and then click OK to return to the Design XSD wizard page.

  The integration component user key is updated with the new field. If the field already exists in the user key, then no changes occur.

10 Click Save XSD Design.

If the XSD file already exists, then the Specify XSD Name dialog box appears asking if you want to overwrite it. Do one of the following:

- If you choose Yes, then the current XSD definition will be overwritten with the same file name.
- If you choose No, then another dialog box appears, which enables you to specify a new XSD file name. Click OK to continue.

The naming convention for saved XSD files is baseioname_X_Y_seq.xsd, where:

- *baseioname* is the base integration object name
- *X* is the base integration object major number
- *Y* is base integration object minor number
- *seq* is the next sequence available when there is an existing file

11 (Optional) Click Show XSDs to view a list of the XSD files you have just saved, and then click OK to return the Design XSD wizard page.

12 Click Next to continue.

13 Review the code generation summary, and then click Finish to start generation.

If there is an existing archive folder structure, then a warning appears asking if you want to overwrite it. One of the following happens:

- If you click Yes, then the folder structure is overwritten.
- If you click No, then a dialog box appears in which you can save the new folder structure in a directory of your choice.

The XSD files are saved in memory until you click Finish. The XSD file will be generated in the XSD file path you defined in the General tab in the Siebel Application Integration for Oracle Fusion Middleware Preferences.

NOTE: If the deployment fails, then check the SiebelToolkitExtension.log file for details and troubleshooting information. Also make sure that at least one user key is defined for the integration components you are using.

Related Topics

"About Working with XSDs Using SAI for Oracle Fusion Middleware" on page 105

"Process of Designing and Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 106

"Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 109

"About Using XSDs with Siebel RESTful Services" on page 110

"Undeploying XSDs That Were Created Using SAI for Oracle Fusion Middleware" on page 110

Deploying XSDs Using SAI for Oracle Fusion Middleware

This topic describes how to deploy XSDs using SAI for Oracle Fusion Middleware.

This task is a step in "Process of Designing and Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 106.

To deploy XSDs using SAI for Oracle Fusion Middleware

1. Make sure you have deployed an appropriate Siebel Resource Adapter.

   For information about deploying Siebel Resource Adapters, see "Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware" on page 46.

2. Start Oracle JDeveloper.
3. From the application-level menu, select File, and then click the New menu item.
4. In the New Gallery dialog box, perform the following:
   - a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.
   - b. Select the Deployment item, and then click OK.
5. At the Welcome to the Deployment wizard screen, click Next.
6. Select XSD, and then click Next.
7. Select Deploy XSDs, and then click Next.
8. Select the XSD files you want to deploy by moving them from the Available section to the Selected section.

   NOTE: The path is the XSD File Path parameter specified in the SAI for Oracle Fusion Middleware Preferences. For more information, see "Setting Preferences for SAI for Oracle Fusion Middleware" on page 29.

9. (Optional) If you want to preview an XSD file that will be deployed, then select the XSD file in the Selected section, click View XSD, and then click Close to return to the Choose XSDs wizard page.
10. Enter the deployed Siebel Resource Adapter information by doing the following:
    - a. Enter details of the resource adapter you are using for the deployment.
    - b. Enter the resource adapter connection factory JNDI name.
    - c. Click Next.

    NOTE: These settings are specified in the Deployment tab in the SAI for Oracle Fusion Middleware Preferences. For more information, see "Setting Preferences for SAI for Oracle Fusion Middleware" on page 29.

11. Enter the Oracle WebLogic Server parameters, and then click Next.

    For a description of the parameters, see *Siebel Application Integration for Oracle Fusion Middleware: Oracle JDeveloper Online Help*.

12. Review the summary, and then click Finish.

    The XSD file is deployed to Oracle WebLogic Server.

Related Topics

"About Working with XSDs Using SAI for Oracle Fusion Middleware" on page 105

"Process of Designing and Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 106

"Designing XSDs Using SAI for Oracle Fusion Middleware" on page 106

"About Using XSDs with Siebel RESTful Services" on page 110

"Undeploying XSDs That Were Created Using SAI for Oracle Fusion Middleware" on page 110

About Using XSDs with Siebel RESTful Services

After you deploy your XSD files to Oracle WebLogic Server, you can use these files for CRUD operations, similar to other integration objects used for REST calls.

