# Oracle® E-Business Suite

Integrated SOA Gateway Implementation Guide

Release 12.2

**Part No. E20925-22**

December 2023

ORACLE

Oracle E-Business Suite Integrated SOA Gateway Implementation Guide, Release 12.2

Part No. E20925-22



Primary Author: Melody Yang

Contributor: Rekha Ayothi, Sudipto Chakraborty, Bhaskar Ghosh, Vardhan Kale, Jackie Lichtenstein, Megha Mathpal, Ravindra Nadakuditi, Aditya Rao, Dilbagh Sardar, Vijay Shanmugam, Vikas Soolapani, Divya Tiwari, Shivdas Tomar, Abhishek Verma, Sarah Zhu

# Contents

## 4 Administering Composite Services - BPEL



## 5 Administering Custom Integration Interfaces and Services



## 6 Securing Web Services



## 7 Logging for Web Services

## 8 Monitoring and Managing Inbound Service Invocation Messages Using Service Monitor



## 9 Implementing SOAP and REST Service Invocation Framework



## 10 Monitoring and Managing Outbound Service Invocation Messages Using Service Invocation Monitor

# Send Us Your Comments

**Oracle E-Business Suite Integrated SOA Gateway Implementation Guide, Release 12.2**

**Part No. E20925-22**

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document. Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

Note: Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the new Oracle E-Business Suite Release Online Documentation CD available on My Oracle Support and www.oracle.com. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: appsdoc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at www.oracle.com.

# Preface

## Intended Audience

Welcome to Release 12.2 of the *Oracle E-Business Suite Integrated SOA Gateway Implementation Guide.*

This guide assumes you have a working knowledge of the following:

- The principles and customary practices of your business area.
- Computer desktop application usage and terminology.
- Oracle E-Business Suite integration interfaces.
- B2B, A2A and BP integrations.

This documentation assumes familiarity with Oracle E-Business Suite. It is written for the technical consultants, implementers and system integration consultants who oversee the functional requirements of these applications and deploy the functionality to their users.

If you have never used Oracle E-Business Suite, we suggest you attend one or more of the Oracle E-Business Suite training classes available through Oracle University.

See Related Information Sources on page x for more Oracle E-Business Suite product information.

## 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.

# Structure



# Related Information Sources

This book is included in the Oracle E-Business Suite Documentation Library. If this guide refers you to other Oracle E-Business Suite documentation, use only the latest Release 12.2 versions of those guides.

**Online Documentation**

All Oracle E-Business Suite documentation is available online (HTML or PDF).

- **Online Help** - Online help patches (HTML) are available on My Oracle Support.

- **Oracle E-Business Suite Documentation Library** - This library, which is included in the Oracle E-Business Suite software distribution, provides PDF documentation as of the time of each release.

- **Oracle E-Business Suite Documentation Web Library** - This library, available on the Oracle Help Center (https://docs.oracle.com/cd/E26401_01/index.htm), provides the latest updates to Oracle E-Business Suite Release 12.2 documentation. Most documents are available in PDF and HTML formats.

- **Release Notes** - For information about changes in this release, including new features, known issues, and other details, see the release notes for the relevant

product, available on My Oracle Support.

- **Oracle Electronic Technical Reference Manual -** The Oracle Electronic Technical Reference Manual (eTRM) contains database diagrams and a detailed description of database tables, forms, reports, and programs for each Oracle E-Business Suite product. This information helps you convert data from your existing applications and integrate Oracle E-Business Suite data with non-Oracle applications, and write custom reports for Oracle E-Business Suite products. The Oracle eTRM is available as an application in Oracle E-Business Suite.

**Related Guides**

You should have the following related books on hand. Depending on the requirements of your particular installation, you may also need additional manuals or guides.

**Oracle Cloud Using the Oracle E-Business Suite Adapter with Oracle Integration 3**

This guide describes how to set up and use Oracle E-Business Suite Adapter connections in Oracle Integration to access supported Oracle E-Business Suite interfaces and REST services as inbound or outbound integrations from Oracle E-Business Suite.

Note that this book is the latest generation of Oracle Integration, Oracle Integration 3. Its prior generation, Oracle Integration Generation 2 is called *Oracle Cloud Using the Oracle E-Business Suite Adapter with Oracle Integration*. Both books are part of the integration documentation in Oracle Cloud Platform as a Service (PaaS) and are available in the Oracle Cloud Library on the Oracle Help Center.

**Oracle E-Business Suite Concepts**

This book is intended for all those planning to deploy Oracle E-Business Suite Release 12.2, or contemplating significant changes to a configuration. After describing the Oracle E-Business Suite architecture and technology stack, it focuses on strategic topics, giving a broad outline of the actions needed to achieve a particular goal, plus any installation and configuration choices that are available.

**Oracle E-Business Suite Electronic Technical Reference Manual User's Guide**

This guide describes how to set up and navigate Oracle E-Business Suite Electronic Technical Reference Manual (eTRM) user interface in Oracle E-Business Suite. It also explains how to browse and search the Oracle eTRM repository to locate desired FND and database metadata and objects, and how to view object details, reports, and diagrams.

**Oracle Application Framework Personalization Guide**

This guide covers the design-time and runtime aspects of personalizing applications built with Oracle Application Framework.

**Oracle E-Business Suite Installation Guide: Using Rapid Install**

This book describes how to run Rapid Install to perform a fresh installation of Oracle E-Business Suite Release 12.2 or to replace selected technology stack executables in an existing instance.

**Oracle E-Business Suite Integrated SOA Gateway User's Guide**

This guide describes the high level service enablement process, explaining how users can browse and view the integration interface definitions and services residing in Oracle Integration Repository.

**Oracle E-Business Suite Integrated SOA Gateway Developer's Guide**

This guide describes how integration developers can perform end-to-end service integration activities. These include orchestrating discrete web services into meaningful end-to-end business processes using business process execution language (BPEL), and deploying BPEL processes at runtime.

This guide also explains how to invoke web services using the Service Invocation Framework. This includes defining web service invocation metadata, invoking web services, and testing web service invocation.

**Oracle E-Business Suite Maintenance Guide**

This guide explains how to patch an Oracle E-Business Suite system, describing the adop patching utility and providing guidelines and tips for performing typical patching operations. It also describes maintenance strategies and tools designed to help keep a system running smoothly.

**Oracle E-Business Suite Mobile Apps Administrator's Guide, Release 12.1 and 12.2**

This guide includes the latest mobile release with new underlying technologies, as well as the earlier mobile releases built with Oracle Mobile Application Framework (MAF). It explains how to set up an Oracle E-Business Suite instance to support connections from Oracle E-Business Suite mobile apps. It also describes common administrative tasks for configuring Oracle E-Business Suite mobile apps. Logging and troubleshooting information is also included in this book.

**Oracle E-Business Suite Mobile Apps Developer's Guide, Release 12.1 and 12.2**

This guide includes information for the latest mobile release with new underlying technologies, as well as the earlier mobile releases built with Oracle Mobile Application Framework (MAF). For mobile releases built with MAF, this guide describes how to develop enterprise-distributed mobile apps by using mobile application archive (MAA) files and how to implement corporate branding. It also explains required tasks on implementing push notifications for supported mobile apps. In addition, it includes how to implement Oracle E-Business Suite REST services to develop custom mobile apps by using the Login component from Oracle E-Business Suite Mobile Foundation or using any mobile app development framework if desired.

**Oracle E-Business Suite Setup Guide**

This guide contains information on system configuration tasks that are carried out either after installation or whenever there is a significant change to the system. The activities described include defining concurrent programs and managers, enabling Oracle Applications Manager features, and setting up printers and online help.

**Oracle E-Business Suite Security Guide**

This guide contains information on a comprehensive range of security-related topics, including access control, user management, function security, data security, secure configuration, and auditing. It also describes how Oracle E-Business Suite can be integrated into a single sign-on environment.

**Oracle Fusion Middleware Oracle E-Business Suite Adapter User's Guide**

This book covers the use of Oracle E-Business Suite Adapter (formerly known as Adapter for Oracle Applications in Oracle Fusion Middleware 11g releases) in developing integrations between Oracle E-Business Suite and trading partners.

This book is available in the Oracle Fusion Middleware 12c Documentation Library and Oracle Fusion Middleware 11g Documentation Library.

**Oracle E-Business Suite User's Guide**

This guide explains how to navigate, enter and query data, and run concurrent requests using the user interface (UI) of Oracle E-Business Suite. It includes information on setting preferences and customizing the UI. In addition, this guide describes accessibility features and keyboard shortcuts for Oracle E-Business Suite.

**Oracle Diagnostics Framework User's Guide**

This manual contains information on implementing and administering diagnostics tests for Oracle E-Business Suite using the Oracle Diagnostics Framework.

**Oracle e-Commerce Gateway Implementation Guide**

This guide describes implementation details, highlighting additional setup steps needed for trading partners, code conversion, and Oracle E-Business Suite. It also provides architecture guidelines for transaction interface files, troubleshooting information, and a description of how to customize EDI transactions.

**Oracle iSetup User's Guide**

This guide describes how to use Oracle iSetup to migrate data between different instances of the Oracle E-Business Suite and generate reports. It also includes information on configuration, instance mapping, and seeded templates used for data migration.

**Oracle Workflow Administrator's Guide**

This guide explains how to complete the setup steps necessary for any product that includes workflow-enabled processes. It also describes how to manage workflow processes and business events using Oracle Applications Manager, how to monitor the progress of runtime workflow processes, and how to administer notifications sent to workflow users.

**Oracle Workflow User's Guide**

This guide describes how users can view and respond to workflow notifications and monitor the progress of their workflow processes.

**Oracle Workflow API Reference**

This guide describes the APIs provided for developers and administrators to access

Oracle Workflow.

**Oracle XML Gateway User's Guide**

This guide describes Oracle XML Gateway functionality and each component of the Oracle XML Gateway architecture, including Message Designer, Oracle XML Gateway Setup, Execution Engine, Message Queues, and Oracle Transport Agent. It also explains how to use Collaboration History that records all business transactions and messages exchanged with trading partners.

The integrations with Oracle Workflow Business Event System, and the Business-to-Business transactions are also addressed in this guide.

### Integration Repository

The Oracle Integration Repository is a compilation of information about the service endpoints exposed by the Oracle E-Business Suite of applications. It provides a complete catalog of Oracle E-Business Suite's business service interfaces. The tool lets users easily discover and deploy the appropriate business service interface for integration with any system, application, or business partner.

The Oracle Integration Repository is shipped as part of the Oracle E-Business Suite. As your instance is patched, the repository is automatically updated with content appropriate for the precise revisions of interfaces in your environment.

## Do Not Use Database Tools to Modify Oracle E-Business Suite Data

Oracle STRONGLY RECOMMENDS that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle E-Business Suite data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Oracle E-Business Suite data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle E-Business Suite tables are interrelated, any change you make using an Oracle E-Business Suite form can update many tables at once. But when you modify Oracle E-Business Suite data using anything other than Oracle E-Business Suite, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle E-Business Suite.

When you use Oracle E-Business Suite to modify your data, Oracle E-Business Suite automatically checks that your changes are valid. Oracle E-Business Suite also keeps track of who changes information. If you enter information into database tables using database tools, you may store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

# 1

# Oracle E-Business Suite Integrated SOA Gateway Overview

## Oracle E-Business Suite Integrated SOA Gateway Overview

Building on top of Oracle Fusion Middleware and service-oriented architecture (SOA) technology, Oracle E-Business Suite Integrated SOA Gateway (ISG) is a complete set of service infrastructure to provide, consume, and administer Oracle E-Business Suite web services.

With service enablement feature, integration interfaces published in the Oracle Integration Repository can be transformed into SOAP and REST based services.

By leveraging Oracle SOA Suite running on Oracle WebLogic Server, Oracle E-Business Suite Integrated SOA Gateway provides greater capabilities and infrastructure for exposing various integration interfaces within Oracle E-Business Suite as SOAP services. SOAP-based services are described in WSDLs and are deployed to Oracle SOA Suite for service consumption.

Unlike SOAP services, REST services, without the dependency on Oracle SOA Suite, are developed with the infrastructure of Oracle E-Business Suite. REST services described in WADLs are directly deployed to an Oracle E-Business Suite WebLogic environment. They can be used for user-driven applications such as Oracle E-Business Suite mobile applications.

Oracle E-Business Suite Integrated SOA Gateway provides Service Invocation Framework to invoke and consume web services provided by other applications.

### Major Features

Oracle E-Business Suite Integrated SOA Gateway can do the following:

- Display all Oracle E-Business Suite integration interface definitions through Oracle Integration Repository

- Support custom integration interfaces from Oracle Integration Repository
- Provide service enablement capability (SOAP and REST services) for seeded and custom integration interfaces within Oracle E-Business Suite
- Use the Integration Repository user interface to perform design-time activities such as generate and deploy Oracle E-Business Suite web services
- Support synchronous and asynchronous (callback without acknowledgement only) interaction patterns for SOAP-based services

  **Note:** In this release, only PL/SQL APIs can be enabled with the support for asynchronous service pattern.

- Support synchronous interaction pattern for REST-based services

  **Note:** In this release, only PL/SQL APIs, Concurrent Programs, Business Service Objects, Java Bean Services, Application Module Services, Open Interface Tables, and Open Interface Views can be exposed as REST services.

- Support multiple authentication types for inbound service requests in securing web service content
- Enforce function security and role-based access control security to allow only authorized users to process administrative functions
- Provide centralized, user-friendly logging configuration for the services generated through the service provider from Oracle E-Business Suite Integrated SOA Gateway
- Audit and monitor Oracle E-Business Suite inbound service operations from Service Monitor
- Audit and monitor outbound service invocations from Oracle E-Business Suite through Service Invocation Monitor
- Leverage Oracle Workflow Business Event System to enable service invocation from Oracle E-Business Suite

## Major Components Features and Definitions

Oracle E-Business Suite Integrated SOA Gateway provides two major service offerings:

- Providing Services

  Oracle E-Business Suite interfaces resided in Oracle Integration Repository can be

service enabled through service provider. The service enablement is the key feature within the Oracle E-Business Suite Integrated SOA Gateway.

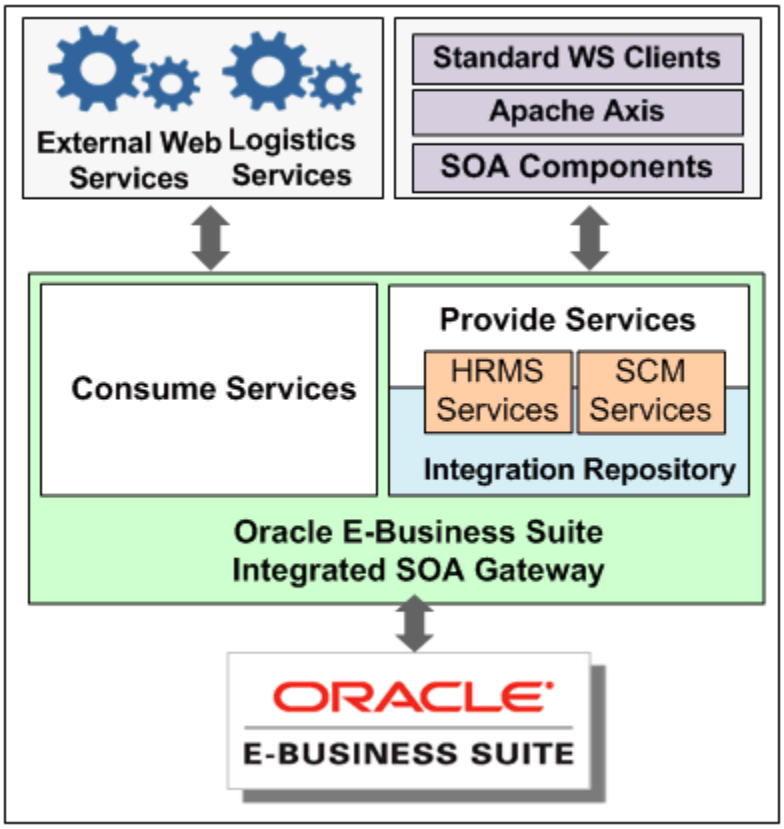
Once services are deployed, web service clients send request messages and then invoke Oracle E-Business Suite services. All these inbound SOAP and REST service invocation messages can be monitored and audited through Service Monitor. See: Monitoring and Managing Inbound Service Invocation Messages Using Service Monitor, page 8-1.

- Consuming Services

  In addition to providing services, Oracle E-Business Suite Integrated SOA Gateway can consume external services through Service Invocation Framework.

  All outbound SOAP and REST service invocations and associated request and response messages through Service Invocation Framework can be monitored and audited using Service Invocation Monitor. See: Monitoring and Managing Outbound Service Invocation Messages Using Service Invocation Monitor, page 10-1.

***Oracle E-Business Suite Integrated SOA Gateway Functional Flow Diagram***




To better understand Oracle E-Business Suite Integrated SOA Gateway, the next

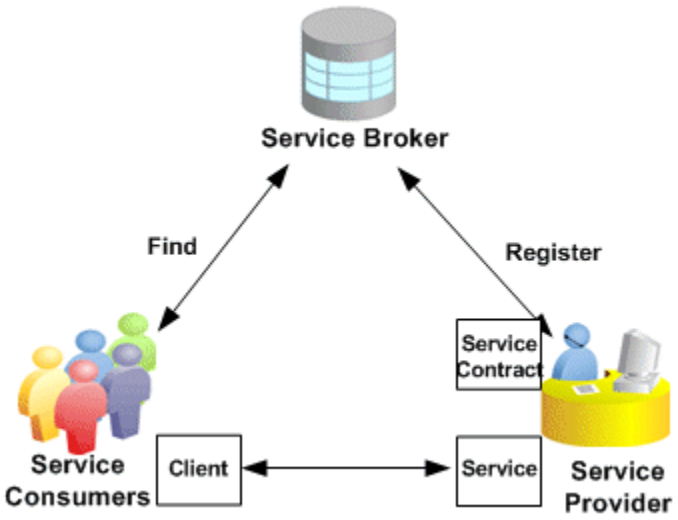
sections explain essential components and how each component is used.

### Enabling Oracle E-Business Suite Web Services

Service enablement is the key feature within Oracle E-Business Suite Integrated SOA Gateway. It provides a mechanism that allows native packaged integration interface definitions resided in Oracle Integration Repository to be transformed into web services. SOAP services are deployed from the Integration Repository to Oracle SOA Suite allowing more consumptions over the web. REST services are deployed to Oracle E-Business Suite.

The basic concept of web service components is illustrated in the following diagram:

***Web Service Components***




- Service Provider is the primary engine underlying the web services. It acts as a bridge between Oracle E-Business Suite and Oracle SOA Suite to facilitate the service enablement for various types of Oracle E-Business Suite interfaces.

  > **Note:** In earlier Oracle E-Business Suite Releases, SOA Provider and Web Service Provider were used in enabling Oracle E-Business Suite services. In the Release 12.2, Service Provider is the engine for service enablement.

  Service Provider leverages Oracle SOA Suite for provisioning Oracle E-Business Suite SOAP-based services. It is the engine that performs the actual service generation and deployment behind the scene.

- Service Consumer (Web Service Client) is the party that uses or consumes the services provided by the Service Provider.

- Service Broker (Service Registry) describes the service's location and contract to ensure service information is available to potential service consumers.

#### Oracle Integration Repository and Service Enablement

Oracle Integration Repository, an integral part of Oracle E-Business Suite, is the centralized repository that contains numerous interface endpoints exposed by applications within the Oracle E-Business Suite. It supports the following interface types:

- PL/SQL
- XML Gateway
- Concurrent Programs
- Business Events
- Open Interface Tables and Open Interface Views
- EDI
- Business Service Object (Service Beans)
- Java

  Apart from normal Java APIs, Java interface includes the following subcategories:

  - Application Module Services

    **Note:** Application Module Implementation class is a Java class that provides access to business logic governing the OA Framework-based components and pages. Such Java classes are called Application Module Services and are categorized as a subtype of Java interface.

  - Java Bean Services

    **Note:** Java APIs whose methods use parameters of either simple data types or serializable Java Beans are categorized as Java Bean Services. Such Java APIs can be exposed as REST-based web services.

  - Security Services

**Note:** Security Services are a set of predefined and predeployed REST services from Oracle Application Object Library. These services include Authentication and Authorization services for mobile applications. These services are built on Java; therefore, they are categorized as a subtype of Java interface.

In this release Java APIs for Forms are not serviceable interfaces and cannot be exposed as SOAP services. Refer to My Oracle Support Knowledge Document 966982.1 for the suggested alternatives to the existing Java APIs for Forms interfaces.

- Composite Interfaces

Oracle E-Business Suite Integrated SOA Gateway leverages Oracle Integration Repository to provide the capabilities of service generation and deployment, as well as service life cycle management.

**Note:** Not all the interface types resided in the Integration Repository can be service enabled. The supported interface types for service enablement are XML Gateway, PL/SQL, Concurrent Program, Business Service Object, Application Module Services, Java Bean Services, Open Interface Tables, and Open Interface Views.

As mentioned earlier, security services are pregenerated REST services from Oracle Application Object Library. Therefore, there is no need to enable the security services from the repository as required by other supported interface types.

### Web Service Security

To protect application data from unauthorized access, Oracle E-Business Suite integrated SOA Gateway enforces the security rules through subject authentication and authorization:

- To authenticate users who request Oracle E-Business Suite services, request messages must be checked based on the selected authentication type:
  - The SOAP messages must be authenticated using the UsernameToken or SAML Token based security. The identified authentication information is embedded in the `wsse:security` Web Security headers.
  - The REST messages are authenticated using the HTTP Basic Authentication security (either user name/password or security token) at the HTTP transport level.
- To authorize users on specific services or operations, the access permissions must

be explicitly given to the users through security grants. Multiple organization access control (MOAC) security rule is also implemented for authorizing interface access related to multiple organizations.

Additionally, input message header (such as SOAHeader and RESTHeader) is used to pass the application context needed in invoking Oracle E-Business Suite services as part of the subject authorization.

#### Service Monitor

Service Monitor is a centralized, light-weight service monitoring and management tool.