The following is a sample URI pattern:

http://*weblogic_server_IP*:*port*/oracle-crm/api/rest/siebel/*JNDI_ra_name*/
deployed_xsd_name?searchexpr=[*IntCompName.IntCompFieldName*] *search_expression*

The following is a sample URI example:

http://localhost:7101/oracle-crm/api/rest/siebel/siebel.*SiebelResourceAdapterConnFactory*/
Base%20Account_0_0.xsd?searchexpr=[Account.Name]like"A*"

For more information about working with Siebel RESTful Services, see Chapter 7, "Working with Siebel RESTful Services."

Related Topics

"About Working with XSDs Using SAI for Oracle Fusion Middleware" on page 105

"Process of Designing and Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 106

"Designing XSDs Using SAI for Oracle Fusion Middleware" on page 106

"Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 109

"Undeploying XSDs That Were Created Using SAI for Oracle Fusion Middleware" on page 110

Undeploying XSDs That Were Created Using SAI for Oracle Fusion Middleware

This topic describes how to undeploy XSDs that were created using SAI for Oracle Fusion Middleware.

This task is a step in "Process of Designing and Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 106.

1. Make sure you have deployed an appropriate Siebel Resource Adapter.

   For information about deploying Siebel Resource Adapters, see "Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware" on page 46.

2. Start Oracle JDeveloper.
3. From the application-level menu, select File, and then click the New menu item.
4. In the New Gallery dialog box, perform the following:
   a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.
   b. Select the Deployment item, and then click OK.
5. At the Welcome to the Deployment wizard screen, click Next.
6. Select XSD, and then click Next.
7. Select Undeploy XSDs, and then click Next.
8. Enter the deployed Siebel Resource Adapter information by doing the following:
   a. Enter details of the resource adapter you are using for the deployment.
   b. Enter the resource adapter connection factory JNDI name.
   c. Click Next.

   NOTE: These settings are specified in the Deployment tab in the SAI for Oracle Fusion Middleware Preferences. For more information, see "Setting Preferences for SAI for Oracle Fusion Middleware" on page 29.

9. Enter the Oracle WebLogic Server parameters, and then click Next.

   For a description of the parameters, see *Siebel Application Integration for Oracle Fusion Middleware: Oracle JDeveloper Online Help*. Use the arrows to move the deployed XSD files from the Available section to the Selected section, and then click Next.

10. Review the summary, and then click Finish.

    The selected XSD files will be undeployed from Oracle WebLogic Server.

Related Topics

"About Working with XSDs Using SAI for Oracle Fusion Middleware" on page 105

"Process of Designing and Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 106

"Deploying XSDs Using SAI for Oracle Fusion Middleware" on page 109

"About Using XSDs with Siebel RESTful Services" on page 110

9 Migrating Siebel Web Services to Oracle WebLogic Server

This chapter describes how to migrate existing Siebel Web services to Oracle WebLogic Server. It includes the following topics:

Benefits of Migrating Siebel Web Services to Oracle WebLogic Server

The following are some of the reasons you might want to migrate existing Siebel Web services to Oracle WebLogic Server:

- **Performance.** Siebel Application Integration for Oracle Fusion Middleware (SAI for Oracle Fusion Middleware) offers a superior performance profile in relation to Siebel EAI.
- **Session management.** SAI for Oracle Fusion Middleware includes built-in session management capabilities.
- **Scalability.** SAI for Oracle Fusion Middleware scales more linearly and to a higher level than Siebel EAI.

For more information about the benefits of using SAI for Oracle Fusion Middleware, see "About SAI for Oracle Fusion Middleware" on page 9. For more information about how SAI for Oracle Fusion Middleware and Siebel EAI relate, see "How SAI for Oracle Fusion Middleware Relates to Siebel EAI" on page 10.

Process of Migrating SOAP Services Using SAI for Oracle Fusion Middleware

There are two phases in working with SOAP (Simple Object Access Protocol) services using SAI for Oracle Fusion Middleware. The first phase is the code generation phase where you use Oracle JDeveloper to create Java artifacts that represent one or more SOAP services. These services may relate to a migration of an existing Siebel EAI Web service or a completely new service implementation. The second phase takes the output of the first phase (the generated code), and deploys it to Oracle WebLogic Server.

NOTE: SAI for Oracle Fusion Middleware supports migration of single-port Siebel Web services. Multiple-port migration is not supported.