It fetches data and statistics for each instance of service request and response messages and lets administrators monitor all *inbound* Oracle E-Business Suite service invocations provided through Oacle E-Business Suite Integrated SOA Gateway. Use Service Monitor to view the runtime request and response data received and sent directly to Oracle E-Business Suite for REST services and the runtime messages passed through Oracle SOA Suite for SOAP services.

For more information about Service Monitor, see Monitoring and Managing Inbound Service Invocation Messages Using Service Monitor, page 8-1.

#### Service Invocation Framework

By leveraging Oracle Workflow Java Business Event System (JBES), Service Invocation Framework (SIF) provides the capability of invoking SOAP and REST services from Oracle E-Business Suite.

It provides an infrastructure allowing developers to interact with SOAP and REST services through service endpoint descriptions. For detailed implementation information, see Implementing SOAP and REST Service Invocation Framework, page 9-1.

#### Service Invocation Monitor

Service Invocation Monitor is also a service monitoring and management tool. It tracks all *outbound* service invocations from Oracle E-Business Suite through Service Invocation Framework.

Administrators use Service Invocation Monitor to view each instance of the runtime request and response messages sent and received through Service Invocation Framework. For more information about Service Invocation Monitor, see Monitoring and Managing Outbound Service Invocation Messages Using Service Invocation Monitor, page 10-1.

## Native Service Enablement Architecture Overview

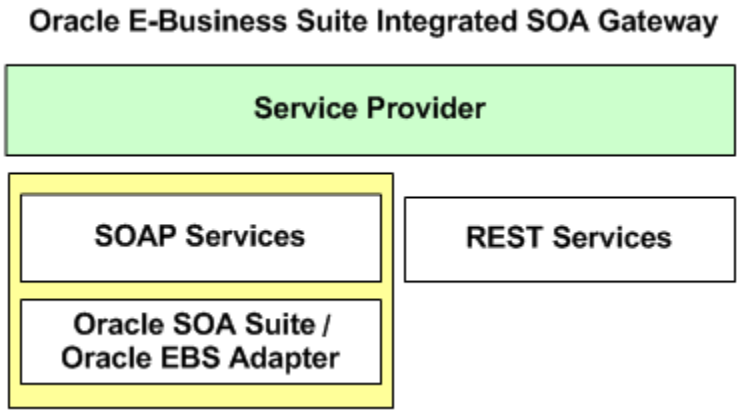
Oracle E-Business Suite Integrated SOA Gateway employs essential components that enable service integration at design time and runtime, and ease the service management throughout the entire service deployment life cycle.

Service Provider is the primary engine enabling the Oracle E-Business Suite services. It is the engine that performs the actual service generation and deployment behind the scene for both SOAP and REST services.

- In SOAP-based service enablement, it leverages Oracle SOA Suite and Oracle Applications Adapter (also called Oracle E-Business Suite Adapter) for provisioning standard web services for business integration.

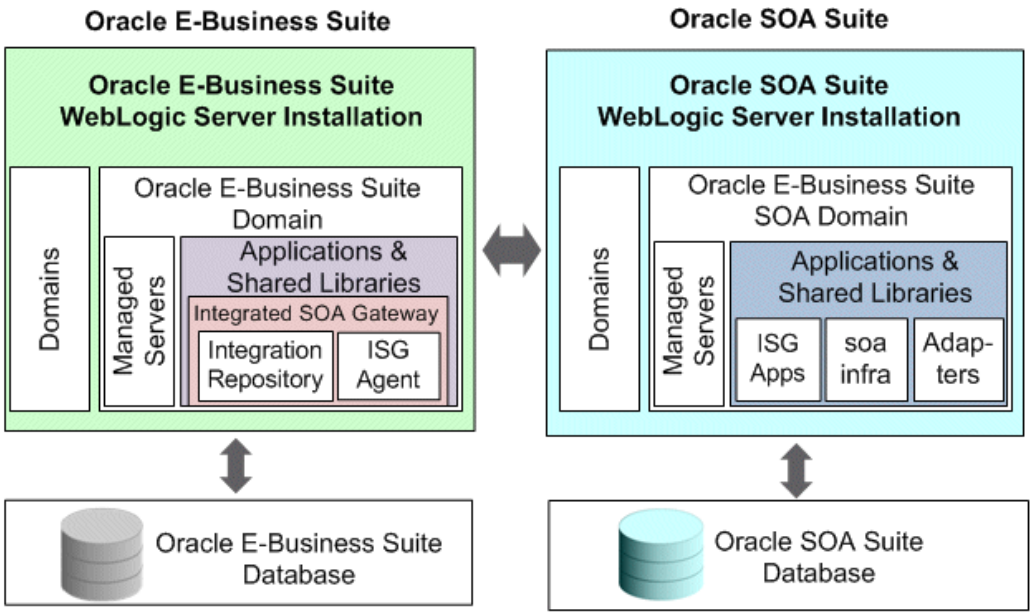
- In REST-based service enablement, it provides light weight, out-of-box services for mobile applications and chatty UI applications.

The high level service enablement diagram can be illustrated in the following diagram:




## SOAP Service Enablement Architecture and Design Time

SOAP services, once successfully generated, are deployed to an Oracle SOA Suite WebLogic environment. The seamless integration between Oracle E-Business Suite and Oracle SOA Suite forms the Oracle E-Business Suite Integrated SOA Gateway architecture.




- **Oracle E-Business Suite on Oracle WebLogic Server**

  Oracle E-Business Suite is integrated with Oracle WebLogic Server (WLS) to provide a complete set of service infrastructure with great flexibility.

  Oracle WebLogic Server is an application server that provides an implementation of Java Platform Enterprise Edition (Java EE, formerly known as J2EE) specification. Its infrastructure enables enterprises to deploy mission-critical applications in a robust, secure, and highly scalable environment and is an ideal foundation for building applications based on service-oriented architecture.

  An Oracle WebLogic Server can include many domains. A domain is an administrative unit or boundary that provides for a single point of administration for a collection of servers. Therefore, a single domain comprises one administration server and one or more managed servers.

  For more information on Oracle WebLogic Server features and system administration, see the *Oracle Fusion Middleware Introduction to Oracle WebLogic Server*.

- **Oracle SOA Suite on Oracle WebLogic Server**

  Oracle SOA Suite is an essential middleware layer of Oracle Fusion Middleware. It contains full range of service components for designing, deploying, and managing composite applications. Furthermore, Oracle SOA Suite provides various integrated capabilities, such as messaging, orchestration, web services management, business monitoring, and so on. These capabilities facilitate the service integration between various enterprises in different platforms.

  With seamless integration with Oracle SOA Suite, Oracle E-Business Suite Integrated SOA Gateway becomes a self-contained web application. Oracle E-

Business Suite integration interfaces can be exposed as web services through SOA Composites in Oracle SOA Suite.

At design time, an integration developer or integration administrator can select a desired interface and perform the service generation from the repository.

Once the service artifact has been generated, an integration administrator can deploy the service from Oracle Integration Repository to an Oracle SOA Suite WebLogic environment where the `soa-infra` application is running.

> **Note:** Users with different roles can perform various tasks in Oracle E-Business Suite Integrated SOA Gateway. Each user role representing a unique permission or permission set can be granted to appropriate users. For example, an integration administrator defined by the Integration Administrator role can perform design-time operations, and other administrative tasks. For information on user roles and how to grant roles to users, see Assigning User Roles, page 2-2 and Role-Based Access Control (RBAC) Security for Oracle E-Business Suite Integrated SOA Gateway, page 6-4.

## REST Service Design Time

Without the dependency on Oracle SOA Suite, REST services are developed based on Oracle E-Business Suite technology infrastructure.

At design time, an integration administrator can select desired methods to be exposed as REST service operations before deploying them to Oracle E-Business Suite.

Additionally, the administrator can undeploy the service if needed.

## Service Enablement Runtime

Oracle E-Business Suite integration interfaces can be exposed as web services and interacted with web service clients at runtime.

When service consumers or web service clients send request messages at runtime, before invoking deployed services in the managed servers, all service-related security and policies are enforced. After authenticating the requests, Oracle E-Business Suite web services can be invoked. Service response messages will be sent back to the web service clients if needed.

For each service operation, SOAP request and response messages passed through Oracle SOA Suite and REST messages received directly to Oracle E-Business Suite are captured in Service Monitor where all Oracle E-Business Suite service activities processed at runtime can be monitored.

For more information on how to monitor web service messages in Service Monitor, see Monitoring and Managing Inbound Service Invocation Messages Using Service Monitor, page 8-1.

Additionally, Oracle E-Business Suite Integrated SOA Gateway can consume external services. Each instance of the runtime request and response messages sent and received through service invocation framework can be tracked and monitored in Service Invocation Monitor. For more information about Service Invocation Monitor, see Monitoring and Managing Outbound Service Invocation Messages Using Service Invocation Monitor, page 10-1.

**Web Service Clients**

Customers or third parties can use the following standard web service client technologies or tools to invoke Oracle E-Business Suite services:

- Apache Axis

  Apache Axis is an open source, XML-based web service framework for constructing SOAP processors such as clients, servers, gateways, etc. It consists of a Java and a C++ implementation of the SOAP server, and various utilities and APIs for generating and deploying service applications. It can help create, publish, and consume services.

- .NET Web Service Client

  .NET web service client enables you to create services and call these services from any client application.

- Oracle JDeveloper

  Oracle JDeveloper is used to help create web service clients through Java SOAP APIs.

- Oracle BPEL Process Manager

  Business process execution language (BPEL) is particularly used in orchestrating complex business processes in a SOA composite application.

- Oracle Service Bus (OSB)

  Oracle Service Bus provides enterprise service level mediation. It can be used for simple transactional service to transport and route messages between service consumers and service providers.

# 2

# Setting Up Oracle E-Business Suite Integrated SOA Gateway

## Setup Overview

Oracle E-Business Suite Integrated SOA Gateway can be set up either on an existing installation of Oracle WebLogic Server or on a newly installed Oracle WebLogic Server. Before the installation, you must first understand the product dependencies.

**Product Dependencies**

Oracle E-Business Suite Integrated SOA Gateway depends on the following products to provide its functionality for SOAP-based web services:

> **Important:** REST-based web services are developed with the infrastructure of Oracle E-Business Suite and they do not depend on Oracle SOA Suite and Oracle Applications Adapter.

- **Oracle SOA Suite running on Oracle WebLogic Server**

  In this release, Oracle E-Business Suite Integrated SOA Gateway leverages the features of Oracle SOA Suite to expose public interfaces in Oracle E-Business Suite as web services.

  Service Provider, one of the essential components in Oracle E-Business Suite Integrated SOA Gateway, uses Oracle SOA Suite for provisioning SOAP requests for Oracle E-Business Suite web services. It generates the SOA Composites which are deployed on Oracle SOA Suite server.

- **Oracle E-Business Suite Adapter** (formerly known as Oracle Applications Adapter)

  Oracle E-Business Suite Adapter provided from Oracle SOA Suite is part of the Oracle Fusion Middleware components. Oracle E-Business Suite Integrated SOA Gateway leverages its features for PL/SQL, Concurrent Program, and XML Gateway based Oracle E-Business Suite web services. The invocation of the web

service is handled by Oracle SOA Suite after the parameters in the inbound SOAP headers are validated by Oracle E-Business Suite Adapter.

For information on how to install Oracle SOA Suite 12c, see *Oracle Fusion Middleware Installing and Configuring Oracle SOA Suite and Business Process Management*.

For information on how to install Oracle SOA Suite 11g, see *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*.

For information on how to configure, troubleshoot, or upgrade Oracle E-Business Suite Integrated SOA Gateway from earlier releases, refer to the following documents:

- For information on how to install or upgrade Oracle E-Business Suite Integrated SOA Gateway from earlier releases, and how to perform setup tasks for both SOAP and REST services, see *Installing Oracle E-Business Suite Integrated SOA Gateway, Release 12.2*, My Oracle Support Knowledge Document 1311068.1.

- For troubleshooting information on potential problem symptoms and corresponding solutions for Oracle E-Business Suite Integrated SOA Gateway, see *Oracle E-Business Suite Integrated SOA Gateway Troubleshooting Guide, Release 12.2*, My Oracle Support Knowledge Document 1317697.1 for details.

After configuring Oracle E-Business Suite Integrated SOA Gateway, administrators should set the required profile options and assign appropriate roles to users which allow them to perform design-time operations, monitor the web services and view logs. The next sections on assigning roles and setting profile options explain these features.

## Assigning User Roles

Oracle E-Business Suite Integrated SOA Gateway uses the following user roles to perform needed administrative and user tasks. Each user role is associated with a specific responsibility by default to access the Integration Repository. A system administrator can assign these user roles to appropriate users if necessary.

- Integration Analyst role (UMX|FND_SYSTEM_INTEGRATION_ANALYST) - Integration Repository responsibility

- Integration Developer role (UMX|FND_SYSTEM_INTEGRATION_DEVELOPER) - Integrated SOA Gateway responsibility

- Integration Administrator role (UMX|FND_IREP_ADMIN) - Integrated SOA Gateway responsibility

  Please note that the Integration Administrator role is assigned to the user ( `SYSADMIN`) by default.

For example, users who have the Integration Analyst role can access the Integration Repository through the Integration Repository responsibility. They can browse integration interfaces and services as well as view each interface details through the

Integration Repository user interface.

Users who have the Integration Developer role can access the Integration Repository through the Integrated SOA Gateway responsibility. They can view each interface through the repository, and also annotate custom integration interfaces based on annotation standards.

Users who have the Integration Administrator role can perform all user and administrative tasks in the Integration Repository through the Integrated SOA Gateway responsibility. These tasks include browsing and viewing each integration interface and service, generating, deploying, and undeploying services, as well as retiring active services, activating retired services, and resetting services.

**To assign a user role:**

1. Log in to Oracle E-Business Suite as a user who has the User Management responsibility.
2. Select the Users link from the navigation menu.
3. Enter appropriate information in the search area to locate a desired user account. Click **Go**.
4. Click the **Update** icon next to the user with 'Active' account status to open the Update User window.
5. Click **Assign Roles**.
6. In the search window, search for either one of the following user roles:
   - Integration Analyst
   - Integration Developer
   - Integration Administrator

   Choose a desired role and click **Select**.
7. Enter a justification in the Justification filed and click **Apply**.

   You will see a confirmation message indicating you have successfully assigned the role.

For more information on assigning or revoking user roles, see the *Oracle E-Business Suite Security Guide*.

## Setting Profile Options

The following table lists the profile options used in Oracle E-Business Suite Integrated SOA Gateway:

| Profile Option | Description | Required | Default Value |
|---|---|---|---|
| FND: XML Gateway Map Generic Service | Use this profile option to display or hide the generic XML Gateway service information for the selected XML Gateway map.<br>• If it is set to 'Yes', the Generic XML Gateway Service subregion is displayed within the Web Service region in the XML Gateway Map interface details page.<br>• If it is set to 'No', the Generic XML Gateway Service subregion will not be displayed in the XML Gateway Map interface details page. | Yes | Yes<br>**Important:** If you do not start from this release and you have been using generic XML Gateway web service, set the profile option to 'Yes'. This allows the Generic XML Gateway Services subregion to be displayed within the Web Service region. Otherwise, subregion will not be shown and any invocations of generic XML Gateway web services will return a fault message. |

| Profile Option | Description | Required | Default Value |
|---|---|---|---|
| ISG: Generic Service WSDL URL for XMLG | Once a generic XML Gateway web service has been deployed, the deployed service WSDL URL is populated as the profile value and the URL is also displayed in the 'Generic XML Gateway Service' subregion.<br><br>If the generic service is not deployed, the profile value will not be shown and hence no WSDL URL is displayed in the subregion for the selected XML Gateway interface. | Yes | N/A |

Use the *FND: XML Gateway Map Generic Service* profile option to display generic XML Gateway service information contained in the subregion only if your system is upgraded from a previous release and you have been using generic XML Gateway web services.

For information on setting profile options, see User Profiles and Profile Options in Oracle Application Object Library, *Oracle E-Business Suite Setup Guide*.

# 3

# Administering Native Integration Interfaces and Services

## Overview

Various Oracle E-Business Suite application interface definitions shipped with Oracle Integration Repository are referred as native integration interfaces. This chapter describes how to transform these interface definitions into SOAP and REST web services through the user interface, and how to manage service lifecycle activities using a script.

Note that an Oracle E-Business Suite user who has the Integration Administrator role, hereafter referred as an integration administrator or the administrator, can manage each state of the services throughout the service life cycle as well as manage grants for them.



## Administering SOAP Web Services Through Integration Repository

**Interfaces Supported for SOAP Service Enablement**

Oracle E-Business Suite Integrated SOA Gateway supports the following interface types for SOAP-based service enablement:

> **Important:** For interfaces that can be exposed as SOAP services, if the setup tasks for SOAP services are not performed, when viewing these interfaces through the Integration Repository, you may find a message indicating that Oracle E-Business Suite Integrated SOA Gateway is not configured for SOAP services and refer to My Oracle Support

Knowledge Document 1311068.1 for configuration details.

***Integration Repository Page with Information About Configuring Oracle E-Business Suite Integrated SOA Gateway***


- PL/SQL
- XML Gateway Map (Inbound)
- Concurrent Program

  **Important:** Oracle Integration Repository supports REST service enablement for open interface tables and open interface views. If a concurrent program is associated with an open interface table or an open interface view, this concurrent program can be viewed and displayed under the "Open Interface" type and can be available as a REST service.

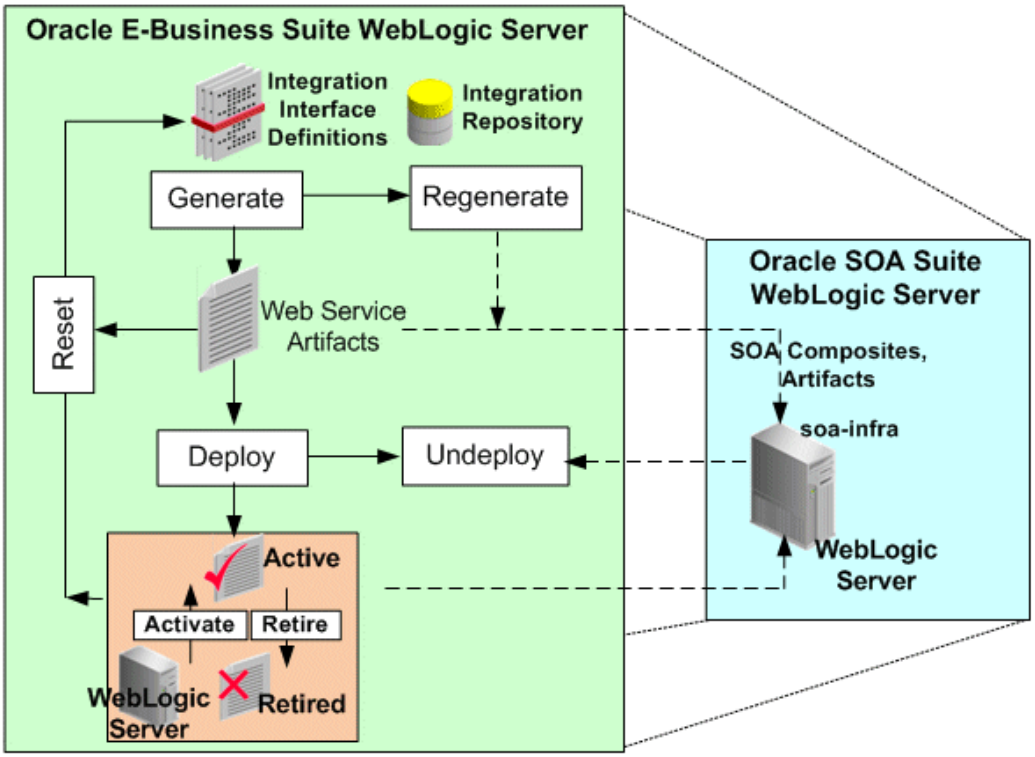
- Business Service Object

In this release Java APIs for Forms are no longer serviceable interfaces and cannot be exposed as SOAP services. If you are planning to use this type of interfaces as web services, you are advised to use alternate serviceable interfaces, such as PL/SQL and Business Service Objects interfaces, which can be deployed as web services. Refer to My Oracle Support Knowledge Document 966982.1 for the suggested alternatives to the existing Java APIs for Forms services.

## Managing SOAP Service Lifecycle Activities

The integration administrators can perform the following administrative tasks in managing each state of SOAP services throughout the entire service life cycle from the Integration Repository user interface:

*Service Generation and Deployment Process Flow*




- Generating SOAP Web Services, page 3-4
- Deploying and Undeploying SOAP Web Services, page 3-11
- Resetting SOAP Web Services, page 3-15
- Retiring SOAP Web Services, page 3-17
- Activating SOAP Web Services, page 3-18
- Subscribing to Business Events, page 3-20
- Managing Security Grants for SOAP Web Services Only, page 3-21
- Enabling Design-Time Log Configuration for SOAP Services, page 3-22
- Viewing Design-Time Logs for SOAP Services, page 3-24

Note that the administrators can also manage SOAP service lifecycle activities and create security grants using an Ant script, see:

- Managing SOAP Service Lifecycle Activities Using an Ant Script, page 3-56

- Managing Security Grants Using an Ant Script, page 3-72

**Managing Other Administrative Tasks for SOAP Services**

Some administrative tasks are performed outside the Integration Repository user interface. These tasks are performed in the **Administration** tab including configuring log setups, and monitoring runtime inbound and outbound SOAP service invocations. See:

- Logging for Web Services, page 7-1
- Monitoring and Managing Inbound Service Invocation Messages Using Service Monitor, page 8-1
- Monitoring and Managing Outbound Service Invocation Messages Using Service Invocation Monitor, page 10-1

## Generating SOAP Web Services

Oracle E-Business Suite Integrated SOA Gateway allows users who have the Integration Administrator role or the Integration Developer role to transform interface definitions to SOAP services.

SOAP services can be generated with the support for synchronous or asynchronous interaction pattern, or both synchronous and asynchronous patterns. Before generating a service, the integration administrator or the integration developer must specify interaction pattern(s) for desired methods to be exposed as service operations. This can be achieved at the method level for one or more methods, or at the interface level for all methods.

> **Important:** In this release, asynchronous operation is supported only in PL/SQL interfaces in enabling SOAP-based services.
>
> - For XML Gateway and Concurrent Program interface types
>
>  Each interface contains only one method and it can only be service enabled synchronously by default; therefore, the Interaction Pattern table is neither displayed in the Web Service region for XML Gateway interfaces nor the SOAP Web Service tab for Concurrent Program interfaces.
>
> - For Business Service Object interface type
>
>  Each interface may contain more than one method; therefore, only the Synchronous column is displayed in the Interaction Pattern table for method selection.

By default, none of the interaction pattern would be selected. However, if your system is upgraded from a previous release, for backward compatibility, 'synchronous' pattern

is selected for all the methods contained in a service.

For more information about synchronous and asynchronous operation patterns, see Synchronous and Asynchronous Web Services, page B-1.

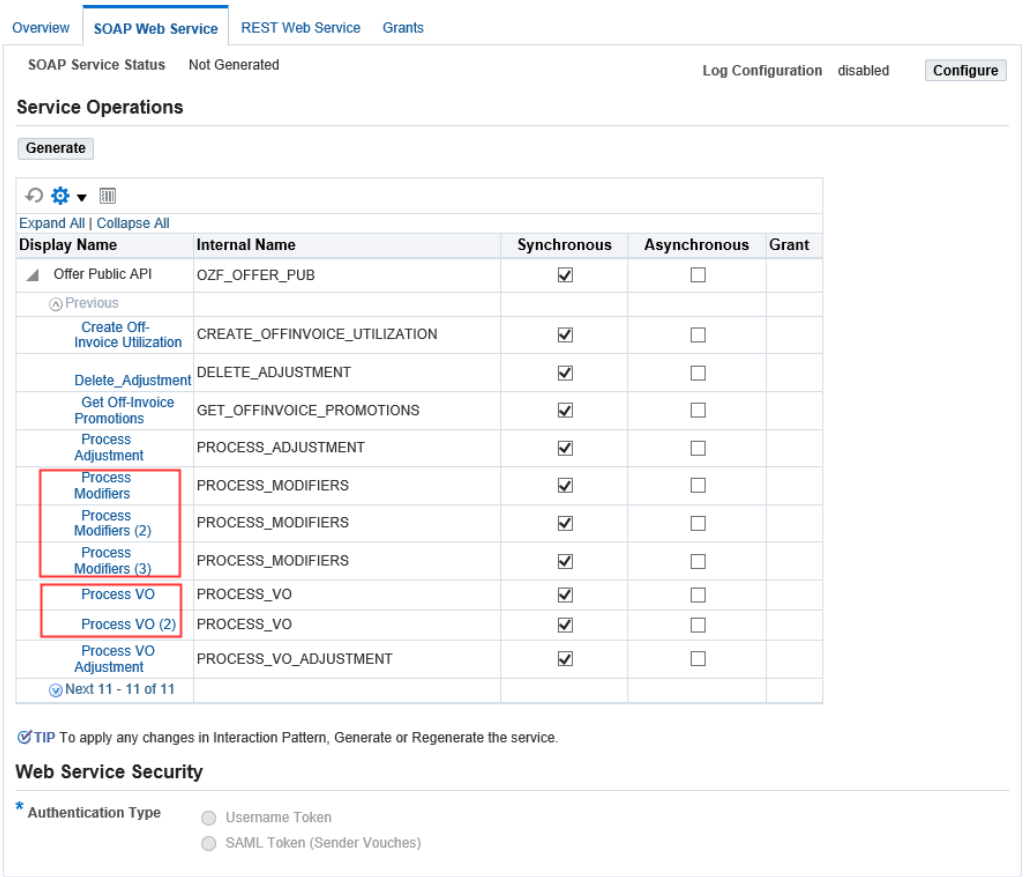
*Generating Services*

For interfaces with the support for SOAP services only, such as XML Gateway maps, service activities are managed in the Web Service region. For interfaces with the support for both REST and SOAP services, such as PL/SQL, Concurrent Programs, and Business Service Objects, these activities are managed in the SOAP Web Service tab of the interface details page.

Once a service is generated, the associated service artifacts are also generated for the selected methods. If only one method is selected, then only that selected method has a service artifact generated.

> **Note:** It's important to note the following for PL/SQL based concurrent program:
>
> - Although at a PL/SQL layer, any concurrent programs can be submitted by FND_REQUEST API, Oracle E-Business Suite Integrated SOA Gateway supports calling of different concurrent programs through separate concurrent program services.
>
> - There may be PL/SQL based APIs exposed through the Integration Repository that are not consistent with the synchronous, auto-committed transaction state of the Web Service Framework in Oracle E-Business Suite Integrated SOA Gateway.
>
> - The WSDL generated by Oracle E-Business Suite Integrated SOA Gateway marks schema elements (parameters) and its related schemas as optional or mandatory, based on the method signature of the underlying API. However, runtime behavior may vary based on API internal implementation.

***Service Generation in the SOAP Web Service Tab with Overloaded Methods Highlighted***




> **Note:** For overloaded functions, sequence number is added to the end of the overloaded method name. Each overloaded function can be uniquely selected and generated with your desired interaction pattern.

### *After Service Generation*

The SOAP Web Service tab or the Web Service region contains the following information:

- **Interaction Pattern Table:** Selected method names with desired interaction patterns are displayed in the table.

  > **Note:** This table is not displayed if the generated service is an XML Gateway map or a concurrent program.

  If changes on the table are required for a generated service:

  - If the generated service has not yet been deployed, after the modification you

must regenerate the service. Upon regeneration, the service definition will be changed to reflect the changes made in the table. You need to modify its web service clients based on the new service definition.

- If the generated service has already been deployed, you must first undeploy the service, modify the pattern selection, regenerate the service, and then deploy the service again.

  For information on service deployment, see Deploying and Undeploying SOAP Web Services, page 3-11.

- **SOAP Service Status (or Web Service Status):** After a service has been generated successfully, the service status is changed from 'Not Generated' to 'Generated'.

  > **Important:** Multiple requests to generate web services for an integration interface are not allowed. If service generation is still in progress, then 'Generating' is displayed as the service status and the **Generate** button is disabled.

- **Interaction Pattern:** 'Synchronous' is displayed by default in the Web Service region or the SOAP Web Service tab if the selected interface is not a PL/SQL API.

- **View WSDL Link:** Click this link to view the generated WSDL description for the selected interface.

  If a method is exposed as a serviceable operation with the support of asynchronous pattern, then `ASYNCH` appears in the WSDL for that method to distinguish it from the rest of the operations generated synchronously. For example, if 'Asynchronous' is selected specifically for the 'CREATE_INVOICE' method within the Invoice Creation API (AR_INVOICE_API_PUB) interface, after service generation, the `ASYNCH` appears in the CREATE_INVOICE operation for both input and output messages as well as binding.

```
...
<portType name="AR_INVOICE_API_PUB_PortType">
   <operation name="CREATE_INVOICE_ASYNCH">
     <input name="tns:CREATE_INVOICE_Input_Msg"/>
   </operation>
</portType>
<portType name="AR_INVOICE_API_PUB_Callback_PortType">
   <operation name="CREATE_INVOICE_ASYNCH_RESPONSE">
     <input name="tns:CREATE_INVOICE_Output_Msg"/>
   </operation>
</portType>
...

<binding name="AR_INVOICE_API_PUB_Binding" type="tns:
AR_INVOICE_API_PUB_PortType">
   <operation name="CREATE_INVOICE_ASYNCH">
     <soap:operation soapAction="CREATE_INVOICE_ASYNCH" />
       <input>
         <soap:header message="tns:CREATE_INVOICE_Input_Msg" part="
header" use="literal" />
         <soap:body use="literal" parts="body" />
       </input>
     </operation>
</binding>
<binding name="AR_INVOICE_API_PUB_CallBack_Binding" type="tns:
AR_INVOICE_API_PUB_CallBack_PortType">
  <soap:binding style="document" transport="http://schemas.xmlsoap.
org/soap/http" />
    <operation name="CREATE_INVOICE_ASYNCH_RESPONSE">
      <soap:operation soapAction="CREATE_INVOICE_ASYNCH_RESPONSE" />

        <input>
         ...
        </input>
    </operation>
 </binding>
```

For more information about WSDL, see: Reviewing SOAP Service WSDL Source, *Oracle E-Business Suite Integrated SOA Gateway User's Guide*.

After service generation, if the interface definition has been changed or the selected interaction pattern information has been modified before service deployment, you can regenerate the service by clicking **Regenerate**. However, if interface definition is not changed, then regenerating the service will not change the service definition.

Click **Reset** to clear up the existing service artifact and change the Web Service Status field from 'Generated' to 'Not Generated'. See: Resetting SOAP Web Services, page 3-15
.

To deploy the generated service, the administrator must select one desired authentication type in the Authentication Type region. The selected authentication type will be used to authenticate Oracle E-Business Suite users at runtime. For more information on deploying a service, see Deploying and Undeploying SOAP Web Services, page 3-11.

**Displaying Generic XML Gateway Service Subregion for Generic XML Gateway Services**

For XML Gateway interface type, if your system is upgraded from a previous release and if you have been using generic XML Gateway web services, the generic XML

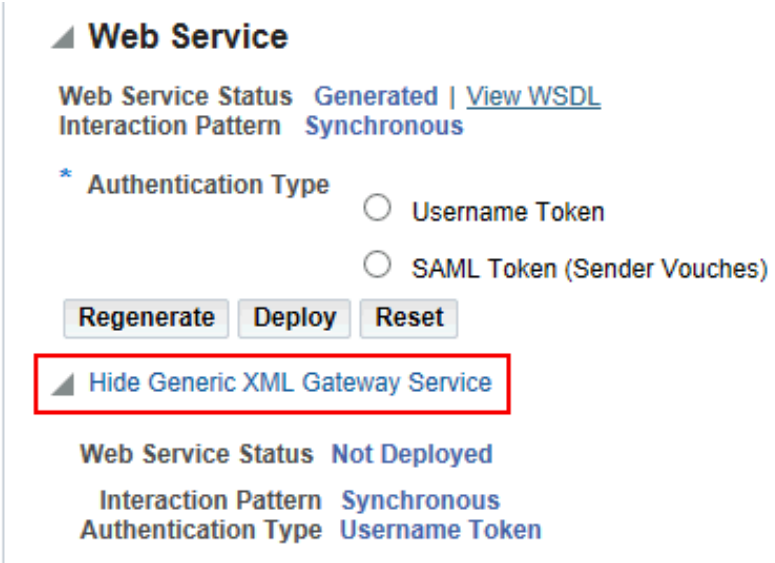
Gateway service information can be displayed by setting the *FND: XML Gateway Map Generic Service* profile value to 'Yes'.

In the Web Service region, click the **Show Generic XML Gateway Service** or **Hide Generic XML Gateway Service** link to display or close the Generic XML Gateway Service subregion for the selected XML Gateway interface.

For more information on setting profile options, see Setting Profile Options, page 2-3.

In addition to setting profile options, the administrator needs to perform additional setup tasks for generic XML Gateway services. For setup information, see *Installing Oracle E-Business Suite Integrated SOA Gateway, Release 12.2*, My Oracle Support Knowledge Document 1311068.1 for details.

***Web Service Region with the Generic XML Gateway Service Subregion Highlighted***




The Generic XML Gateway Service subregion contains the following fields:

- **Web Service Status:** This field indicates the current state of the selected XML Gateway interface.

  If the setup is not configured for the generic XML Gateway service, the Web Service Status field is displayed as 'Not Deployed'.

- **View Generic WSDL:** Click the **View Generic WSDL** link to display the deployed generic WSDL URL for the selected XML Gateway interface.

  The deployed generic WSDL URL has the following syntax:

  ```
  http://<SOA server host>:<SOA Suite managed server port>/soa-infra/services/default/XMLGatewayService!<version chosen while deploying>XMLGateway?WSDL
  ```

  - `<SOA Suite managed server port>`: It is the port of the server where

SOA composite is deployed.

- `<version chosen while deploying>`: At the time of deployment, deployement version will be asked. Default version value is 1.0.

  For example, `http://<SOA server host>:<SOA Suite managed server port>/soa-infra/services/default/XMLGatewayService!1.0/XMLGateway?WSDL`.

After the upgrade to Oracle E-Business Suite Release 12.2, the deployed WSDL URL information has been changed from an earlier release. Therefore, you may have to replace it with the new WSDL URL and service location or address accordingly in web service clients while invoking the generic XML Gateway service.

The updated WSDL URL is also populated in the *ISG: Generic Service WSDL URL for XMLG* profile option by default if the setup tasks for generic XML Gateway services are configured properly.

- **Interaction Pattern:** 'Synchronous' is displayed by default in read-only mode.
- **Authentication Type:** 'Username Token' is displayed by default in read-only mode.

**To generate a web service:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.
2. In the Integration Repository tab, select 'Interface Type' from the View By drop-down list.
3. Expand an interface type node to locate your desired interface definition.
4. Click the interface definition name link to open the interface details page.
5. If this selected interface definition does not have service generated, specify at least one interaction pattern in the Interaction Pattern table. This can be done at the interface level or at the method level before clicking **Generate** in the Web Service region to generate the WSDL description.

   For interfaces that can be supported with both REST and SOAP services, **Generate** is located in the Service Operations region of the SOAP Web Service tab in the interface details page.

   After service generation, the interaction pattern table and the Interaction Pattern field are displayed with selected patterns for your interface.

   The Web Service Status field marked as 'Generated' also appears which indicates that this selected interface has WSDL description available.

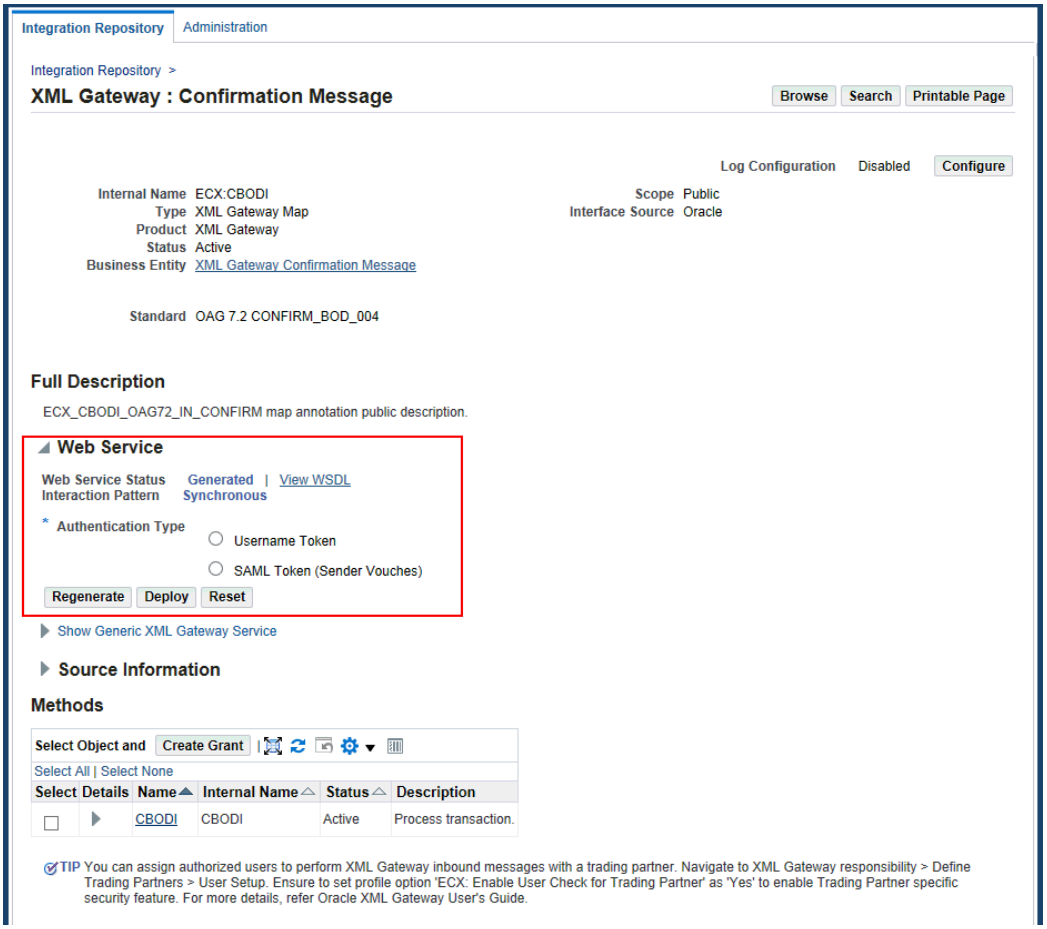
6. Click the **View WSDL** link to view the WSDL description.

7. Click **Regenerate** to regenerate the WSDL description if necessary.

## Deploying and Undeploying SOAP Web Services

If a SOAP service has been generated successfully, the administrator has the privilege to deploy the generated service in the Web Service region or the SOAP Web Service tab if the interface can be exposed as both SOAP and REST services.

***XML Gateway Details Page with Web Service Region Highlighted***




### Deploying Web Services with Authentication Types

Prior to deploying a SOAP web service, the administrator must first select one of the following authentication types:

- Username Token

  This authentication type provides user name and password in the security header

for a web service provider to use in authenticating the users. It is the concept of Oracle E-Business Suite user name/password (or the user name/password created through the Users window in defining an application user).

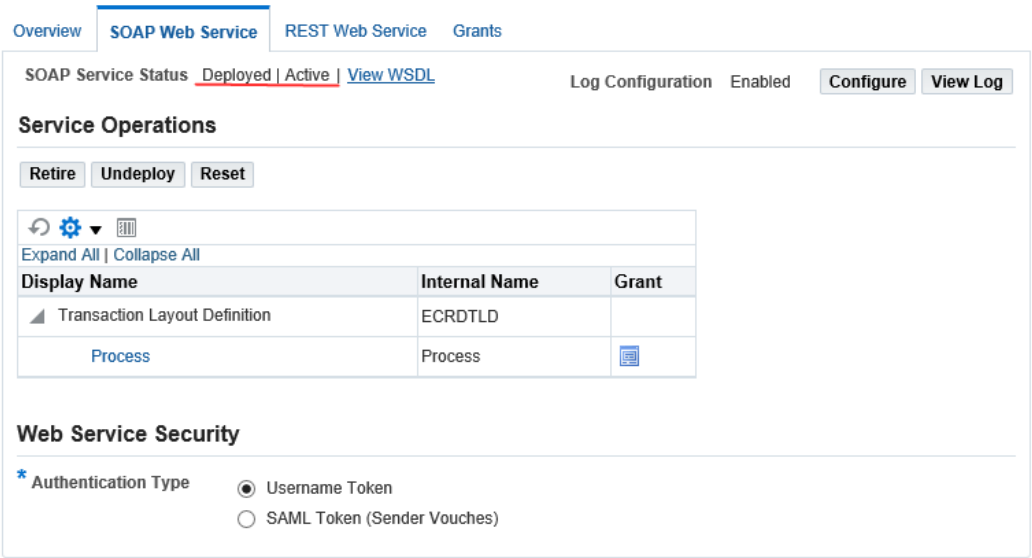
- SAML Token (Sender Vouches)

  This type is used to authenticate web services relying on sending a user name only through SAML Assertion.

### Deployment with Active State

Once a SOAP web service has been successfully deployed, the newly-deployed service has 'Deployed with Active' service status in Oracle SOA Suite where Oracle E-Business Suite services can be used at runtime.

***SOAP Web Service Tab with Deployed and Active Status Highlighted***




The SOAP Web Service tab or the Web Service region for an XML Gateway interface has the following changes:

- The service status is changed from 'Generated' to 'Deployed' with 'Active' state indicating that the deployed service is ready to be invoked and accept new SOAP requests.

- The selected authentication type is displayed.

- Click the **View WSDL** link to display the deployed WSDL information. It shows the physical location of service endpoint where the service is hosted in `soa-infra`.

- The following buttons appear if the service has been successfully deployed with 'Active' state:

- **Retire**: It disables the active service. The service status is changed to 'Deployed' with 'Retired' state indicating that this deployed service will no longer accept new requests. It also ensures that current running requests are finished.

  Once the service has been successfully retired, the **Activate** button appears allowing you to activate the retired service. For more information on retiring and activating web services, see:

  - Retiring SOAP Web Services, page 3-17
  - Activating SOAP Web Services, page 3-18

- **Undeploy**: It undeploys the web service from Oracle SOA Suite back to Oracle Integration Repository. Deployed services can be undeployed with the following reasons:

  - Changes on an interface definition for a deployed service.
  - Changes on interaction pattern for a deployed service.
  - Changes on the Authentication Type field for a deployed service.
  - The original service was corrupt.

  After undeploying the service, make desired changes first (such as interaction pattern or authentication type). Next, regenerate the service, and then deploy the service again.

- **Reset**: It clears up the deployed service artifact and changes the service status from 'Deployed' with 'Active' to 'Not Generated'.

  For more information, see Resetting SOAP Web Services, page 3-15.

For more information on service generation, see Generating SOAP Web Services, page 3-4.

For more information on supported authentication types, see Managing Web Service Security, page 6-8.