To migrate SOAP services using SAI for Oracle Fusion Middleware, perform the following tasks:

1. “Migrating Existing Siebel Inbound Web Services to Oracle WebLogic Server” on page 114
2. “Migrating Existing Siebel Outbound Web Services to Oracle WebLogic Server” on page 115

Migrating Existing Siebel Inbound Web Services to Oracle WebLogic Server

This topic describes how to migrate existing Siebel inbound Web services to Oracle WebLogic Server.

This task is a step in “Process of Migrating SOAP Services Using SAI for Oracle Fusion Middleware” on page 113.

To migrate existing Siebel inbound Web services to Oracle WebLogic Server

1. Launch Oracle JDeveloper, if not already active.
2. Select File, and then click the New menu item.
3. In the New Gallery dialog box, perform the following:
   a. In the Business Tier category, select Siebel Application Integration for Oracle Fusion Middleware.
   b. Select the Migration item, and then click OK.
4. At the Welcome to the Migration Wizard screen, click Next.
5. Select the service or services you want to migrate by checking the relevant check boxes from the list of Siebel Inbound Web Services. Check the Auto-deploy to Oracle WebLogic Server option if you want to directly deploy the migrated services, and then click Next.

   NOTE: The list populates automatically with the inbound Web services that are noted as active in the Administration - Web Services screen, Inbound Web Services view in the Siebel application.

6. If you selected the Auto-deploy to Oracle WebLogic Server option, then enter the inbound deployment parameters, and then click Next. Otherwise, skip to Step 8.

   For information about these parameters, see Step 8 on page 73 in “Deploying Enterprise Java Beans Using SAI for Oracle Fusion Middleware.”

7. Enter the Oracle WebLogic Server parameters, and then click Next.

   For information about these parameters, see Step 7 on page 46 in “Process of Deploying Siebel Resource Adapters Using SAI for Oracle Fusion Middleware.”

8. Click Finish to start the migration.

   The code required to expose existing inbound Web services generates and is available for deployment to Oracle WebLogic Server at a later time. If the autodeploy option is selected, then the Web service will be deployed to Oracle WebLogic Server.

   NOTE: If code was previously generated for this service, then a warning appears asking if you want to overwrite it. If you click Yes, then the folder structure is overwritten. If you click No, then a dialog box appears in which you can name a new folder structure and save that structure in a directory of your choice.

Migrating Existing Siebel Outbound Web Services to Oracle WebLogic Server

This topic describes how to migrate existing Siebel outbound Web services to Oracle WebLogic Server. When a request is submitted from the Siebel application (which has outbound Web services implemented to a third party outbound Web service server), the request is sent using the framework provided by SAI for Oracle Fusion Middleware.

This task is a step in "Process of Migrating SOAP Services Using SAI for Oracle Fusion Middleware" on page 113.

To migrate existing Siebel outbound Web services to Oracle WebLogic Server

1. Change the value of the EnableNewOutboundDispatcher server component parameter for the EAI object manager component for your Siebel application by doing the following:

   a. From the Server Manager command-line interface, enter the following command:

      ```
      change param EnableNewOutboundDispatcher=Y for comp EAI_OM_Alias
      ```

      For example:

      ```
      change param EnableNewOutboundDispatcher=Y for comp EAIObjMgr_enu
      ```

      NOTE: By default, this parameter is set to N.

   b. From the Server Manager command-line interface, enter the following command:

      ```
      list param EnableNewOutboundDispatcher for comp EAI_OM_Alias
      ```

2. Deploy the outbound Web service WSDL on Oracle WebLogic Server.

   For more information, see "Deploying Outbound Web Services Using SAI for Oracle Fusion Middleware" on page 79.

3. Import the outbound Web service WSDL into Siebel Tools by doing the following:

   a. Start Siebel Tools.

   b. From the Tools menu, choose File, New Object, EAI, the Web Service menu item, and then click OK.

c. Browse for the WSDL you want to import, and then click Next.

   NOTE: Make sure you have locked the project in which the objects created from the WSDL file will reside.

d. Check the Deploy the Integration Object(s) and the Proxy Business Service(s) check box, and then click Finish.

e. Compile the server SRF.

4. Test the outbound Web service.

   For instructions, see "Example of Testing Migrated Outbound Web Services by Using SAI for Oracle Fusion Middleware" on page 116.