**Reviewing Deployed WSDL**

To view the deployed web service, click the **View WSDL** link. The following example shows the deployed WSDL code:

> **Note:** The deployed WSDL shows the physical location of service endpoint where the service is hosted in the `soa-infra` in `<soap:address location>` element. Generated WSDL does not display the physical service endpoint, but with the following information:
>
> ```
> <soap:address location="#NOT_DEPLOYED#" />
> ```

```
<definitions name="ECRDTLD" targetNamespace="http://xmlns.oracle.
com/apps/ec/soaprovider/concurrentprogram/ecrdtld/">
<documentation>
 <abstractWSDL>
   http://<hostname>:<port>/soa-
infra/services/default/<jndi_name>_CONCURRENTPROGRAM_ECRDTLD!
1/ECRDTLD_soap.wsdl
</abstractWSDL>
</documentation>
<types>
   <schema elementFormDefault="qualified" targetNamespace=http://xmlns.
oracle.com/apps/ec/soaprovider/concurrentprogram/ecrdtld/">
  <include schemaLocation="http://<hostname>:<port>/soa-
infra/services/default/<jndi_name>_CONCURRENTPROGRAM_ECRDTLD/ECRDTLD_Ser
vice/?XSD=APPS_ISG_CP_REQUEST_CP_SUBMIT.xsd"/>
   </schema>
   <schema elementFormDefault="qualified" targetNamespace="http://xmlns.
oracle.com/apps/ec/soaprovider/concurrentprogram/ecrdtld/">
  <element name="SOAHeader">
    <complexType>
     <sequence>
      <element name=="Responsibility" minOccurs="0" type="string"/>
      <element name="RespApplication" minOccurs="0" type="string"/>
      <element name="SecurityGroup" minOccurs="0" type="string" />
      <element name="NLSLanguage" minOccurs="0" type="string" />
      <element name="Org_Id" minOccurs="0" type="string" />
     </sequence>
   </complexType>
  </element>
 </schema>
</types>
<message name="ECRDTLD_Input_Msg">
   <part name="header" element="tns1:SOAHeader"/>
   <part name="body" element="tns1:InputParameters"/>
</message>
<message name="ECRDTLD_Output_Msg">
   <part name="body" element="tns1:OutputParameters"/>
</message>
<portType name="ECRDTLD_PortType">
   <operation name="ECRDTLD">
     <input message="tns1:ECRDTLD_Input_Msg"/>
     <output message="tns1:ECRDTLD_Output_Msg"/>
   </operation>
 </portType>
<binding name="ECRDTLD_Binding" type="tns1:ECRDTLD_PortType">
  <soap:binding style="document" transport="http://schemas.xmlsoap.
org/soap/http"/>
  <operation name="ECRDTLD">
   <soap:operation soapAction="ECRDTLD"/>
     <input>
       <soap:header message="tns1:ECRDTLD_Input_Msg" part="header" use="
literal"/>
       <oap:body use="literal" parts="body"/>
     </input>
     <output>
       <soap:body use="literal"/>
     </output>
   </operation>
  </binging>
  <service name="ECRDTLD_Service">
    <port name="ECRDTLD_Port" binding="tns1:ECRDTLD_Binding">
     <soap:address location="http://<hostname>:<port>/soa-
infra/services/default/<jndi_name>_CONCURRENTPROGRAM_ECRDTLD/ECRDTLD_Ser
vice/"/>
   </port>
 </service>
```

```
</definitions>
```

**To deploy or undeploy a web service:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.

2. In the Integration Repository tab, select 'Interface Type' from the View By drop-down list.

3. Expand an interface type node to locate your desired interface definition.

4. Click the interface definition name link to open the interface details page.

5. From the SOAP Web Service tab or the Web Service region of an XML Gateway interface, select one of the following authentication types:
   - Username Token
   - SAML Token (Sender Vouches)

6. Click **Deploy** to deploy the service with active state to an Oracle SOA Suite WebLogic environment.

7. Click the deployed **View WSDL** link to view the deployed WSDL description.

8. Click **Undeploy** to undeploy the service.

9. If a service has been deployed with active state, **Retire** appears letting you disable the active service so that it will no longer accept new requests.

10. Click **Reset** to clear up the existing service artifact.

## Resetting SOAP Web Services

Once an integration interface becomes a web service, the associated service artifact is also generated. No matter if the generated service has been deployed or not, you can clear up the service artifact and *reset* the web service status to its initial state - 'Not Generated' regardless of its current state. This action can be performed at any stage of service generation and deployment life cycle.

For example, an interface definition needs to be modified or has been changed. Instead of regenerating the service if it has not yet been deployed, or undeploying the service if it has been deployed, you can:

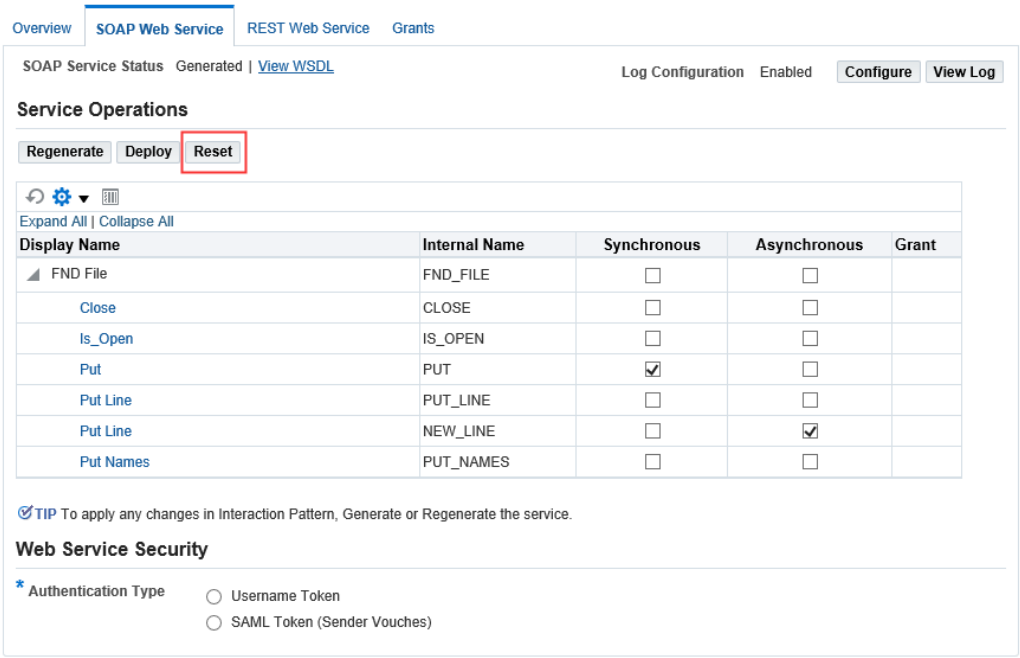
1. Reset the service to clear up the existing service artifact.

2. Modify the interface.

3. Generate the service again.

For information on how to generate a web service for a given interface, see Generating SOAP Web Services, page 3-4.

***SOAP Web Service Tab with Reset Button Highlighted***




**To reset a web service:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.

2. Click **Search** to open the main Search page.

3. Enter appropriate search information such as product family, product, interface type, or business entity.

4. Click **Show More Search Options** and select 'Deployed' or 'Generated' in the Web Service Status field.

5. Locate the interface definition that match your search criteria from the result table.

6. Click the interface definition name link to open the interface details page.

7. In the Web Service region (or the SOAP Web Service tab for the interface with the support for both SOAP and REST services), click **Reset** to clear up the existing service artifact for the selected service. The service status is changed to 'Not

Generated'.

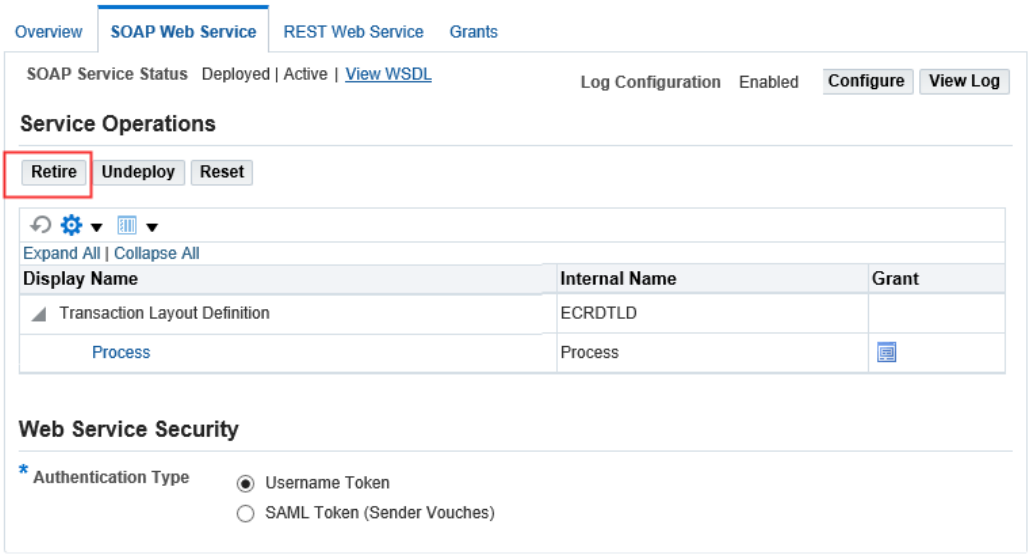
## Retiring SOAP Web Services

When a service has been successfully deployed to Oracle SOA Suite with active state, **Retire** appears allowing you to change the state of the deployed service from 'Active' to 'Retired'.

> **Note:** This action also ensures that current running requests are finished while retiring the service.

Service with 'Retired' state means that the deployed service is no longer active for service invocation and will not accept new SOAP requests.

***SOAP Web Service Tab with Retire Button Highlighted***




Please note that a service with 'Retire' state, the selected interaction pattern and authentication type information remains the same.

After retiring a deployed service, the SOAP Web Service tab for the interface with the support for both SOAP and REST services or the Web Service region has the following changes:

- **Web Service Status:** 'Deployed' with 'Retired' state appears indicating that this deployed service will no longer accept new requests.

- **Activate:** This action lets you change the retired service back to an active service again.

  For information on how to activate a service, see Activating SOAP Web Services,

page 3-18.

- **Undeploy:** This action lets you undeploy the retired service from an Oracle SOA Suite managed server to the repository. See: Deploying and Undeploying SOAP Web Services, page 3-11.

- **Reset:** This action lets you reset the retired service to its initial state - 'Not Generated'. See: Resetting SOAP Web Services, page 3-15.

**To retire a web service:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.

2. Click **Search** to open the main Search page.

3. Enter appropriate search information such as product family, product, interface type, or business entity.

4. Click **Show More Search Options** and select 'Deployed' for the Web Service Status field.

5. Locate the interface definition that match your search criteria from the result table.

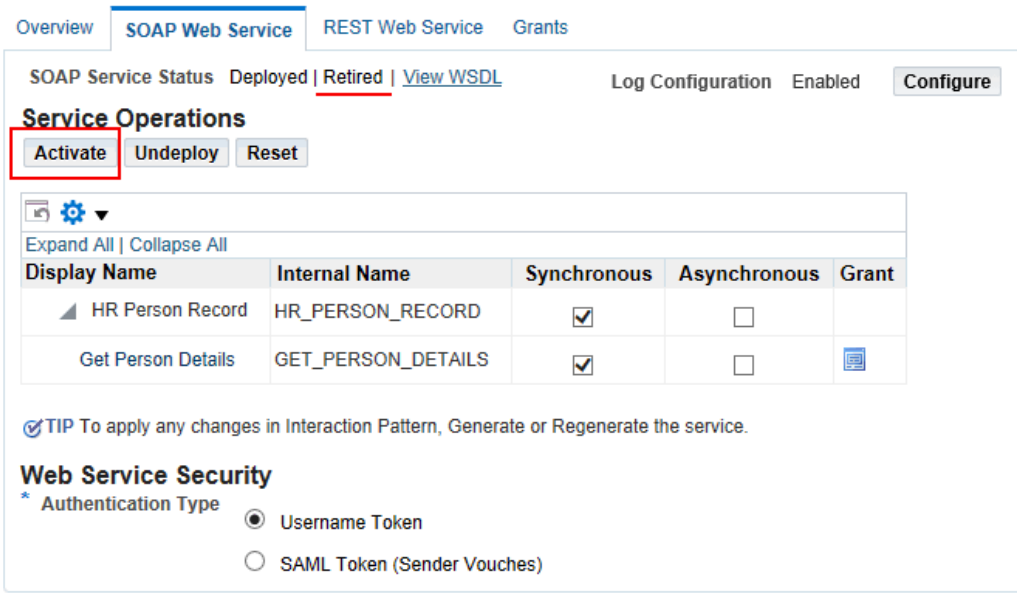
6. Click the interface definition name link to open the interface details page.

7. In the SOAP Web Service tab (or in the Web Service region of an XML Gateway interface), click **Retire** if needed to retire the active deployed service.

## Activating SOAP Web Services

After a service has been deployed with 'Retired' state, it is not available to participate in any web service activities at runtime. To bring it back to work and to be invoked by web service clients, you must change the 'Retired' state to 'Active'. This can be achieved by clicking **Activate** to take the retired service back to an active state again.

*SOAP Web Service Tab with Retired Status and Activate Button Highlighted*




Activating a service will not change its service definition. That is, the selected interaction pattern and authentication type remain the same as they were before.

After activating a service, the following fields are changed in the SOAP Web Service tab of the selected interface (or in the Web Service region of an XML Gateway interface) :

- **Web Service Status:** This field is changed from 'Deployed' with 'Retired' state back to 'Deployed' with 'Active' state. This indicates that the deployed service becomes available again and is ready to be invoked and accept new requests.

- **Retire:** This action lets you retire the activated service again. See: Retiring SOAP Web Services, page 3-17.

- **Undeploy:** This action lets you undeploy the active service from an Oracle SOA Suite managed server to the repository. See: Deploying and Undeploying SOAP Web Services, page 3-11.

- **Reset:** This action cleans up the service artifact and takes it back to its initial state - 'Not Generated'. See: Resetting SOAP Web Services, page 3-15.

**To activate a retired web service:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.

2. Click **Search** to open the main Search page.

3. Enter appropriate search information such as product family, product, interface type, or business entity.

4. Click **Show More Search Options** and select 'Deployed' for the Web Service Status field.

5. Locate the interface definition that match your search criteria from the result table.

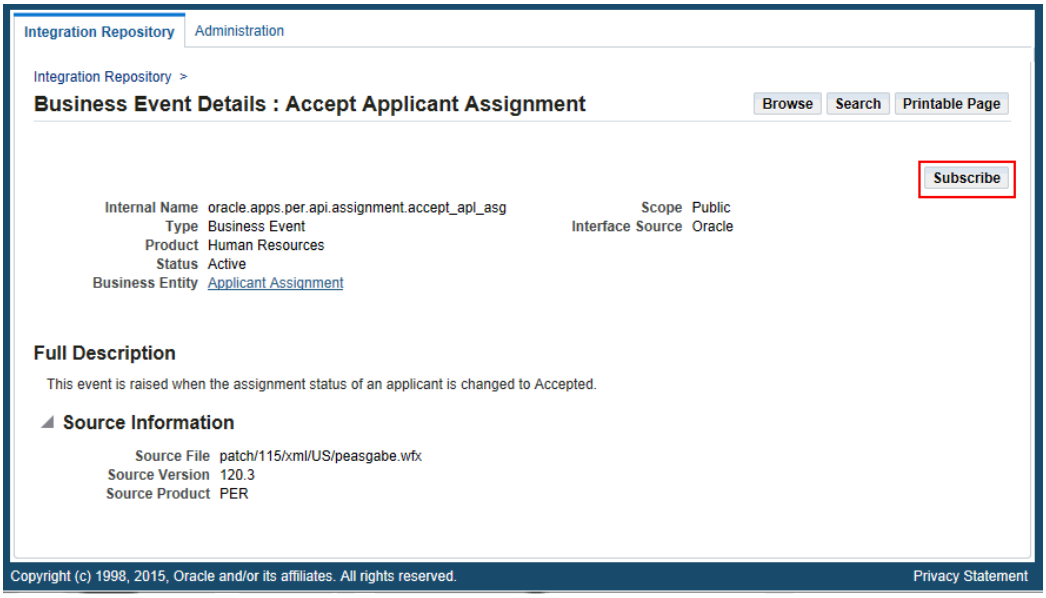
6. Click the interface definition name link to open the interface details page.

7. In the SOAP Web Service tab or the Web Service region, click **Activate** if available to activate the retired service.

## Subscribing to Business Events

An integration administrator can find **Subscribe** in the business event interface details page which allows the administrator to subscribe to a selected business event and create an event subscription for that selected event.

***Business Event Details Page with Subscribe Button Highlighted***




Internally, an event subscription is automatically created for that event with `WF_BPEL_QAGENT` as Out Agent. Once the event subscription has been successfully created, a confirmation message appears on the Business Event interface detail page.

To consume the business event message, you should register to dequeue the event from Advanced Queue `WF_BPEL_Q`. If a business event is enabled and if there is at least one subscriber registered to listen to the `WF_BPEL_Q` queue, then the event message will be enqueued in `WF_EVENT_T` structure to Advanced Queue `WF_BPEL_Q`.

**Unsubscribing to Business Events**

Once an event subscription has been successfully created, **Unsubscribe** appears instead. Clicking **Unsubscribe** removes the event subscription from the `WF_BPEL_Q` queue. A confirmation message also appears after the subscription has been successfully removed.

For more information on how to dequeue messages, see the *Oracle Streams Advanced Queuing User's Guide*.

For more information about business events, see Managing Business Events, *Oracle Workflow Developer's Guide*.

**To subscribe to a business event:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.

2. In the Integration Repository tab, select 'Interface Type' from the View By drop-down list.

3. Expand the Business Event interface type node to locate your desired event.

4. Click the business event interface that you want to subscribe to it to open the Interface details page for the event.

5. Click **Subscribe** to subscribe to the selected event. Internally, an event subscription is created with Out Agent as `WF_BPEL_QAGENT`. A confirmation message appears after the event subscription is successfully created.

   Remove the subscribed event by clicking **Unsubscribe** to remove or delete the event subscription if needed.

## Managing Security Grants for SOAP Web Services Only

To protect application data from unauthorized access, Oracle E-Business Suite Integrated SOA Gateway provides security grant feature allowing only authorized users to access certain methods in an API through Integration Repository.

In this release, XML Gateway (inbound) is the only interface type that can be exposed as SOAP services only. To manage user security for XML Gateway interfaces, you need to log in to Oracle XML Gateway user interface.

> **Note:** For interfaces that can be exposed as REST services, security grants are managed in the Grants tab of the selected interface details page. For example, PL/SQL APIs, Concurrent Programs, and Business Service Objects can be exposed as both SOAP and REST services; Java Bean Services, Application Module Services, Open Interface Tables, and

Open Interface Views can be exposed as REST services only.

Please note that when a method access permission is authorized to a grantee, if the selected method can be exposed as both SOAP and REST service operations, then this grants the permission to the associated SOAP and REST services simultaneously. For information on managing security grants in the Grants tab, see Managing Security Grants for SOAP and REST Web Services, page 3-48.

**Managing XML Gateway User Security in the Trading Partner User Setup Form**

For XML Gateway interfaces, user security is managed in the Oracle XML Gateway user interface through the Trading Partner User Setup form where the administrator needs to associate users with a trading partner. Only these authorized users can perform XML Gateway inbound transactions with the trading partner. Specifically, the administrator needs to:

- Set the "ECX: Enable User Check for Trading Partner" profile option to "Yes" to enable the trading partner specific security feature

- Associate users with a trading partner

  Log in to Oracle E-Business Suite as a user who has the XML Gateway responsibility. Navigate to **Setup** and then select **Define Trading Partners** from the navigation menu. In the Define Trading Partner Setup form, click the **User Setup** button to access the Trading Partner User Setup form.

For more information about trading partner user security, refer to Trading Partner Setup, XML Gateway Setup chapter, *Oracle XML Gateway User's Guide*.

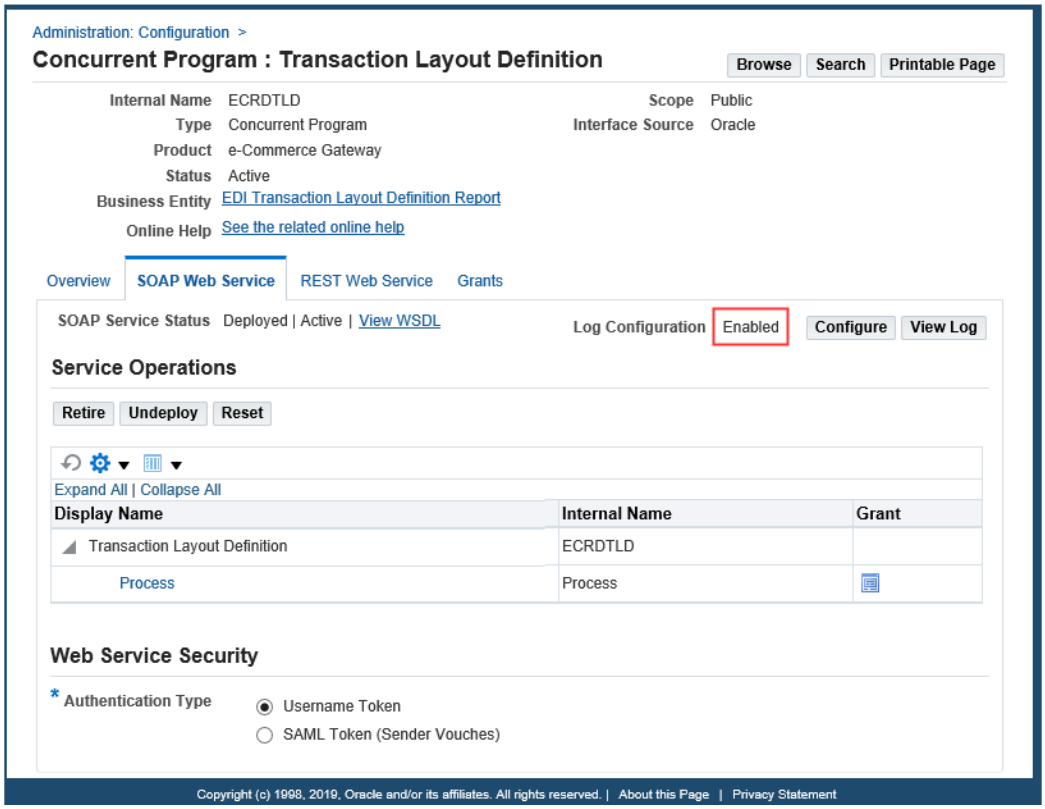
## Enabling Design-Time Log Configuration for SOAP Services

To troubleshoot any issues or exceptions encountered during service generation and deployment life cycle, users who have the Integration Administrator role can enable design-time log for an interface that can be exposed as a SOAP service.

If the design-time log is enabled for an interface with 'SOAP' service type, 'Enabled' is shown as the Log Configuration value in the SOAP Web Service tab. Otherwise, 'Disabled' is displayed instead.

If an interface can be exposed as both SOAP and REST services and that interface has the design-time log enabled for both 'SOAP' and 'REST' service types in two separate configurations, 'Enabled' can be shown in the SOAP Web Service tab and the REST Web Service tab.

*SOAP Web Service Tab with Log Configuration 'Enabled' Highlighted*




### Changing an Existing Log Configuration

To change the design-time log configuration for the selected interface, click **Configure** next to the Log Configuration field in the SOAP Web Service tab. The Log & Audit Setup Details page appears with the selected interface where the administrator can add a new log configuration or update an existing configuration.

> **Note:** The Log & Audit Setup Details page can also be accessed by selecting the **Administration > Configuration** from the navigation menu.

For detailed information about how to configure log settings at the service type level of an interface, see Adding a New Configuration, page 7-6.

### Viewing Design-Time Logs

If the design-time log is enabled for an interface with 'SOAP' service type, **View Log** appears in the SOAP Web Service tab allowing you to view both log messages and error messages if occurred during design-time activities. See Viewing Design-Time Logs for SOAP Services, page 3-24.

## Viewing Design-Time Logs for SOAP Services

To effectively troubleshoot any issues or exceptions encountered at design time during each stage of service generation and deployment life cycle , error messages and activity information can be logged and viewed through the Log & Error Details page.

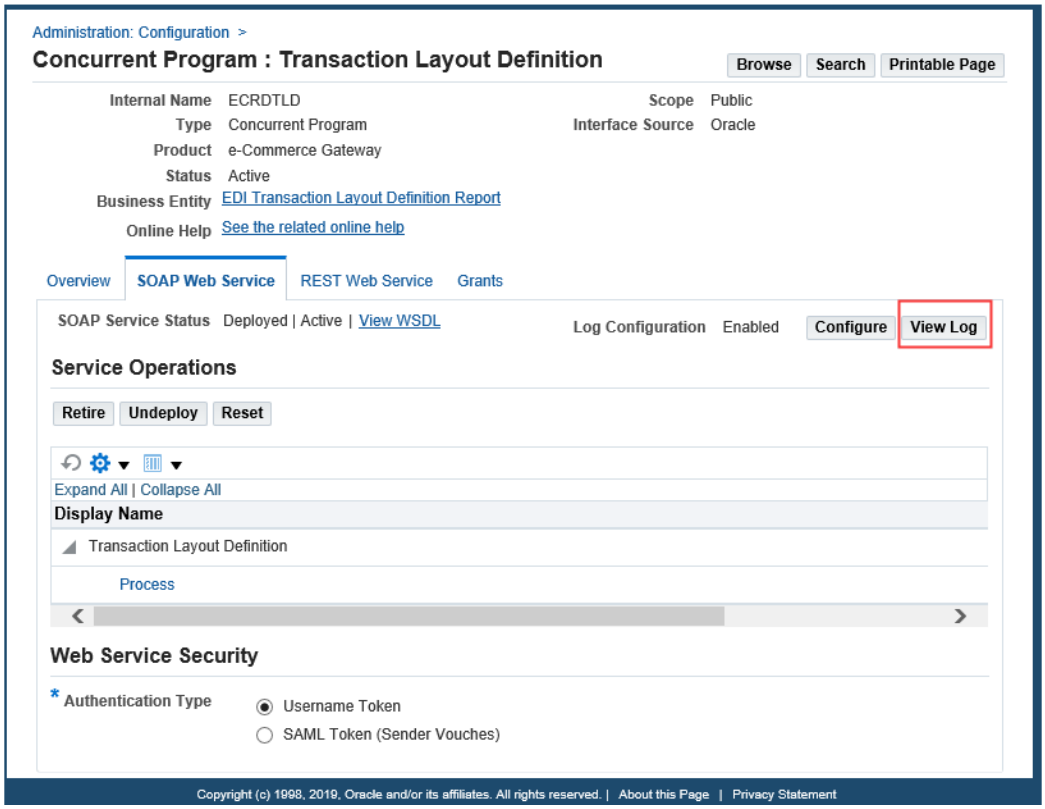
> **Note:** These design-time activities include generating, deploying, retiring, resetting, and activating actions for SOAP services.

- If the design-time log is enabled for 'SOAP' service type of an interface, **View Log** appears in the SOAP Web Service tab for that interface. For XML Gateway interface type that can be exposed as SOAP services only, the **View Log** appears in the header of the interface details page.

  Clicking **View Log** lets you view both log messages and error messages if occurred during design time.

- If the design-time log is not enabled for 'SOAP' service type of an interface and errors occurred while performing the design-time activities for that SOAP service, **View Error** appears instead letting you view the error messages only.

*SOAP Web Service Tab with 'View Log' Highlighted*

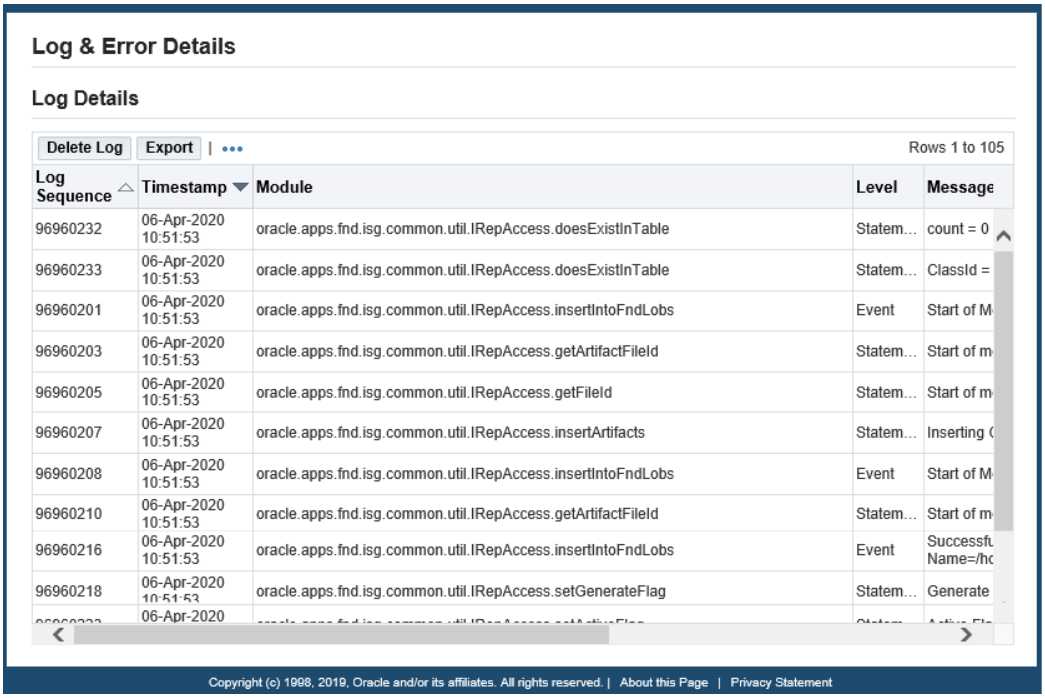



For information on enabling the design-time log for an interface with a desired service type, see Adding a New Configuration, page 7-6.

### Viewing Error and Log Details from the View Log Button

Click **View Log** to display the Log & Error Details page.

*Log & Error Details Page with Log Details Region*




- **Error Details region:** If any errors or exceptions encountered during the design-time activities, error messages are displayed in the Error Details region.

- **Log Details region:** All messages recorded for SOAP service type of the interface are listed in the table. Each log contains log sequence, log timestamp, module, log level, and actual message recorded at the design time.

  **Deleting and Exporting Logs in the Log Details Region**

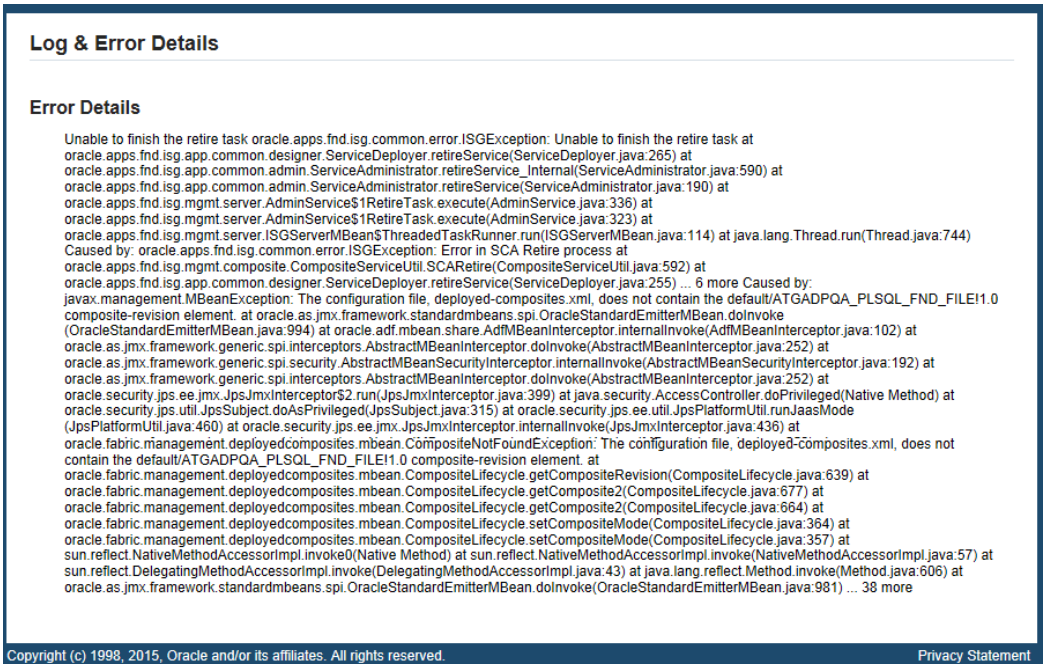
  After viewing the log messages, you can delete them if needed by clicking **Delete Log** in the Log Details region. A warning message appears alerting you that this will permanently delete all the logs retrieved in the region. Click **Yes** to confirm the action. An empty log table appears after logs have been successfully deleted.

  Before deleting the logs, you can save a backup copy by clicking **Export**. This allows you to export the records listed in the Log Details region to Microsoft Excel and use it later.

### Viewing Error Details from the View Error Button

If the selected interface does not have the design-time log enabled for the 'SOAP' service type, and if any errors occurred during the design-time activities for the SOAP service, **View Error** appears instead allowing you to view only the error or exception messages displayed in the Error Details region.

***Log & Error Details Page with Error Details Region***



**Log & Error Details**

**Error Details**

Unable to finish the retire task oracle.apps.fnd.isg.common.error.ISGException: Unable to finish the retire task at
oracle.apps.fnd.isg.app.common.designer.ServiceDeployer.retireService(ServiceDeployer.java:265) at
oracle.apps.fnd.isg.app.common.admin.ServiceAdministrator.retireService_Internal(ServiceAdministrator.java:590) at
oracle.apps.fnd.isg.app.common.admin.ServiceAdministrator.retireService(ServiceAdministrator.java:190) at
oracle.apps.fnd.isg.mgmt.server.AdminService$1RetireTask.execute(AdminService.java:336) at
oracle.apps.fnd.isg.mgmt.server.AdminService$1RetireTask.execute(AdminService.java:323) at
oracle.apps.fnd.isg.mgmt.server.ISGServerMBean$ThreadedTaskRunner.run(ISGServerMBean.java:114) at java.lang.Thread.run(Thread.java:744)
Caused by: oracle.apps.fnd.isg.common.error.ISGException: Error in SCA Retire process at
oracle.apps.fnd.isg.mgmt.composite.CompositeServiceUtil.SCARetire(CompositeServiceUtil.java:592) at
oracle.apps.fnd.isg.app.common.designer.ServiceDeployer.retireService(ServiceDeployer.java:255) ... 6 more Caused by:
javax.management.MBeanException: The configuration file, deployed-composites.xml, does not contain the default/ATGADPQA_PLSQL_FND_FILE!1.0
composite-revision element. at oracle.as.jmx.framework.standardmbeans.spi.OracleStandardEmitterMBean.doInvoke
(OracleStandardEmitterMBean.java:994) at oracle.adf.mbean.share.AdfMBeanInterceptor.internalInvoke(AdfMBeanInterceptor.java:102) at
oracle.as.jmx.framework.generic.spi.interceptors.AbstractMBeanInterceptor.doInvoke(AbstractMBeanInterceptor.java:252) at
oracle.as.jmx.framework.generic.spi.security.AbstractMBeanSecurityInterceptor.internalInvoke(AbstractMBeanSecurityInterceptor.java:192) at
oracle.as.jmx.framework.generic.spi.interceptors.AbstractMBeanInterceptor.doInvoke(AbstractMBeanInterceptor.java:252) at
oracle.security.jps.ee.jmx.JpsJmxInterceptor$2.run(JpsJmxInterceptor.java:399) at java.security.AccessController.doPrivileged(Native Method) at
oracle.security.jps.util.JpsSubject.doAsPrivileged(JpsSubject.java:315) at oracle.security.jps.ee.util.JpsPlatformUtil.runJaasMode
(JpsPlatformUtil.java:460) at oracle.security.jps.ee.jmx.JpsJmxInterceptor.internalInvoke(JpsJmxInterceptor.java:436) at
oracle.fabric.management.deployedcomposites.mbean.CompositeNotFoundException: The configuration file, deployed-composites.xml, does not
contain the default/ATGADPQA_PLSQL_FND_FILE!1.0 composite-revision element. at
oracle.fabric.management.deployedcomposites.mbean.CompositeLifecycle.getCompositeRevision(CompositeLifecycle.java:639) at
oracle.fabric.management.deployedcomposites.mbean.CompositeLifecycle.getComposite2(CompositeLifecycle.java:677) at
oracle.fabric.management.deployedcomposites.mbean.CompositeLifecycle.getComposite2(CompositeLifecycle.java:664) at
oracle.fabric.management.deployedcomposites.mbean.CompositeLifecycle.setCompositeMode(CompositeLifecycle.java:364) at
oracle.fabric.management.deployedcomposites.mbean.CompositeLifecycle.setCompositeMode(CompositeLifecycle.java:357) at
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method) at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57) at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43) at java.lang.reflect.Method.invoke(Method.java:606) at
oracle.as.jmx.framework.standardmbeans.spi.OracleStandardEmitterMBean.doInvoke(OracleStandardEmitterMBean.java:981) ... 38 more

Copyright (c) 1998, 2015, Oracle and/or its affiliates. All rights reserved. Privacy Statement

For example, if the administrator receives errors or exceptions while trying to perform any actions at design time such as Generate, Deploy, Activate, Retire, or Reset for a SOAP service, these errors are recorded and displayed in the Error Details region even if the design-time log for the SOAP service type is not configured for the interface.

Note that the Log Details region will not appear in this page because the design-time log is not configured for the SOAP service type of the selected interface.

- For error messages, error codes, and possible solutions, see Error Messages, page C-1.
- For more logging information, see Logging for Web Services, page 7-1.
- For information on adding a new configuration, see Adding a New Configuration, page 7-6.

At runtime during the service invocation, if a service has the runtime log enabled, log messages can be viewed in Service Monitor against that instance. For information on viewing log messages through Service Monitor, see Viewing Service Processing Logs, page 7-15.

**To view service development log messages:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.

2. In the Integration Repository tab, select 'Interface Type' from the View By drop-down list.

3. Expand an interface type node to locate your desired interface definition.

4. Click the interface definition name link to open the interface details page.

5. If the selected interface does not have the design-time log enabled for the 'SOAP' service type, **View Error** appears instead if errors occurred during the design-time activities.

   Click **View Error** to view the error details that occurred during design time.

6. If the selected interface has the design-time log enabled for a desired service type, **View Log** appears in the SOAP Web Service tab for that interface.

   Click **View Log** to view the log and error details.

   Click **Delete Log** to delete all the logs listed in the table if needed.

   Click **Export** to export log list table to Microsoft Excel and save the records.

## Administering REST Web Services Through Integration Repository

In addition to supporting SOAP-based service generation and deployment, Oracle E-Business Suite Integrated SOA Gateway allows supported interface types to become REST-based services. REST services can be used for user-driven applications such as mobile, tablet, or handheld devices. In this release, PL/SQL APIs, Concurrent Programs, and Business Service Objects can be exposed as both SOAP and REST services; Java Bean Services, Application Module Services, Open Interface Tables, and Open Interface Views can be exposed as REST services only.

> **Note:** Security services are also REST services; however, unlike other service-enabled interfaces, they are predefined and predeployed REST services from Oracle Application Object Library. This type of services provides security related features for mobile applications. See: Supporting Security Services - Predeployed REST Services, page 3-29.

REST services support only *synchronous* (request-response and request-only) interaction pattern and have a simplified service life cycle.

***Simplified Service Life Cycle***

REST services have a simplified service life cycle. The administrator can perform the following tasks in the REST Web Service tab to manage the REST service life cycle:

- Deploy a Service

  A supported interface can be exposed as a REST service through a 'Deploy' action.

Note that REST services are deployed on an Oracle E-Business Suite WebLogic managed server, while SOAP services are deployed on an Oracle SOA Suite WebLogic managed server.

- Undeploy a Service

  The administrator can undeploy a deployed REST service. This action not only undeploys the REST service, but also resets the service to its initial state - 'Not Deployed'. Any existing or running service requests will be completed and no new request is honored.

Note that REST services can be deployed and undeployed using the Ant script `$JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml`. See: Managing REST Service Lifecycle Activities Using an Ant Script, page 3-64.

Additionally, the administrator can manage security grants through the Grants tab of the interface details page and the Ant script. It assigns grants to specific users to access or invoke the deployed REST services. See: Managing Security Grants Using an Ant Script, page 3-72.

***Supporting Security Services - Predeployed REST Services***

In addition to exposing a supported interface as a REST service, Oracle E-Business Suite Integrated SOA Gateway supports Oracle Application Object Library's Authentication and Authorization services as REST security services. Security services are used for mobile applications to validate or invalidate user credentials, initialize user sessions with application context, and authorize users.

Unlike other service-enabled interfaces requiring administrative actions on service development, security services are a set of predeployed REST services which can be invoked by all the Oracle E-Business Suite users.

Security services support token based authentication for invoking other REST services. With token based authentication, it is possible to authenticate a user once based on user name and password, and then authenticate the user in the consecutive REST requests using a security token (such as Oracle E-Business Suite user session ID). For more information about the REST service security, see REST Service Security, page 3-37.

To better understand each administrative task performed through the Integration Repository user interface, this section includes the following topics:


- Deploying REST Web Services, page 3-30
- Undeploying REST Web Services, page 3-46
- Managing Grants for Interfaces with Support for SOAP and REST Services, page 3-48
- Enabling Design-Time Log Configuration for REST Services, page 3-52
- Viewing Design-Time Logs for REST Services, page 3-53

### Managing Other Administrative Tasks for REST Services

Similar to managing SOAP service activities, administrators can perform additional tasks in the **Administration** tab to configure logging, and monitor runtime inbound and outbound REST service invocation messages. See:

- Logging for Web Services, page 7-1
- Monitoring and Managing Inbound Service Invocation Messages Using Service Monitor, page 8-1
- Monitoring and Managing Outbound Service Invocation Messages Using Service Invocation Monitor, page 10-1

## Deploying REST Web Services

Oracle E-Business Suite Integrated SOA Gateway allows the administrator to deploy interface definitions as REST services. These interfaces are PL/SQL APIs, Concurrent Programs, Business Service Objects, Java Bean Services, Application Module Services, Open Interface Tables, and Open Interface Views. Among these interfaces, only PL/SQL APIs, Concurrent Programs, and Business Service Objects can be exposed as both SOAP and REST services.

### Deploying REST Services in the REST Web Service Tab

Before deploying a REST service, the administrator must perform the following tasks:

- **Specify Service Alias**

  Each REST service should be associated with a unique alias name. Alias is a set of characters and is used in the service endpoint which shortens the URL for the service.

  For example, 'Invoice' is entered as the service alias for an interface Create Invoice (AR_INVOICE_API_PUB) before being deployed. The alias will be displayed as the service endpoint in the WADL and schema for a selected service operation CREATE_INVOICE as follows:

  ```
  href="http://<hostname>:<port>/webservices/rest/Invoice/?
  XSD=CREATE_INVOICE_SYNCH_TYPEDEF.xsd" />
  ```

  **Guidelines for Entering Service Alias**

  - Use simple and meaningful name to represent the service, such as "person", "employee", and so on.
  - Do not use "rest", "soap", and "webservices" as the alias.
  - Do not start with a number or a special character, such as #, $, %, _, - and more.
  - Do not end with a special character.

- Characters such as `.`, _, and `-` are allowed in a service alias.

- **Select Desired Methods or Service Operations**

  In the Service Operations table, select one or more methods to be exposed as REST service operations.

  For example, select the CREATE_INVOICE method for the PL/SQL API Create Invoice (AR_INVOICE_API_PUB). After service deployment, only the selected method CREATE_INVOICE will be exposed as a REST service operation.

- **Select Desired HTTP Verbs (PL/SQL APIs, Java Bean Services, Application Module Services, Business Service Objects, Open Interface Tables, and Open Interface Views Only)**

  For PL/SQL APIs, Java Bean Services, Application Module Services, Business Service Objects, Open Interface Tables, and Open Interface Views, in addition to selecting desired methods to be exposed as REST service operations, the administrator needs to select HTTP method checkboxes for the desired methods.