Example of Testing Migrated Outbound Web Services by Using SAI for Oracle Fusion Middleware

This topic gives one example of testing a migrated outbound Web service by using SAI for Oracle Fusion Middleware. You might use this feature differently, depending on your business model.

To test a migrated outbound Web service by using SAI for Oracle Fusion Middleware

1. Navigate to the Administration - Business Service screen, then the Simulator view.
2. Create a new business service with the following values:
   - Service name is *Web_Service_Name*

     where *Web_Service_Name* is obtained from the import of the external WSDL in Siebel Tools.
   - Method Name is *MethodName*

     where you select a *MethodName* from the Proxy Business Service.
3. Pass the XML for the method being called.

   For example, the following is sample XML input created for GlobalWeather:

```
<?xml version="1.0" encoding="UTF-8"?>

<?Siebel-Property-Set EscapeNames="true"?>

<PropertySet remote_undjndi_undname="eis.siebel.outbound.globalweather">

   <GetWeatherSoapIn_clnparameters>

      <ListOfGetWeather>

         <GetWeather CountryName="INDIA" CityName="BANGALORE">

         </GetWeather>

      </ListOfGetWeather>
```

```
    </GetWeatherSoapIn_clnparameters>

</PropertySet>
```

10 Troubleshooting SAI for Oracle Fusion Middleware

This chapter describes how to enable and examine troubleshooting information on the Oracle WebLogic Server and provides guidelines for resolving errors when using Siebel Application Integration for Oracle Fusion Middleware (SAI for Oracle Fusion Middleware). It includes the following topics:

- Enabling and Examining Troubleshooting Information on the Oracle WebLogic Server on page 117
- Setting Log Levels for SAI for Oracle Fusion Middleware
- Troubleshooting Errors for SAI for Oracle Fusion Middleware on page 119

Enabling and Examining Troubleshooting Information on the Oracle WebLogic Server

You can enable and examine troubleshooting information on Oracle WebLogic Server.

To enable and examine troubleshooting information on the Oracle WebLogic Server

1. In the Oracle WebLogic Server Administration Console, click Environment, click Servers, and then drill down on the Oracle WebLogic Server, for example, AdminServer.
2. Click the Debug tab, make sure that the WebLogic check box is selected, and then click Enable.
3. Expand the WebLogic tree.
4. Make sure that the Connector check box and Security check box are both selected, and then click Enable.
5. In the Domain Structure window, expand the Diagnostics tree, and then click Log Files.
6. In the Summary of Log Files window, choose the ServerLog option, and then click View.
7. After you finish examining the troubleshooting information, deselect the check marks that you added in Step 2 and Step 4.

   Removing these check marks helps to prevent performance degradation.

Setting Log Levels for SAI for Oracle Fusion Middleware

This topic describes the various log levels for troubleshooting SAI for Oracle Fusion Middleware and how to set them.

NOTE: Typically, you set the log level as part of the process of deploying SAI for Oracle Fusion Middleware. However, you can reset the level at any time.

To set log levels for SAI for Oracle Fusion Middleware

1. Start Oracle JDeveloper.
2. From the application-level menu, select Tools, and then the Preferences menu item.
3. From the explorer, scroll down to select Siebel Application Integration for Oracle Fusion Middleware.
4. Under Deployment tab, set the log level specific to your deployment.

   The log levels are described in the following table.

Log Level	Description
FATAL	This level logs severe errors that cause termination of an application.
ERROR	This level logs run-time errors or unexpected conditions, however, only the specified error in the log file is logged.
WARN	This level indicates a potential problem, such as errors in run-time situations that are undesirable or unexpected, but not necessarily causing run-time termination.
INFO	This level provides information on run-time events, such as startup and shutdown.
DETAIL	This level logs run-time errors or unexpected conditions. This level provides the most information.
DEBUG	This level logs detailed information on the flow through the data management system (the Siebel application server and the Oracle WebLogic Server) and prints the stack trace.

Troubleshooting Errors for SAI for Oracle Fusion Middleware

To resolve problems while working with SAI for Oracle Fusion Middleware, look for the symptom or error message in Table 9. Additional troubleshooting tips follow the table.