  The following table lists the interfaces that can be exposed as REST services and their supported HTTP methods:

***REST-based Interfaces with Supported HTTP Methods***

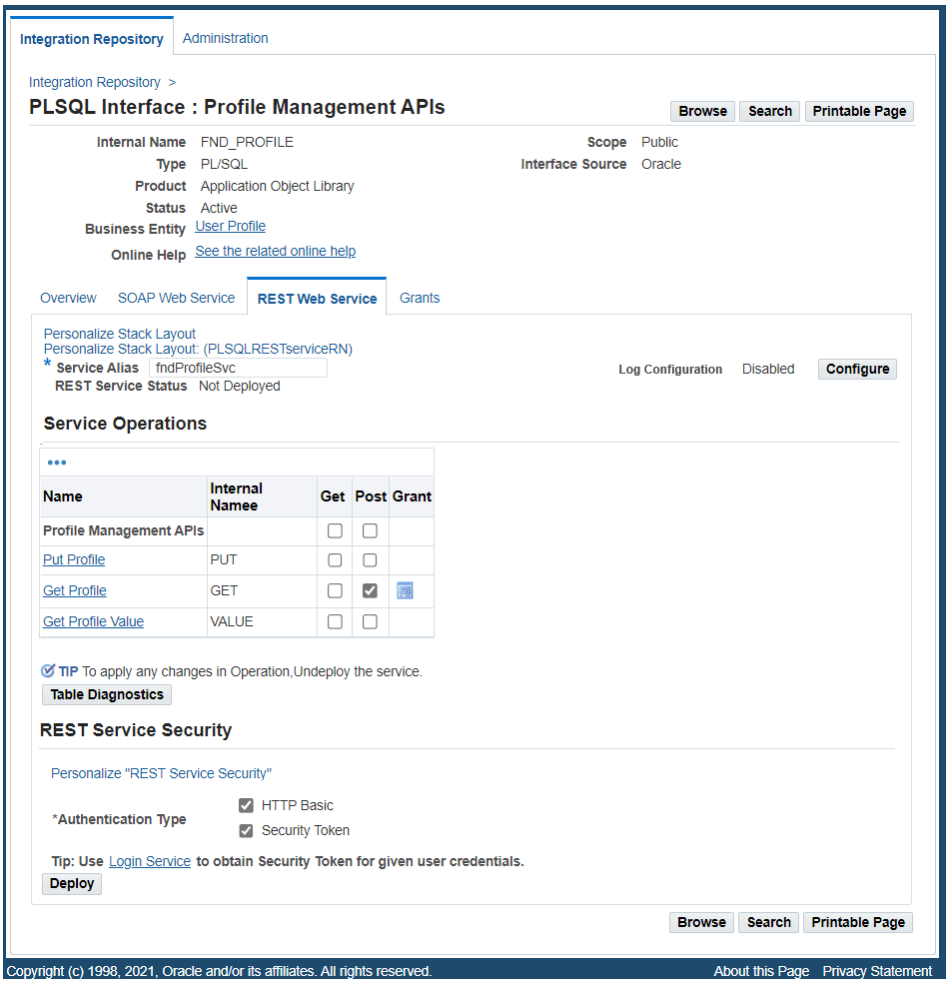
| Interface Type | Supported HTTP Methods |
|---|---|
| Concurrent Program | POST only |
| PL/SQL API | POST and GET |
| Java Bean Service | POST and GET |
| Application Module Service | POST and GET |
| Business Service Objects | POST and GET |
| Open Interface Table (Inbound) | POST, GET, PUT, and DELETE |
| Open Interface Table (Outbound) | GET only |
| Open Interface View | GET only |

**Note:** Concurrent Programs can be exposed as REST services with

the POST HTTP method only; therefore, there is no need to further specify the HTTP method for this interface type.

- *For PL/SQL APIs*

***REST Web Service Tab for PL/SQL API with GET and POST Methods***



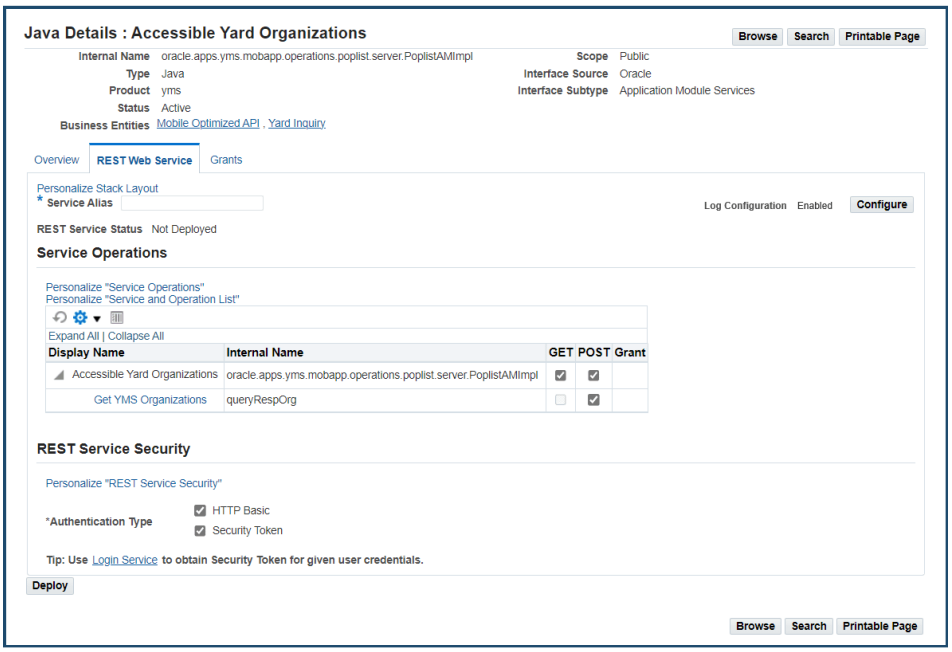

You can deploy a PL/SQL API of non-complex type as a REST service operation with the support of the `GET` and `POST` methods.

- The `GET` checkbox is enabled only if the selected API is a simple data type. The checkbox is disabled if the selected API is a complex data object type.

The administrator can select desired methods for an operation before deploying a PL/SQL API of a simple data type as a REST service.

- *For Java Bean Services and Application Module Services*

***REST Web Service Tab for Application Module Services with GET and POST Methods***




If the Java or Application Module method is annotated (`rep:httpverb`) with a specific HTTP method, then the corresponding HTTP method checkbox is preselected for that method in the table.

- If the `GET` HTTP method is not annotated, then the `GET` checkbox becomes inactive or disabled for further selection. This means that the Java or Application Module method will never be deployed as a REST service operation with the `GET` method.

- If the `POST` HTTP method is not annotated, unlike the `GET` method, the `POST` checkbox is still active or enabled by default. This allows the administrator to select the `POST` checkbox if needed for the Java or Application Module method as a REST service operation before deploying the service.

For example, if the "Get YMS Organizations" method within the "Accessible Yard Organizations" is annotated only with the `POST` HTTP method, then the `POST` method checkbox is preselected for the method. The `GET` method checkbox that is not annotated for the "Get YMS Organizations" method is shown as inactive or disabled which cannot be chosen for that method before deploying the service.

The administrator can modify the desired HTTP methods before deploying the REST service. For example, uncheck the preselected `POST` checkbox if the "Get

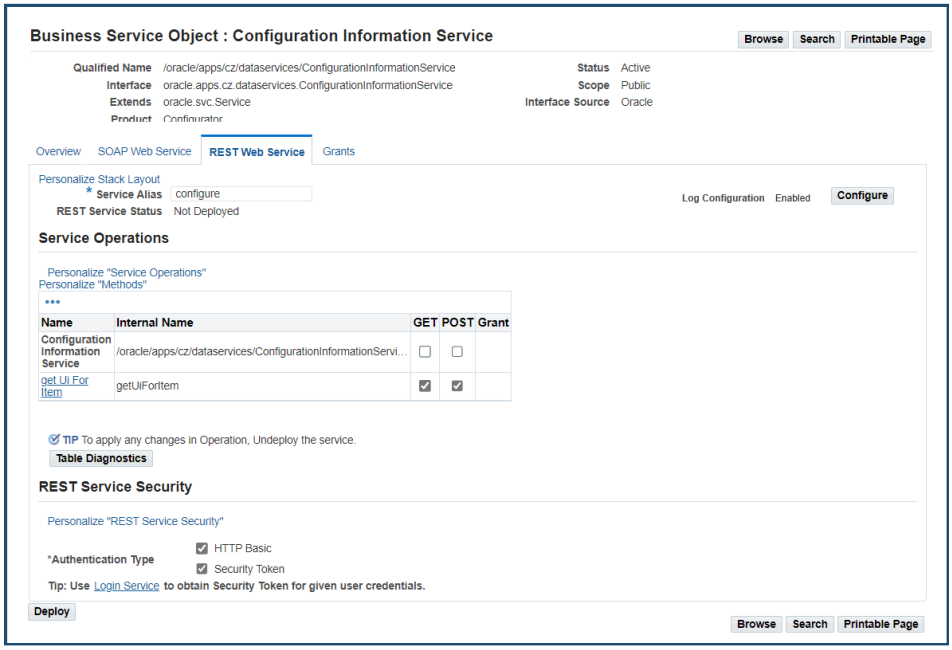
YMS Organizations" method will not be exposed as a REST service operation with the `POST` method.

For information about the `rep:httpverb` annotation, see rep:httpverb, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*. For more Java Bean Services annotation guidelines, see Annotations for Java Bean Services, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

For more Application Module Services annotation guidelines, see Annotations for Application Module Services, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

- ***For Business Service Object Interfaces***

***REST Web Service Tab for Business Service Objects with GET and POST Methods***




A Business Service Object interface can be exposed as a REST service operation with the support of the `GET` and `POST` methods.

- The `GET` checkbox is enabled only if input parameters of the selected interface are of simple data types (String, Number, etc.). The checkbox is disabled if input parameters consist of complex data object types (AccountMergeRequest, etc.).

- There is no annotation required for enabling or using the `GET` and `POST` methods.

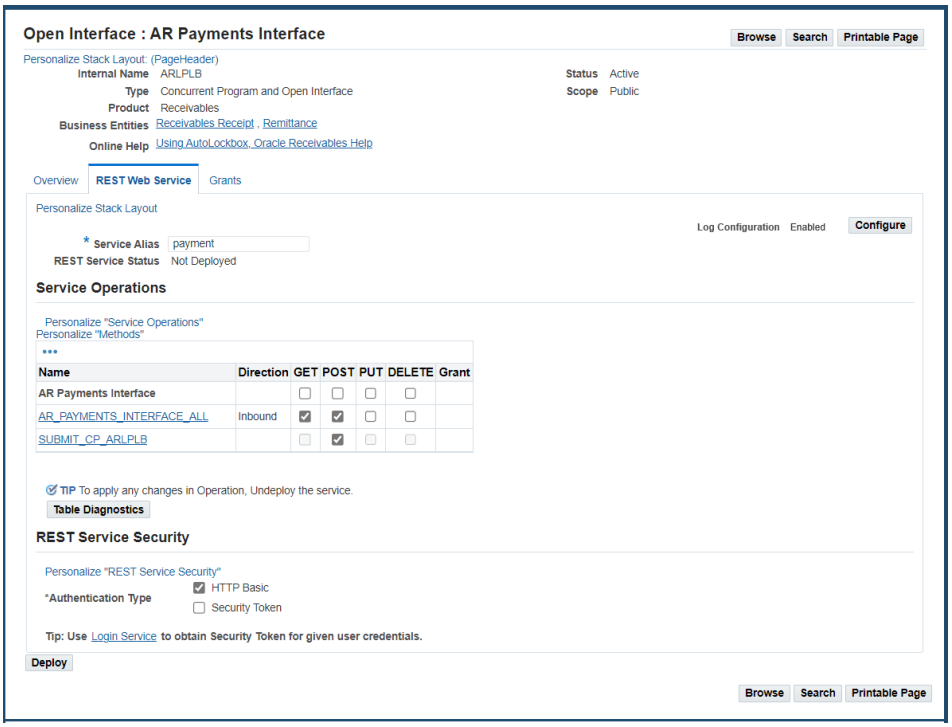
The administrator can select desired methods for an operation before deploying

the Business Service Object interface as a REST service.

- *For Open Interface Tables and Open Interface Views*

  For **open interface tables**, the supported HTTP methods are determined by the direction of the open interfaces.

***Open Interface Table Details Page***




- An open interface table with `Inbound` direction
  - All four HTTP methods (`GET`, `POST`, `PUT`, and `DELETE`) are available for selection.
    - `GET` - It reads or selects one or more records from the open interface table or view.
    - `POST` - It creates or inserts one or more records to the open interface table.
    - `PUT` - It updates or edits one or more records in the open interface table.
    - `DELETE` - It deletes or removes one or more records from the open interface table.

- An additional method called `SUBMIT_CP_<internal name of the associated concurrent program>` appears as the last entry of the method table with the `POST` HTTP method only.

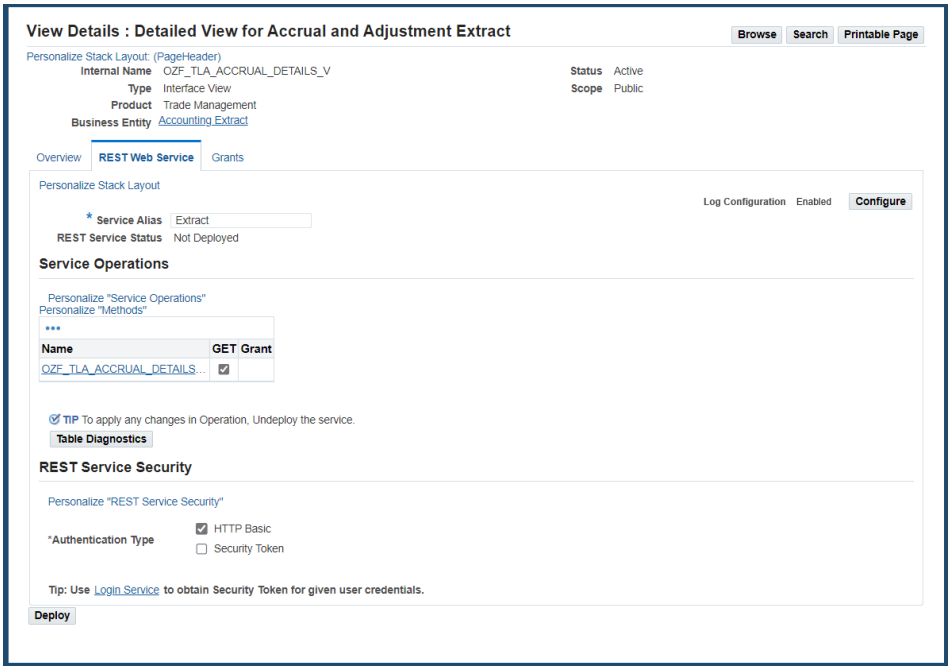
  Please note that open interface is a combination of a concurrent program and associated open interface tables. Therefore, all these components including each open interface table and the concurrent program contained in a selected open interface table should be service enabled if desired. You can submit the associated concurrent program through this SUBMIT_CP `POST` service operation which is internally mapped to the "process" method of the associated concurrent program.

- An open interface table with `Outbound` direction

  For open interface tables with `Outbound` direction, only the `GET` method is supported.

For **open interface views** which are always with `Outbound` direction, only the `GET` method is supported.

***Open Interface View Details Page***




### REST Service Security

All REST services are secured by the HTTP Basic Authentication or Security Token Authentication at the HTTP or HTTPS transport level. Before deploying an interface as a REST service, the administrator must ensure that at least one authentication type is selected for use in authenticating users who invoke the REST services.

> **Note:** By default, both the HTTP Basic and Security Token authentication types are selected. The administrator can update the default selection to deploy a service with only one desired authentication type (HTTP Basic or Security Token) if needed.

- *HTTP Basic Authentication:* This authentication is for an HTTP client application to provide a user name and corresponding password when making a REST request that is typically over HTTPS.

- *Security Token Authentication:* This token-based security method authenticates a user using a security token provided by the server. When a user tries to log on to a server, a token (such as Oracle E-Business Suite session ID) may be sent as `Cookie` in HTTP header. This authentication method can be used in multiple consecutive REST invocations.

  For example, an Oracle E-Business Suite user has been initially authenticated on a given user name and password. After successful login, the security Login service

creates an Oracle E-Business Suite user session and returns the session ID. The session ID that points to the user session will be passed to HTTP headers of all subsequent web service calls for user authentication.

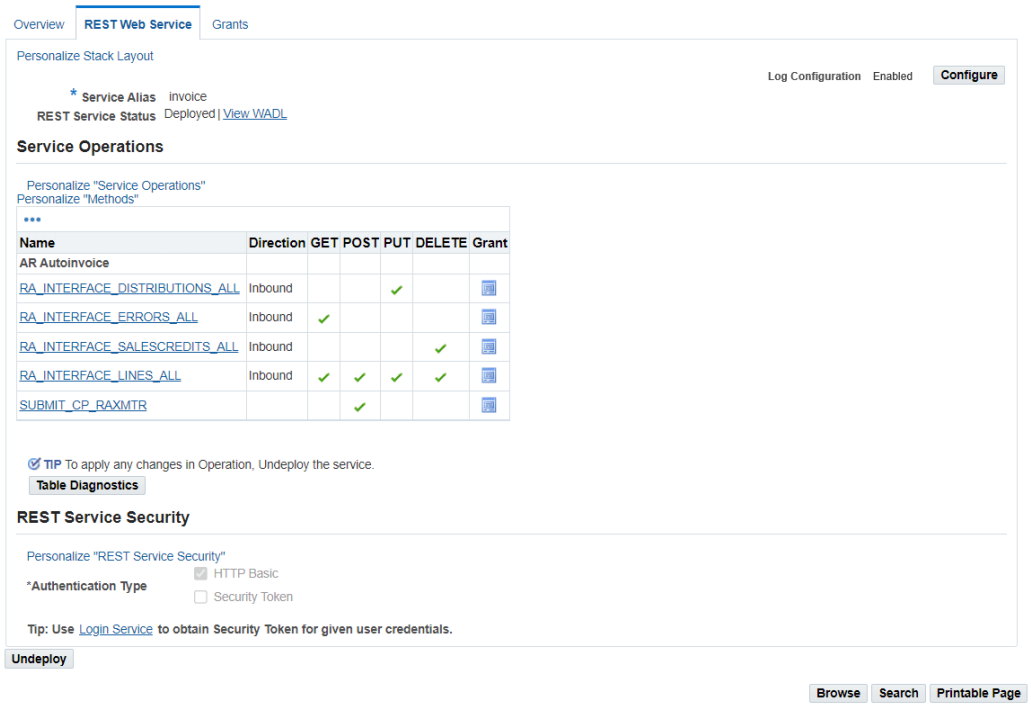
> **Note:** Login service validates the user credentials and returns an access token. It is a predeployed Java security service, and is part of the Authentication services that help validate and invalidate users, as well as initialize applications context required by the service before being invoked.
>
> For more information on applications context in REST service, see REST Header for Applications Context, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.
>
> For more information on supported authentication types, see Managing Web Service Security, page 6-8.

Click **Deploy** to deploy the selected service operations to an Oracle E-Business Suite managed server for consumption.

**After Service Deployment**

Once the REST service has been successfully deployed, the REST Web Service tab has the following changes:

***REST Web Service Tab with 'Deployed' Status***




- **Service Alias**: The REST alias should be displayed as a read-only text field.
- **REST Service Status**: This field is changed from its initial state 'Not Deployed' to 'Deployed' indicating that the deployed service is ready to be invoked and to accept new requests.
- **View WADL**: The **View WADL** link is shown. Click the link to display the deployed WADL information.

  It shows the physical location of the service endpoint where the service is hosted.
- **Verb** (Concurrent Programs Only): This field appears only if the selected interface is a concurrent program.

  'POST' is shown by default in this field as it is the only supported HTTP method for concurrent programs.
- **Service Operations**: This table displays the list of methods (or procedures and functions) contained in the selected interface in read-only mode.
  - A concurrent program contains only one method. If the selected interface is a concurrent program, then the Included Operations column will be checked for the method that has been exposed as a REST service operation with the POST HTTP method.

- If the selected interface is an interface type of PL/SQL, Business Service Object, Java Bean Services, or Application Module Services, then the GET and POST columns will appear with the check marks indicating which HTTP methods have been used to assist the REST service operations.

- If the selected interface is an open interface table with `Inbound` direction, then all four HTTP methods (GET, POST, PUT, and DELETE) will appear with the check marks indicating which HTTP methods have been used to assist the REST service operations.

- If the selected interface is an open interface table with `Outbound` direction or an open interface view, then only the GET column will appear with the check marks for the methods that have been exposed as REST service operations.

- Click the **Grant** icon to view the read-only grant details for a selected method.

- **REST Service Security**: This region displays the selected authentication type (HTTP Basic or Security Token) or both types in read-only mode for the deployed service.

**Reviewing Deployed WADL**

To view the deployed REST service WADL, click the **View WADL** link.

The following example shows the deployed WADL for the selected CREATE_INVOICE service operation contained in the PL/SQL API Invoice Creation ( `AR_INVOICE_API_PUB`):

> **Note:** 'Invoice' highlighted here is the service alias entered earlier prior to the service deployment. After the service is deployed, the specified alias name (Invoice) becomes part of the service endpoint in the .xsd schema file.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<application xmlns:tns="http://xmlns.oracle.
com/apps/ar/soaprovider/plsql/rest/ar_invoice_api_pub/" xmlns="http:
//wadl.dev.java.net/2009/02"
xmlns:tns1="http://xmlns.oracle.com/apps/ar/rest/ar/create_invoice/"
name="AR_INVOICE_API_PUB"
targetNamespace="http://xmlns.oracle.
com/apps/ar/soaprovider/plsql/rest/ar_invoice_api_pub/">
   <grammars>
    <include xmlns="http://www.w3.org/2001/XMLSchema" href="http:
//<hostname>:<port>/webservices/rest/Invoice/?
XSD=CREATE_INVOICE_SYNCH_TYPEDEF.xsd" />
   </grammars>
 <resources base="http://<hostname>:<port>/webservices/rest/Invoice/">
  <resource path="/create_invoice/">
   <method id="CREATE_INVOICE" name="POST">
    <request>
     <representation mediaType="application/xml" type="tns1:
InputParameters" />
     <representation mediaType="application/json" type="tns1:
InputParameters" />
    </request>
    <response>
     <representation mediaType="application/xml" type="tns1:
OutputParameters" />
     <representation mediaType="application/json" type="tns1:
OutputParameters" />
    </response>
   </method>
  </resource>
 </resources>
</application>
```

If the deployed REST service is an interface type of PL/SQL, Business Service Object, Java Bean Services, or Application Module Services, then both GET and POST can be shown as the supported methods. For example, the following WADL description shows two Java methods contained in the Employee Information service. The `getAllReports` operation is implemented with the GET method, and the `getPersonInfo` operation is implemented with both the POST and GET HTTP methods.

```
<xml version="1.0" encoding="UTF-8">
<application name="EmployeeInfo" targetNamespace="http://xmlns.oracle.
com/apps/per/soaprovider/pojo/employeeinfo/"
 xmlns:tns="http://xmlns.oracle.
com/apps/per/soaprovider/pojo/employeeinfo/"
 xmlns="http://wadl.dev.java.net/2009/02" xmlns:xsd="http://www.w3.
org/2001/XMLSchema"
 xmlns:tns1="http://xmlns.oracle.
com/apps/fnd/rest/empinfo/getallreports/"
 xmlns:tns2="http://xmlns.oracle.
com/apps/fnd/rest/empinfo/getdirectreports/"
 xmlns:tns3="http://xmlns.oracle.
com/apps/fnd/rest/empinfo/getpersoninfo/">

<grammars>
   ...
</grammars>
<resources base="http://<hostname>:<port>/webservices/rest/empinfo/">
  <resource path="/getAllReports/">
 <method id="getAllReports" name="GET">
     <request>
     <param name="ctx_responsibility" type="xsd:string" style="query"
required="false" />
           <param name="ctx_respapplication" type="xsd:string" style="
query" required="false" />
    <param name="ctx_securitygroup" type="xsd:string" style="query"
required="false" />
    <param name="ctx_nlslanguage" type="xsd:string" style="query"
required="false" />
    <param name="ctx_orgid" type="xsd:int" style="query" required="
false" />
   </request>
   <response>
     <representation mediaType="application/xml" type="tns1:
getAllReports_Output" />
     <representation mediaType="application/json" type="tns1:
getAllReports_Output" />
    </response>
   </method>
 </resource>
 ...
 <resource path="="/getPersonInfo/ {personId}/">
  <param name="personId" style="template" required="true" type="xsd:int"
/>
 <method id="getPersonInfo" name="GET">
     <request>
     <param name="ctx_responsibility" type="xsd:string" style="query"
required="false" />
           <param name="ctx_respapplication" type="xsd:string" style="
query" required="false" />
    <param name="ctx_securitygroup" type="xsd:string" style="query"
required="false" />
    <param name="ctx_nlslanguage" type="xsd:string" style="query"
required="false" />
    <param name="ctx_orgid" type="xsd:int" style="query" required="
false" />
   </request>
   <response>
     <representation mediaType="application/xml" type="tns3:
getPersonInfo_Output" />
     <representation mediaType="application/json" type="tns3:
getPersonInfo_Output" />
    </response>
   </method>
  </resource>
 <resource path="/getPersonInfo/">
```

```
<method id="getPersonInfo" name="POST">
    <request>
       <representation mediaType="application/xml" type="tns3:
getPersonInfo_Input" />
     <representation mediaType="application/xml" type="tns3:
getPersonInfo_Output" />
    </request>
      <response>
       <representation mediaType="application/xml" type="tns3:
getPersonInfo_Input" />
     <representation mediaType="application/xml" type="tns3:
getPersonInfo_Output" />
           </response>
   </method>
  </resource>
  </resource path>
</application>
```

If the deployed REST service is an open interface table with `Inbound` direction, then the service operation table displays all four HTTP methods. In the following WADL example for the `AR Autoinvoice` open interface table (associated concurrent program internal name `RAXMTR`), the `RA_INTERFACE_LINES_ALL` operation is implemented with all four HTTP methods, and the associated concurrent program `SUBMIT_CP_RAXMTR` is implemented with the POST method.

- Each open interface table name contained in the selected open interface "AR Autoinvoice" is shown in one resource entry ( `<resource path>`) with the selected HTTP methods. For example, table name `RA_INTERFACE_LINES_ALL` in this example is shown with four selected methods (`GET`, `POST`, `PUT`, and `DELETE` ) in one resource entry, and the associated concurrent program `SUBMIT_CP_RAXMTR` with `POST` is contained in another resource entry.

- For the `GET` and `DELETE` methods, application context values, including Responsibility, Responsibility Application, Security Group, NLS Language, and Organization ID complex types, are passed as query strings in the RESTHeader element.

```
 <?xml version = '1.0' encoding = 'UTF-8'?>
<application name="RAXMTR" targetNamespace="http://xmlns.oracle.
com/apps/ar/rest/autoinvoice" xmlns="http://wadl.dev.java.net/2009/02"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:tns1="http://xmlns.
oracle.com/apps/ar/rest/autoinvoice/RA_INTERFACE_LINES_ALL">
 <grammars>
  <include href="http://<hostname>:<port>/webservices/rest/autoinvoice/?
XSD=RA_INTERFACE_LINES_ALL_post.xsd" xmlns="http://www.w3.
org/2001/XMLSchema"/>
  <include href="http://<hostname>:<port>/webservices/rest/autoinvoice/?
XSD=RA_INTERFACE_LINES_ALL_get.xsd" xmlns="http://www.w3.
org/2001/XMLSchema"/>
  <include href="http://<hostname>:<port>/webservices/rest/autoinvoice/?
XSD=RA_INTERFACE_LINES_ALL_put.xsd" xmlns="http://www.w3.
org/2001/XMLSchema"/>
  <include href="http://<hostname>:<port>/webservices/rest/autoinvoice/?
XSD=RA_INTERFACE_LINES_ALL_delete.xsd" xmlns="http://www.w3.
org/2001/XMLSchema"/>
  <include href="http://<hostname>:<port>/webservices/rest/autoinvoice/?
XSD=SUBMIT_CP_RAXMTR_post.xsd" xmlns="http://www.w3.org/2001/XMLSchema"
/>
 </grammars>
  <resources base="http://<hostname>:<port>/webservices/rest/autoinvoice
/"><resource path="RA_INTERFACE_LINES_ALL/">
 <method id="RA_INTERFACE_LINES_ALL_get" name="GET">
     <request>
    <param name="ctx_responsibility" type="xsd:string" style="query"
required="false"/>
           <param name="ctx_respapplication" type="xsd:string" style="
query" required="false" />
    <param name="ctx_securitygroup" type="xsd:string" style="query"
required="false" />
    <param name="ctx_nlslanguage" type="xsd:string" style="query"
required="false" />
    <param name="ctx_orgid" type="xsd:int" style="query" required="
false" />
    <param name="select" type="xsd:string" style="query" required="
false"/>
    <param name="filter" type="xsd:string" style="query" required="
false"/>
    <param name="sort" type="xsd:string" style="query" required="false"
/>
    <param name="offset" type="xsd:string" style="query" required="
false"/>
    <param name="limit" type="xsd:string" style="query" required="false"
/>
   </request>
   <response>
    <representation mediaType="application/xml" type="tns1:
RA_INTERFACE_LINES_ALL_Output"/>
    <representation mediaType="application/json" type="tns1:
RA_INTERFACE_LINES_ALL_Output" />
    <representation mediaType="text/csv" type="tns1:
RA_INTERFACE_LINES_ALL_Output"/>
   </response>
 </method>
 <method id="RA_INTERFACE_LINES_ALL_post" name="POST">
     <request>
    <representation mediaType="application/xml" type="tns1:
RA_INTERFACE_LINES_ALL_Input"/>
    <representation mediaType="application/json" type="tns1:
RA_INTERFACE_LINES_ALL_Input" />
    <representation mediaType="text/csv" type="tns1:
RA_INTERFACE_LINES_ALL_Input"/>
   </request>
   <response>
```

```
<representation mediaType="application/xml" type="tns1:
RA_INTERFACE_LINES_ALL_Output"/>
    <representation mediaType="application/json" type="tns1:
RA_INTERFACE_LINES_ALL_Output" />
    <representation mediaType="text/csv" type="tns1:
RA_INTERFACE_LINES_ALL_Output"/>
   </response>
 </method>
 <method id="RA_INTERFACE_LINES_ALL_put" name="PUT">
     <request>
    <representation mediaType="application/xml" type="tns1:
RA_INTERFACE_LINES_ALL_Input"/>
    <representation mediaType="application/json" type="tns1:
RA_INTERFACE_LINES_ALL_Input" />
    <representation mediaType="text/csv" type="tns1:
RA_INTERFACE_LINES_ALL_Input"/>
   </request>
   <response>
    <representation mediaType="application/xml" type="tns1:
RA_INTERFACE_LINES_ALL_Output"/>
    <representation mediaType="application/json" type="tns1:
RA_INTERFACE_LINES_ALL_Output" />
    <representation mediaType="text/csv" type="tns1:
RA_INTERFACE_LINES_ALL_Output"/>
   </response>
 </method>
 <method id="RA_INTERFACE_LINES_ALL_delete" name="DELETE">
     <request>
    <param name="ctx_responsibility" type="xsd:string" style="query"
required="false"/>
           <param name="ctx_respapplication" type="xsd:string" style="
query" required="false" />
    <param name="ctx_securitygroup" type="xsd:string" style="query"
required="false" />
    <param name="ctx_nlslanguage" type="xsd:string" style="query"
required="false" />
    <param name="ctx_orgid" type="xsd:int" style="query" required="
false" />
    <param name="filter" type="xsd:string" style="query" required="
false"/>
   </request>
   <response>
    <representation mediaType="application/xml" type="tns1:
RA_INTERFACE_LINES_ALL_Output"/>
    <representation mediaType="application/json" type="tns1:
RA_INTERFACE_LINES_ALL_Output" />
    <representation mediaType="text/csv" type="tns1:
RA_INTERFACE_LINES_ALL_Output"/></response>
 </method>
 </resource><resource path="SUBMIT_CP_RAXMTR/">
 <method id="SUBMIT_CP_RAXMTR_post" name="POST">
      <request>
    <representation mediaType="application/xml" type="tns1:
SUBMIT_CP_RAXMTR_Input"/>
    <representation mediaType="application/json" type="tns1:
SUBMIT_CP_RAXMTR_Input" />
    <representation mediaType="text/csv" type="tns1:
SUBMIT_CP_RAXMTR_Input"/>
   </request>
   <response>
    <representation mediaType="application/xml" type="tns1:
SUBMIT_CP_RAXMTR_Output"/>
    <representation mediaType="application/json" type="tns1:
SUBMIT_CP_RAXMTR_Output" />
    <representation mediaType="text/csv" type="tns1:
SUBMIT_CP_RAXMTR_Output"/>
```

```
</response></resource>
 </resources>
</application>
```

For more information about WADL description, see Reviewing WADL Element Details, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

**To deploy a REST service:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.

2. In the Integration Repository tab, select 'Interface Type' from the View By drop-down list.

3. Expand an interface type node to locate your desired interface definition.

4. Click the interface definition name link to open the interface details page.

5. In the REST Web Service tab, enter the following information:

   - Service Alias: Specify service alias information.

   - In the Service Operations table, select one or more methods to be exposed as REST service operations.

     If the selected interface is an interface type of PL/SQL, Business Service Object, Java Bean Services, Application Module Services, Open Interface Tables, or Open Interface Views, select the desired HTTP method checkboxes for the methods to be exposed as REST service operations.

   - In the REST Service Security region, ensure that you select at least one authentication type.

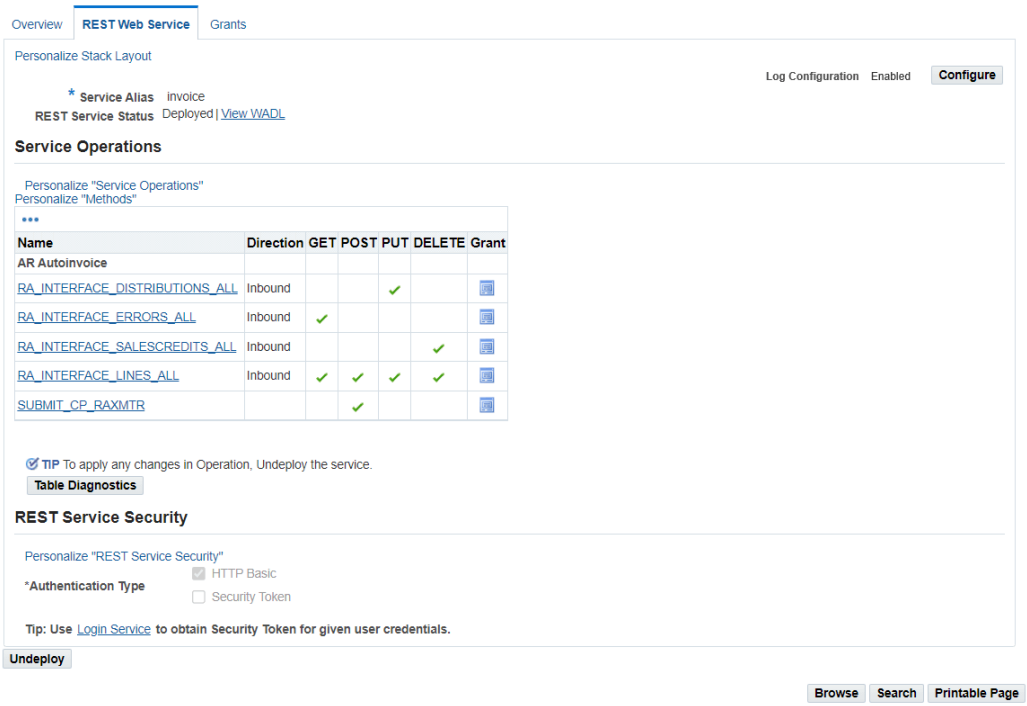
6. Click **Deploy** to deploy the service to an Oracle E-Business Suite environment.

7. Click the deployed **View WADL** link to view the deployed WADL description.

## Undeploying REST Web Services

Once a REST service has been successfully deployed, the **Undeploy** button appears in the REST Web Service tab. This allows the administrator to undeploy the service and at the same time to bring the service back to its initial state - 'Not Deployed'.

***Interface Details Page with a Deployed REST Service***




Please note that when a service is undeployed, any existing or running service requests will be completed and no new request is honored. The associated service artifact will be removed from the system.

After a successful undeployment, 'Not Deployed' is shown in the REST Service Status field. The value of the service alias entered earlier now disappears which allows the administrator to enter it again before next deployment.

**To undeploy a REST service:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.

2. In the Integration Repository tab, select 'Interface Type' from the View By drop-down list.

3. Expand an interface type node to locate your desired interface definition.

4. Click the interface definition name link to open the interface details page.

5. In the REST Web Service tab, click **Undeploy** to undeploy the service.

## Managing Grants for Interfaces with Support for SOAP and REST Web Services

Users who have the Integration Administrator role can create grants to a specific user, users, or a group of users. Grants given to a user for specific services or operations are applicable for both SOAP and REST services.

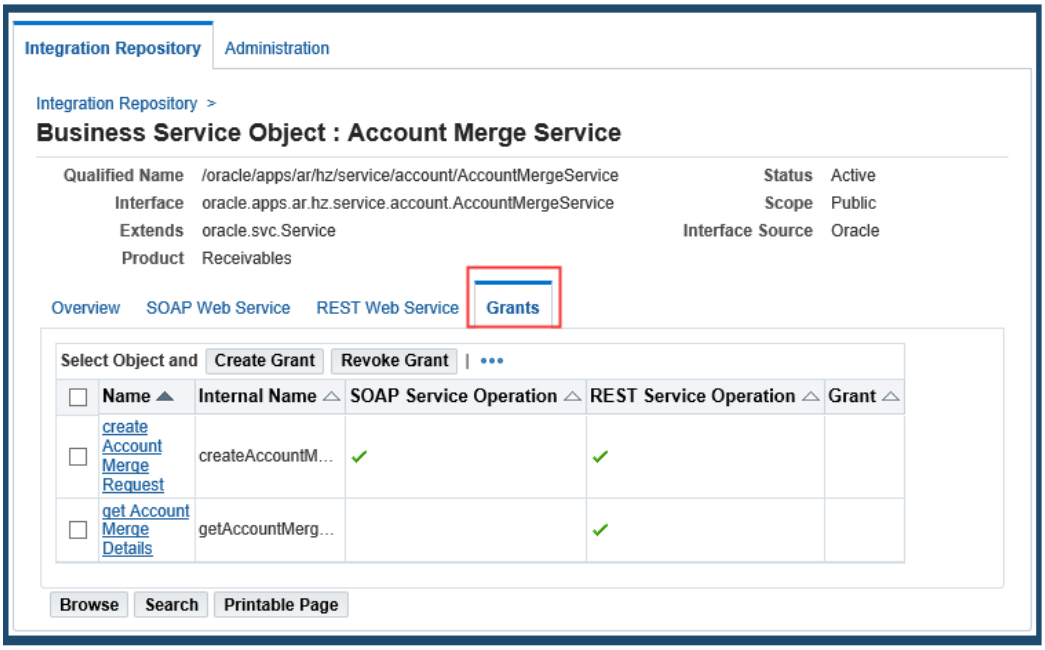
> **Note:** In this release, only PL/SQL APIs, Concurrent Programs, and Business Service Objects can be exposed as both SOAP and REST services. Java Bean Services, Application Module Services, Open Interface Tables, and Open Interface Views can be exposed as REST services only.

### Managing Grants in the Grants Tab for PL/SQL APIs, Concurrent Programs, Business Service Objects, Java Bean Services, Application Module Services, Open Interface Tables, and Open Interface Views

With the exception of XML Gateway interfaces that the user security is managed in the XML Gateway user interface, security grants for all other interface types that can be exposed as web services are managed in the Grants tab of the interface details page. These interfaces are PL/SQL APIs, Concurrent Programs, Business Service Objects, Java Bean Services, Application Module Services, Open Interface Tables, and Open Interface Views.

For information on managing the user security for XML Gateway interfaces, see: Managing Security Grants for SOAP Web Services Only, page 3-21.

***Interface Details Page with Grants Tab Highlighted***




### *Creating Security Grants*

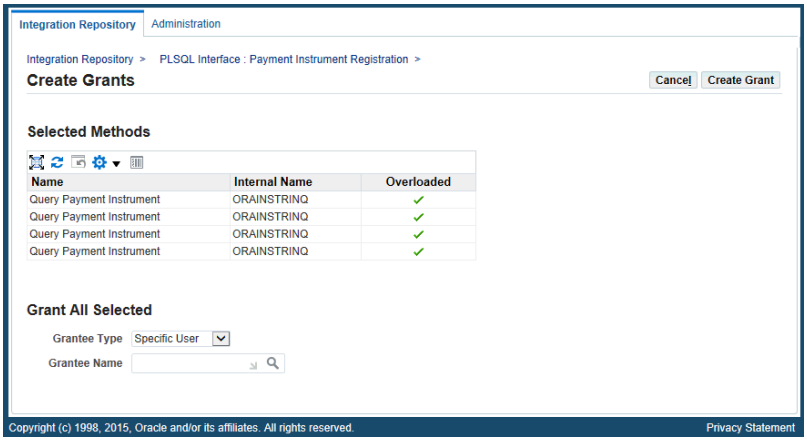
The administrator can select one or more procedures and functions or methods contained in the selected interface, and then click **Create Grant**. The Create Grants page is displayed where the administrator can grant the selected method access permissions to a user, user group, or all users.

Once a method access permission is authorized to a grantee, it grants the permission to access the associated SOAP and REST service operations simultaneously. For example, when a user (OPERATIONS) is authorized to have access permission on a method called 'Change User Name', regardless if the method has been exposed as a SOAP or REST service operation or not, the user OPERATIONS has the permission to access the 'Change User Name' operation of BOTH service types through the same grant.

- PL/SQL interfaces can be exposed as SOAP services with the support for both synchronous and asynchronous patterns. The security grants given for the selected method names would be applicable to the generated services of both patterns.

- If a selected interface contains overloaded functions, each of them can be uniquely granted through the create grant feature. If you select more than one overloaded function for the grant, an Overloaded column appears in the table with the selected function names checked.

***Create Grants Page with Overloaded Functions***




### *Revoking Security Grants*

The administrator can revoke security grants in the following ways:

- ***Revoking Commonly Assigned Grants to All Selected Procedures or Methods***

  Select more than one procedure and function or method that you want to revoke the grants created earlier, and click **Revoke Grant**. This opens the Revoke Grants page where you can find the existing grants that are commonly assigned to the selected methods.

  For example, a selected interface has the following grants:

| Method Names | Grantee |
|---|---|
| Change User Name | SYSADMIN<br>**OPERATIONS** |
| Test User Name | **OPERATIONS**<br>MKTMGR<br>BUSER |
| Validate User Name | BUSER<br>**OPERATIONS** |

  A specific User (grantee type) 'OPERATIONS' (grantee name) is commonly authorized to all the methods contained in the selected interface. Therefore, only

User 'OPERATIONS' is listed as the common grant for all the methods.

To revoke this common grant, select these three method checkboxes first, and then click **Revoke Grant**. This revokes the common grant, User 'OPERATIONS, assigned to these selected methods.

If there is more than one common grant listed in the table, select desired common grants from the table before clicking **Revoke Grant**.

- ***Revoking Grants for a Single Procedure and Function or Method***

  In the Grants tab of the interface details page, select a desired method and then click **Revoke Grant**. The Revoke Grants page displays the existing grants that have been created for the selected method.

  Select the grants that you want to revoke from the table, and click **Revoke Grant** to revoke the selected grants.

***Viewing Grant Details***

Each grant contains information about grantee type, grantee name, and whether the grant is authorized through a direct grant (such as a specific user 'OPERATIONS') or other grant method (such as through a user group 'Marketing Group').

To view grant details, click the **Grant** icon for the method that you want to view. A pop-up window appears with the grant details.

In addition to the Grants tab, you can view the grant details for a desired method from the SOAP Web Service tab and the REST Web Service tab.

**To create grants:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.

2. In the Integration Repository tab, select 'Interface Type' from the View By drop-down list.

3. Expand an interface type node and click an interface definition that can be exposed as a REST service or as both SOAP and REST services.

   The interface details page appears.

4. In the Grants tab, select one or more procedure and function or method names for which you want to create grants.

5. Click **Create Grant**. The Create Grants page appears.

6. Select a grantee type:

   - `Specific User`

- `Group of Users`
- `All Users`

7. If you select `Specific User` or `Group of Users`, specify the user or group for which to create the grants in the Grantee Name field.

8. Click **Create Grant**.

   The interface details page reappears.

**To view or revoke grants:**

You can view and revoke existing grants directly in the methods list on the interface details page.

1. Navigate to the selected interface that can be exposed as a REST service.

2. To view grant details:

   In the Grants tab, the REST Web Service tab, or the SOAP Web Service tab if it appears, click the **Grant** icon for a given operation. A pop-up window appears allowing you to view the grant details for the selected operation.

3. To revoke grants in the Grants tab:

   - To revoke common grants for all selected methods

     Select more than one method from the table and click **Revoke Grant**. The Revoke Grants page appears. Select one or more common grants from the table and click **Revoke Grant**.

   - To revoke grants for a single method

     Select a desired method from the table and then click **Revoke Grant**.

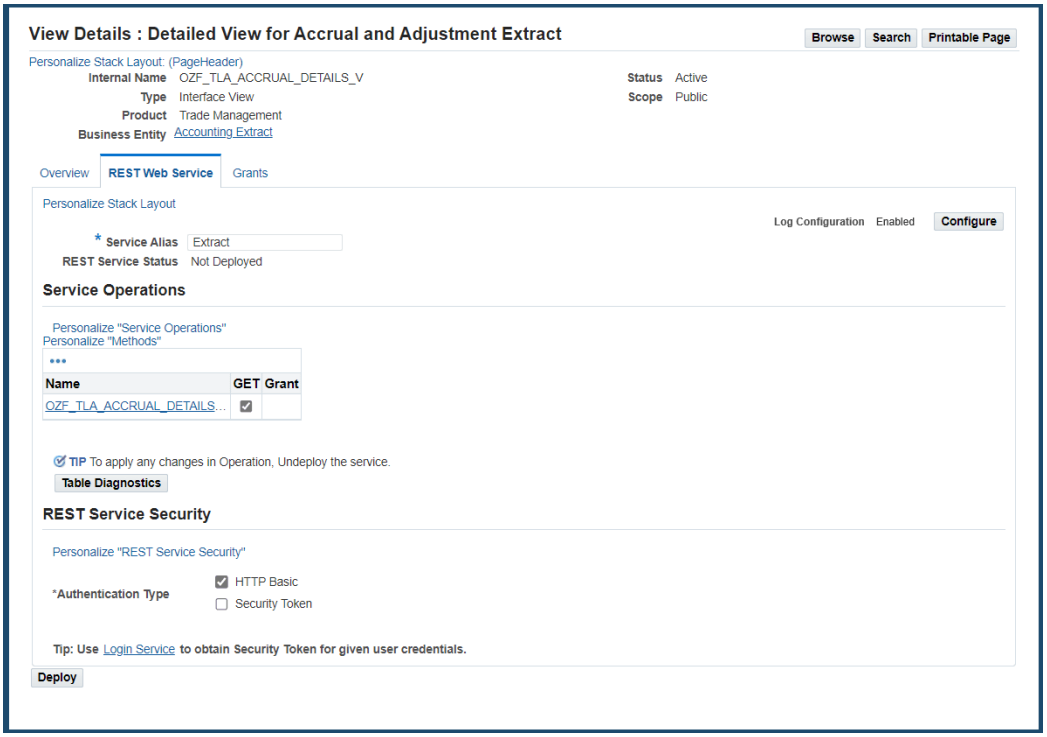
     Select one or more existing grants from the table and click **Revoke Grant** to revoke the grants.

## Enabling Design-Time Log Configuration for REST Services

Users who have the Integration Administrator role can enable design-time log for an interface that can be exposed as a REST service. This action records any issues if encountered during the REST service deployment and undeployment activities.

If the design-time log is enabled for a REST service, 'Enabled' is shown as the Log Configuration value in the REST Web Service tab of that REST service. Otherwise, 'Disabled' is displayed instead. If the same REST service can also be exposed as a SOAP service and the 'SOAP' design-time log is also enabled in a separate configuration, then you can find 'Enabled' shown in both the SOAP Web Service tab and the REST Web Service tab.

*REST Web Service Tab with Log Configuration 'Enabled'*




### Changing Existing Log Configurations

To change the existing design-time log configuration for the selected REST interface, click **Configure** next to the Log Configuration field in the REST Web Service tab. The Log & Audit Setup Details page is displayed with the selected interface where the administrator can add a new log configuration or update an existing configuration.

> **Note:** The Log & Audit Setup Details page can also be accessed by selecting the **Administration > Configuration** from the navigation menu.

For detailed information about how to configure log settings at the integration interface level, see Adding a New Configuration, page 7-6.

### Viewing Design-Time Logs

If the design-time log is enabled for a REST service, **View Log** appears allowing you to view both log messages and error messages if occurred during the design-time activities. See Viewing Design-Time Logs for REST Services, page 3-53.

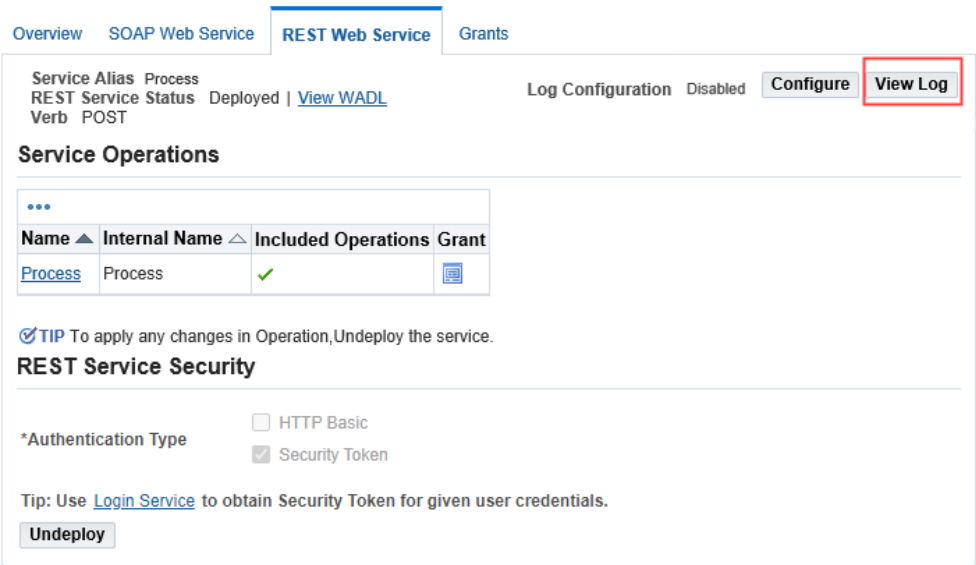
## Viewing Design-Time Logs for REST Services

Similar to viewing the generate and deploy time logs for SOAP services, the administrator can view design-time logs if available for REST services to troubleshoot

issues or exceptions encountered during each stage of REST service deployment lifecycle activities, such as deploy and undelploy actions.

- If the design-time log is enabled for the 'REST' service type of an interface during log configuration, **View Log** appears in the REST Web Service tab. Clicking **View Log** lets you view both log messages and error messages if occurred during design time for the selected REST service.

***REST Web Service Tab with 'View Log' Highlighted***




Clicking **View Log** displays the Log & Error Details page containing the following regions:

- **Error Details region:** If any errors or exceptions encountered during the design-time activities, error messages are displayed in the Error Details region.

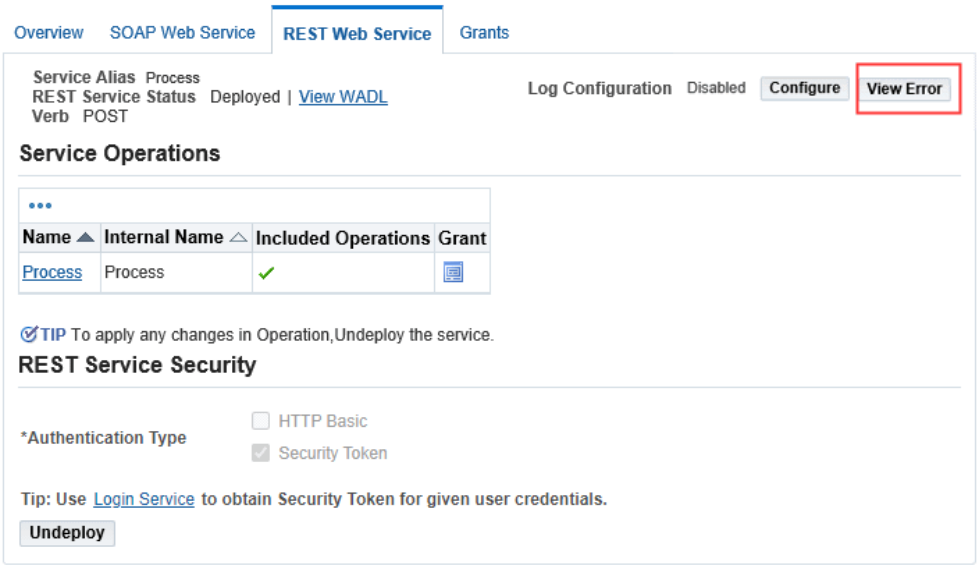
- **Log Details region:** All messages recorded for REST service type of the interface are listed in the table. Each log contains log sequence, log timestamp, module, log level, and actual message recorded at the design time.

  You can delete logs if needed by clicking **Delete Log** in the Log Details region.

  Before deleting the logs, you can save a backup copy by clicking **Export**. This allows you to export the records listed in the Log Details region to Microsoft Excel and use it later.

- If the design-time log is not enabled for the 'REST' service type of an interface, **View Error** appears instead in the REST Web Service tab allowing you to view the error messages only.

***REST Web Service Tab with 'View Error' Highlighted***




Clicking **View Error** allows you to view only the error or exception messages displayed in the Error Details region.

> **Note:** The Log Details region will not appear in this page because the design-time log is not configured for the REST service type of the selected interface.

For example, if the administrator receives errors or exceptions while trying to deploy a REST service, these errors are recorded and displayed in the Error Details region even if the design-time log for the REST service type is not configured for the interface.

For error messages, error codes, and possible solutions, see Error Messages, page C-1.

- For information on enabling the design-time log for an interface with the REST service type, see Adding a New Configuration, page 7-6.
- For more logging information, see Logging for Web Services, page 7-1.

At runtime during the service invocation, if a service has the runtime log enabled, log messages can be viewed in Service Monitor against that instance. For information on viewing log messages through Service Monitor, see Viewing Service Processing Logs, page 7-15.

# Managing Service Life Cycle and Security Grants Using an Ant Script

In addition to managing service lifecycle activities and creating security grants through the Integration Repository user interface, administrators can use an Ant script to perform these tasks through the command line:


- Managing SOAP Service Lifecycle Activities Using an Ant Script, page 3-56
- Managing REST Service Lifecycle Activities Using an Ant Script, page 3-64
- Managing Security Grants Using an Ant Script, page 3-72


## Managing SOAP Service Lifecycle Activities Using an Ant Script

An Ant script `$JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml` is used to implement the design-time activities for SOAP services such as generate, regenerate, deploy, undeploy, activate, retire, and reset services as well as to upgrade or postclone services from command line.

Note that `$JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml` is a multipurpose script. It can also be used to run the diagnostic tests or download the configuration file from the instance. The configuration file is the present state of instance in the view of Oracle E-Business Suite Integrated SOA Gateway context. The same configuration file is sometimes referred as service descriptor file.

> **Note:** When services are generated from command line, the settings selected from the Integration Repository user interface will take effect while generating the service artifacts. For example, if 'Asynchronous' interaction pattern is selected for a method contained in a PL/SQL interface, no matter if the service is generated from the UI or command line, only that selected single method has the associated artifact generated for asynchronous operation.

**Usage of $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml:**