Table 9. Siebel Application Integration for Oracle Fusion Middleware Problems and Solutions

Symptom or Error Message	Diagnostic Steps or Cause	Solution
REST Automation	Automation fails	Make sure the test cases are in ANSI file format. Open each test case, and then select save as Change Encoding to ANSI.
Managed setup errors	None	See "Troubleshooting Setup Errors for Managed Applications" on page 119.

Troubleshooting Setup Errors for Managed Applications

To troubleshoot setup errors for managed applications in the Oracle WebLogic Server, perform one or more of the following:

- Make sure the resource adapter is deployed on the Oracle WebLogic Server Administration Console, and then test the connection.
- Using Oracle JDeveloper, make sure that the parameters match those in the deployed resource adapter.
  - In the web.xml file, verify the <res-ref-name> parameter.
  - In the weblogic.xml file, verify the <res-ref-name> and <jndi-name> parameters.
- Make sure that the JAR files in the resource adapter path are the same as the resource adapter that is used in Oracle JDeveloper.

For more information for troubleshooting setup errors for managed applications, see "Configuring Siebel Server Load Balancing and SSL for SAI for Oracle Fusion Middleware" on page 55.

Glossary

Administration Console

A Web browser-based, graphical user interface. It is used to manage an Oracle WebLogic Server domain.

administration server

Provides a central point for managing an Oracle WebLogic Server domain.

Apache Ant (ANT)

ANT (Another Neat Tool) is a software tool for automating software build processes using Java. Ant uses XML-based configuration files (called build.xml by default) to execute tasks written in Java. Oracle provides a number of Ant tasks that help you generate important Web Service-related artifacts.

The Apache Web site provides other useful Ant tasks for packaging EAR, WAR, and Enterprise Java Bean (EJB) JAR files. For more information, see the Apache Ant Manual at

http://jakarta.apache.org/ant/manual/

NOTE: ANT is similar to the UNIX make command, but ANT uses XML to describe the build process and its dependencies.

application integration adapter

A software component that connects two incompatible business applications with the specific purpose of facilitating information exchange.

application programming interface (API)

A series of software routines and development tools that comprise an interface between a computer application and lower-level services and functions (such as the operating system, device drivers, and other software applications). APIs serve as building blocks for programmers putting together software applications.

application server

A server designed to host applications and their environments, permitting server applications to run. A typical example is Oracle Application Server, which is able to host Java, C, C++, and PL/SQL applications in cases where a remote client controls the interface. Another example is the Siebel Server.

artifact

Any resulting physical entity, or data, generated and persisted to disk by the Oracle WebLogic Diagnostic Framework that can be used later for diagnostic analysis. For example, the diagnostic image file that is created when the server fails is an artifact. The diagnostic image artifact is provided to support personnel for analysis to determine why the server failed. The Oracle WebLogic Diagnostic Framework produces a number of different artifacts.

base domain

The domain you use in the Administration Console to configure administrative options that apply to all servers in the current domain.

business-to-business (B2B)

Describes the communication between businesses in the selling of goods and services to each other. The software infrastructure to enable this is referred to as an exchange.

business-to-consumer (B2C)

Describes the communication between businesses and consumers in the selling of goods and services.

class

A category of objects. Classes allow grouping of data and method.

CLASSPATH

The operating system environmental variable that the Java Virtual Machine (JVM) uses to find the classes it needs to run applications.

cluster

An entity of a particular Oracle WebLogic Server domain. A deployment in which multiple Oracle WebLogic Server instances (servers) run simultaneously and work together can provide increased scalability and reliability. The servers that constitute a cluster can run on the same computer or on different computers.

connection factory

A public interface that a Java EE application component uses to access a connection instance. The component then uses it to connect to the underlying enterprise information system (EIS). Examples of connections include: database connections and JMS (Java Message Service) connections.

Common Object Request Broker API (CORBA)

An Object Management Group standard for communicating between distributed in a network. These self-contained software modules can be used by applications running on different platforms or operating systems. CORBA objects and their data formats and functions are defined in the Interface Definition Language (IDL), which can be compiled in a variety of languages including Java, C, C++, Smalltalk, and COBOL.

container

An application program or subsystem in which the program building block, known as a component, is run.

credential

Security-related attribute of a subject that might contain information used to authenticate the subject to new services. Types of credentials include, but are not limited to, user name and password combinations and public key certificates.

credential mapping

The process whereby a legacy system's database is used to obtain an appropriate set of credentials to authenticate users to a target resource. The Oracle WebLogic Server uses credential mapping to map credentials used by Oracle WebLogic Server users to credentials used in a legacy (or any remote) system. The Oracle WebLogic Server then uses the credential maps to log in to a remote system on behalf of a subject that has already been authenticated.

domain

An interrelated set of Oracle WebLogic Server resources that is managed as a unit. It is the basic administrative unit of the Oracle WebLogic Server. A domain includes one or more Oracle WebLogic Server instances, which can form clusters, instances, or a combination of both.

Domain Name System (DNS)

A system for naming computers and network services that is organized into a hierarchy of domains. DNS is used in TCP/IP networks to locate computers through user-friendly names. DNS resolves a friendly name into an IP address, which is understood by computers.

EAR (enterprise application archive)

A Java 2 Platform Enterprise Edition (Java EE) archive file that is used in deploying applications on a Java EE application server. EAR files simplify application deployment by reducing the possibility of errors when moving an application from development to test, and test to production.

See also “JAR (Java archive file)” and “WAR (Web application archive file).”

element

The basic logical unit of an XML document that can serve as a container for other elements such as children, data, and attributes and their values. Elements are identified by start-tags, such as <name>, and end-tags, such as </name>, or in the case of empty elements, <name/>.

Enterprise Application Integration (EAI)

See “Siebel Enterprise Application Integration (EAI).”

enterprise information system (EIS)

Provides a technology platform that enables organizations to integrate and coordinate their business processes. An EIS is a data management system that operates centrally and makes shareable information available on multiple functional levels and management hierarchies.

Enterprise Java Bean (EJB)

An independent program module that runs within a JVM (Java Virtual Machine) on the server. CORBA provides the infrastructure for EJBs, and a container layer provides security, transaction support, and other common functions on any supported server.

eXtensible Markup Language

See “XML.”

Extensible Style Sheet Language Transformation (XSLT)

A declarative, XML-based language used for the transformation of XML documents into other XML documents.

failover

Failover is when one node fails in a clustered environment, but transactions continue to process on other nodes. There is no interruption of the end-user experience.

HTTP protocol

Hypertext Transfer Protocol. A protocol that provides the language that enables browsers and the origin server to communicate. On the Internet, HTTP is a standard protocol that is used to exchange data.

HTTP request header

A header that enables Web browsers to pass additional information about the request and about itself to the origin server.

HTTP request method

A method included in the HTTP request that specifies the purpose of the client's request.

HTTPS protocol

Secure Hypertext Transfer Protocol. A protocol that uses the Secure Sockets Layer (SSL) to encrypt and decrypt user page requests as well as the pages that are returned by the origin server. On the Internet, HTTPS is a standard protocol that is used to exchange data in a secured environment.

instantiate

To create an instance of a Java class.

Integrated Development Environment (IDE)

A tool that develops application artifacts. One example is Oracle Developer.

See also “Oracle JDeveloper.”

IP address

Each computer on the network is assigned a unique IP address, which is made up of the network ID and a unique host ID. This address is typically represented in dotted-decimal notation, with the decimal value of each octet separated by a period, for example 144.45.9.22.

JAR (Java archive file)

Contains the class, image, and sound files for a Java application or applet. JAR files might also be compressed.

See also “EAR (enterprise application archive)” and “WAR (Web application archive file).”

Java

A high-level programming language where applications run in a virtual machine known as a JVM. The JVM is responsible for all interfaces to the operating system. This architecture permits developers to create Java applications that can run on any operating system or platform that has a JVM.

Java API for XML Messaging (JAXM)

Enables applications to send and receive document-oriented XML messages using a pure Java API.

JavaBeans

An independent program module that runs within a JVM, typically for creating user interfaces on the client. Also known as Java Bean (two words). The server equivalent is called an Enterprise JavaBean (EJB).

See also “Enterprise Java Bean (EJB).”

Java Database Connectivity (JDBC)

The programming API that enables Java applications to access a database through the SQL language. JDBC drivers are written in Java for platform independence but are specific to each database.

Java Developer's Kit (JDK)

The collection of Java classes, runtime, compiler, debugger, and usually source code for a version of Java that makes up a Java development environment. JDKs are designated by versions, and Java 2 is used to designate versions from 1.2 onward.

Java EE Connector Architecture (JCA)

A Java-based technology solution for connecting application servers and enterprise information systems (EIS) as part of EAI solutions. For more information on JCA, see:

http://java.sun.com/j2ee/connector

Java Message Service (JMS)

Java Message Oriented Middleware (MOM) API for sending messages between two or more clients. It allows the communication between different components of a distributed application to be loosely coupled, reliable, and asynchronous.

Java Platform, Enterprise Edition (Java EE)

A platform-independent, Java-centric environment for developing, building, and deploying Web-based enterprise applications online. The Java EE platform consists of a set of services, APIs, and protocols that provide the functionality for developing multitiered, Web-based applications.

Java Server Page (JSP)

An extension to servlet functionality that provides a simple programmatic interface to Web pages. JSPs are HTML pages with special tags and embedded Java code that is executed on the Web or application server. JSPs provide dynamic functionality to HTML pages. They are actually compiled into servlets when first requested and run in the servlet container.

JavaServer Faces (JSF)

A standard Java framework for building Web applications. It simplifies development by providing a component-centric approach to developing Java Web user interfaces. JSF offers rich and robust APIs that provide programming flexibility and makes sure that applications are well designed with greater maintainability by integrating the Model-View-Controller (MVC) design pattern into its architecture.

JavaServer Faces JavaServer Page (JSF JSP)

JSF JSPs differ from plain JSPs through their support of Oracle ADF Faces components for the user interface and JSF technology for page navigation. JSF JSP pages leverage the advantages of the Oracle Application Development Framework (Oracle ADF) by using the ADF Model binding capabilities for the components in the pages.

Java Messaging Service (JMS)

The JMS API is a protocol for communication that provides asynchronous communication between components in a distributed computing environment.

Java Web service (JWS)

An ordinary Java class file that uses annotations to specify the shape and characteristics of a Web service.

Java Naming and Directory Interface (JNDI)

A programming interface for connecting Java programs to naming and directory services such as DNS, LDAP, and NDS.

Java Virtual Machine (JVM)

The Java interpreter that converts the compiled Java bytecode into the machine language of the platform and runs it. JVMs can run on a client, in a browser, in a middle tier, on an intranet, on an application server, or in a database server.

JAX-RS (Java API for RESTful Web Services)

A Java programming language API that provides support in creating Web services according to the Representational State Transfer (REST) architectural style.

load balancing

A feature in which HTTP requests are distributed among origin servers so that no single server is overloaded.

managed server

In a production environment, a managed server hosts applications and the resources needed by those applications. A domain, which is a logically related group of Oracle WebLogic Server resources, can have any number of managed servers. An administration server manages these servers.

Oracle Fusion Middleware

A suite of Oracle software products, including Java EE and developer tools, integration services, business intelligence, collaboration, and content management. It is used for the development, deployment, and management of service-oriented architecture (SOA) and to facilitate integration with existing applications and systems from other software vendors.

ORACLE_HOME

The operating system environment variable that identifies the location of the Oracle database installation for use by applications.

Oracle JDeveloper

An integrated development environment (IDE) for developing and deploying Java-based service-oriented architecture (SOA) applications and user interfaces.

Oracle WebLogic Server

A scalable, enterprise-ready Java Platform, Enterprise Edition (Java EE) application server. The Oracle WebLogic Server infrastructure supports the deployment of many types of distributed applications and is an ideal foundation for building applications based on Service Oriented Architectures (SOA). SOA is a design methodology aimed at maximizing the reuse of application services.

origin server

A server that is either an application Web server for internal sites or a proxy server for external sites outside a firewall.

path name

The name of a resource that reflects its location in the repository hierarchy. A path name is composed of a root element (the first /), element separators (/) and various subelements (or path elements). A path element might be composed of any character in the database character set except ("\", "/"). These characters have a special meaning for Oracle XML DB. Forward slash is the default name separator in a path name and backward slash might be used to escape characters.

proxy server

An origin server that substitutes for the real server, forwarding client connection requests to the real server or to other proxy servers. Proxy servers provide access control, data and system security, monitoring, and caching.

Remote Method Invocation (RMI)

Specifies how distributed Java applications must operate over multiple Java virtual machines.

repository

The set of database objects in any schema that are mapped to path names. There is one root to the repository ("/"), which contains a set of resources, each with a path name.

resource

An object in the repository hierarchy.

resource name

The name of a resource within its parent folder. Resource names must be unique (potentially subject to case-insensitivity) within a folder. Resource names are always in the UTF-8 character set (NVARCHAR2).

resource adapter

A resource adapter is one implementation of an application integration adapter. In SAI for Oracle Fusion Middleware, the resource adapter is the core Java EE component that provides the link between Siebel Business Applications and Oracle WebLogic Server and is compliant with the Java EE Connector Architecture (JCA) specification.

See also "Siebel Resource Adapter."

Representational State Transfer (REST)

A style of software architecture for distributed hypermedia systems such as the World Wide Web. Hypermedia is an extension to hypertext that supports linking graphics, sound, and video elements in addition to text elements. REST-style architectures consist of clients and servers that govern the proper behavior of participants. Clients initiate requests to servers; servers process requests and return appropriate responses. Requests and responses are built around the transfer of representations of resources. A resource can be essentially any coherent and meaningful concept that might be addressed.

See also "Siebel Representational State Transfer (RESTful) Services."

schema

The definition of the structure and data types within a database. It can also be used to refer to an XML document that supports the XML Schema W3C recommendation.

Secure Shell (SSH)

A network protocol that allows data to be exchanged using a secure channel between two networked devices. It is used primarily on Linux and UNIX operating systems to access shell accounts.

Secure Sockets Layer (SSL)

The primary security protocol on the Internet; it utilizes a public key and private key form of encryption between browsers and servers.

Service-oriented architecture (SOA)

A flexible set of design principles used during the phases of systems development and integration. A deployed SOA-based architecture provides a loosely integrated suite of services that you can use in multiple business domains.

servlet

A Java application that runs in a server, typically a Web or application server, and performs processing on that server. Servlets are the Java equivalent to CGI scripts.

security realm

A security realm comprises mechanisms for protecting Oracle WebLogic Server resources. Each security realm consists of a set of configured security providers, users, groups, security roles, and security policies.

Siebel Enterprise Application Integration (EAI)

The set of products for Siebel Business Applications that includes tools, technologies, and prebuilt functional integrations that facilitate application integration.

Siebel Resource Adapter

The adapter used for deploying SAI for Oracle Fusion Middleware and Siebel Representational State Transfer (RESTful) Services.

Siebel Representational State Transfer (RESTful) Services

A set of Java EE services that makes available a Siebel Repository instance or a Siebel enterprise instance to a connected application or system. It provides a Web service architecture that defines how you use Web standards, such as HTTP and URIs (Uniform Resource Identifier).

See also "Representational State Transfer (REST)."

Simple Object Access Protocol (SOAP)

An XML-based protocol for exchanging information in a decentralized, distributed environment.

Structured Query Language (SQL)

The standard language used to access and process data in a relational database.

Uniform Resource Identifier (URI)

The address syntax that is used to create URLs and XPaths.

Uniform Resource Locator (URL)

The address that defines the location and route to a file on the Internet. URLs are used by browsers to navigate the World Wide Web and consist of a protocol prefix, port number, domain name, directory and subdirectory names, and the file name. For example, http://www.oracle.com/technetwork/index.html specifies the location and path a browser will travel to find the Oracle Technology Network XML site on the World Wide Web.

WAR (Web application archive file)

WAR files encapsulate in a single module all of the components necessary to run an application. WAR files typically contain an application's servlet, JSP, and JSF JSP components.

See also "EAR (enterprise application archive)" and "JAR (Java archive file)."

Web context root

A name that is mapped to the document root of a Web client and is the starting place from which the Oracle WebLogic Server attempts to find servlets or other programs that are to be executed.

World Wide Web

A worldwide hypertext system that uses the Internet and the HTTP protocol.

World Wide Web Consortium (W3C)

An international industry consortium started in 1994 to develop standards for the World Wide Web. It is located at

http://www.w3.org

Web Services Description Language (WSDL)

The standard format for describing a Web service using XML. A WSDL definition describes how to access a Web service and what operations it performs.

XML

An open standard for describing data developed by the World Wide Web Consortium (W3C) using a subset of the SGML syntax and designed for Internet use.

XPath

The open standard syntax for addressing elements within a document used by XSL and XPointer. XPath is currently a W3C recommendation. It specifies the data model and grammar for navigating an XML document utilized by XSLT, XLink and XML Query.

XSD (XML Schema Definition)

An XML-based language used to describe and control XML document contents.

Index