```
ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml usage
```

> **Note:** Script creates log file at the script location; hence, it is suggested to copy `isgDesigner.xml` to some `<TEMP_DIRECTORY>` and then use the script present in `<TEMP_DIRECTORY>`.

### Usage Related to Design Activities

You can use the `isgDesigner.xml` script in one of the following ways:

- Run the script without arguments

```
ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml.
```

When prompted, enter the arguments.

> **Note:** Do not enclose any input between double quotes.

- Run the script with arguments, along with the commands

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
  Dactions=<comma separated list of operations> -
  DserviceType=SOAP -DirepNames=<comma separated list of API
  Names> -Dverbose=<ON|OFF>
  ```

  While passing actions and irepNames using this method, be aware of the following conditions:

  - If more than one actions or irepNames are passed as command line argument, enclose them between double quotes. For example,

    ```
    -Dactions="method1, method2,.."
    ```

    ```
    -DirepNames="ECRDTLD,FND_USER_PKG[{function1:SYNC}
    {function2:}...]"
    ```

  - If only one action or irepName is passed as command line argument, then there is no need to enclose between double quotes.

- Run the script by providing a descriptor file, along with the commands, as an input

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
  Dfile=<absolute path of service descriptor file> -
  Dverbose=<ON|OFF>
  ```

  The argument values required for processing the design-time activities are provided in the file.

  See Using the Script with an Input Descriptor File for SOAP Services, page 3-60.

**Argument Description**

Valid arguments for `isgDesigner.xml` are described as follows:

- **actions**: Comma separated list of actions to be performed. Supported operations are listed as follows:
  - generate: It generates or regenerates the service.
  - deploy: It deploys the generated service.
  - undeploy: It undeploys the deployed service.
  - activate: It activates the deployed service if it is in 'Retire' state.
  - retire: It retires the deployed service if it is in 'Active' state.

- reset: It resets the web service status to its initial state - 'Not Generated' and also deletes artifacts from the file system of Oracle SOA Suite server.

- upgrade: It upgrades a service from Oracle E-Business Suite Release 12.1.x to Release 12.2.

- postclone: It carries out postclone steps including redeploying the services on the Release 12.2 cloned environment.

While passing the action names, ensure that they have been given in the order of their life cycle. For example:

- Incorrect Usage: `-Dactions="deploy,generate"`

- Correct usage: `-Dactions="generate,deploy"`

Actions 'upgrade' and 'postclone' should be called independently. This means if the 'upgrade' action is given, actions argument should look like `-Dactions=upgrade`. It is similar to the case with action 'postclone'. More information on how actions arguments are used is described in the following examples:

- `-Dactions="generate,deploy,retire,activate,undeploy,reset"`

- `-Dactions=upgrade`

- `-Dactions=postclone`

Additionally, if action is 'upgrade' or 'postclone', only 'actions' and 'verbose' arguments will be used. However if you have given other arguments as well, only the three arguments mentioned above will be used.

- **serviceType**: (SOAP, [REST], BOTH, GRANT): "`REST`" is the default value for Service Type. Press the **Enter** key or choose the value "`SOAP`". To clone both SOAP and REST services, select "`BOTH`" as the value.

  "`GRANT`" is used to manage security grants. See: Managing Security Grants Using an Ant Script, page 3-72.

- **irepNames**: Comma separated list of interface names.

  Use either one of the following syntax for the interface name:

  - `interface_name`

  - `interface_name[{function1:<interactionPattern>}{function2:<interactionPattern>}{function3...}]`

    - The colon "`:`" after each `function` and before `<interactionPattern>` is mandatory.

    - `<interactionPattern>` is optional.

For example, if `<interactionPattern>` is not included such as `interface_name[{function1:}]`, then the interaction pattern will be defaulted to "`SYNC`".

If only `interface_name` is mentioned, then the old patterns will be generated. If the interface is a new API or has been reset, then all the functions will be generated with SYNC interaction pattern.

- Supported interaction patterns are SYNC, ASYNC, and BOTH. For example:
  - `interface_name[{function1:SYNC}]`
  - `interface_name[{function1:ASYNC}]`
  - `interface_name[{function1:BOTH}]`

  Passing an unsupported interaction pattern will result in an error.

Multiple interfaces can be separated by comma and passed as irepName using the following syntax:

```
interface_name1[{function1:}],interface_name2,interface_name3
[{function1:ASYNC}]
```

For example, `-DirepNames="ECRDTLD,FND_USER_PKG[{function1:SYNC}]"`

- **file**: Absolute path of the (service descriptor) XML file containing interfaces and actions to be performed on these interfaces.

  For example, `-Dfile=/u01/oracle/isg_service.xml`

- **verbose**: [ON|OFF] Default value is "`OFF`".

  For example, `-Dverbose=OFF`

**Usage Examples**

- Sample command for actions other than 'upgrade' and 'postclone' (actions and interface names are being passed):

```
ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
Dactions="generate,deploy,undeploy" -DserviceType=SOAP -
DirepNames="ECRDTLD,FND_USER_PKG"
```

- Sample command for performing design time actions from XML file:

```
ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
Dfile=/u01/oracle/isg_service.xml
```

- Sample command for action 'upgrade':

```
ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
Dactions=upgrade -Dverbose=OFF
```

- Sample command for action 'postclone':

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
  Dactions=postclone -Dverbose=ON
  ```

  ```
  Enter Service Type : (SOAP, [REST], BOTH)
  ```

**Using the Script with an Input Descriptor File for SOAP Services**

This section describes how to use a descriptor file with the required argument values to manage the design-time activities.

**Example 1 - Generating and Deploying a PL/SQL Service with All Functions and Generating SOAP Services for a Concurrent Program and a Business Service Object**

The following descriptor file for a PL/SQL API `FND_USER_PKG` provides required argument values highlighted in bold text, such as `<SOAP_ACTIONS>`, `<POLICY>`, and `<ALL_FUNCTIONS/>`, indicating that this is to generate SOAP service operations for all the functions contained in the API with synchronous pattern, and then deploy them with SAML Token authentication type.

```
<INTERFACE>
 <NAME>FND_USER_PKG</NAME>
 <TYPE>PLSQL</TYPE>
 <REST_ACTIONS>
 ...
 </REST_ACTIONS>
 <SOAP_ACTIONS>
  <RESET/>
  <GENERATE>
  <!-- GENERATES ALL FUNCTIONS WITH DEFAULT ITERACTION PATTERN "SYNC"
FOR PLSQL-->
   <ALL_FUNCTIONS/>
  </GENERATE>
  <!-- DEPLOYS WITH GIVEN POLICY "SAML" -->
  <DEPLOY>
  <POLICY>SAML</POLICY>
  </DEPLOY>
  <RETIRE/>
  <ACTIVATE/>
 </SOAP_ACTIONS>
</INTERFACE>
```

The following descriptor file is used for a concurrent program called INTERFACE5 to generate a SOAP service. Two functions, FUNCTION1 and FUNCTION2, contained in this interface are generated with the default synchronous pattern.

```
<INTERFACE>
 <NAME>INTERFACE5</NAME>
 <TYPE>CONCURRENTPROGRAM</TYPE>
 <REST_ACTIONS>
 ...
 </REST_ACTIONS>
   <SOAP_ACTIONS>
  <RESET/>
   <GENERATE>
   <FUNCTIONS_LIST>
    <FUNCTION>FUNCTION1</FUNCTION>
    <FUNCTION>FUNCTION2</FUNCTION>
   </FUNCTIONS_LIST>
  </GENERATE>
  <DEPLOY/>
  <RETIRE/>
  <ACTIVATE/>
 </SOAP_ACTIONS>
</INTERFACE>
```

A similar descriptor file can be used to generate a SOAP service for the business service object (BSO) interface type called `INTERFACE6` when the `<TYPE>SERVICEBEAN</TYPE>` is used, shown as follows:

```
<INTERFACE>
 <NAME>INTERFACE6</NAME>
 <TYPE>SERVICEBEAN</TYPE>
<REST_ACTIONS>
...
</REST_ACTIONS><SOAP_ACTIONS>
  <RESET/>
   <GENERATE>
   <FUNCTIONS_LIST>
    <FUNCTION>FUNCTION1</FUNCTION>
    <FUNCTION>FUNCTION2</FUNCTION>
   </FUNCTIONS_LIST>
  </GENERATE>
  <DEPLOY/>
  <RETIRE/>
  <ACTIVATE/></SOAP_ACTIONS>
</INTERFACE>
```

PL/SQL APIs, Concurrent Programs, and Business Service Objects can be exposed as both SOAP and REST services; therefore, the same descriptor file can include required argument values for REST service design-time activities as well. See: Using the Script with an Input Descriptor File for REST Services, page 3-69.

**Example 2 - Generating SOAP Service Operations with Synchronous, Asynchronous, or Both Synchronous and Asynchronous Patterns**

In this example, a PL/SQL interface called `INTERFACE1` contains six functions or operations. The descriptor file indicates the required task is just to generate SOAP services operations with various interaction patterns.

Specifically, `FUNCTION1` and `FUNCTION2` contained in the interface are generated with both synchronous and asynchronous patterns (`<FUNCTIONS_LIST pattern="BOTH" >`), `FUNCTION3` and `FUNCTION4` are generated with asynchronous pattern ( `<FUNCTIONS_LIST pattern="ASYNC">`), and `FUNCTION5` and `FUNCTION6` are generated with synchronous pattern (`<FUNCTIONS_LIST pattern="SYNC">`).

```
<INTERFACE>
 <NAME>INTERFACE1</NAME>
 <TYPE>PLSQL</TYPE>
 <REST_ACTIONS>
   ...
 </REST_ACTIONS>
 <SOAP_ACTIONS>
  <RESET/>
   <GENERATE>
   <!-- GENERATES GIVEN FUNCTIONS WITH ITERACTION PATTERN "BOTH" FOR
PLSQL-->
  <FUNCTIONS_LIST pattern="BOTH">
    <FUNCTION>FUNCTION1</FUNCTION>
    <FUNCTION>FUNCTION2</FUNCTION>
   </FUNCTIONS_LIST>
   <!-- GENERATES GIVEN FUNCTIONS WITH ITERACTION PATTERN "ASYNC" FOR
PLSQL-->
  <FUNCTIONS_LIST pattern="ASYNC">
    <FUNCTION>FUNCTION3</FUNCTION>
    <FUNCTION>FUNCTION4</FUNCTION>
   </FUNCTIONS_LIST>
   <!-- GENERATES GIVEN FUNCTIONS WITH ITERACTION PATTERN "SYNC" FOR
PLSQL-->
  <FUNCTIONS_LIST pattern="SYNC">
    <FUNCTION>FUNCTION5</FUNCTION>
    <FUNCTION>FUNCTION6</FUNCTION>
  </GENERATE>
  <DEPLOY/>
  <RETIRE/>
  <ACTIVATE/>
 </SOAP_ACTIONS>
</INTERFACE>
```

**Example 3 - Generating a SOAP Service for XML Gateway Interface Type**

The following example shows a descriptor file that is used to generate a SOAP service for the XML Gateway interface called `INTERFACE7`. In this example, all the functions contained in this interface are generated with SOAP service operations with the default synchronous pattern.

```
<INTERFACE>
 <NAME>INTERFACE7</NAME>
 <TYPE>XMLGATEWAY</TYPE>
   <SOAP_ACTIONS>
  <RESET/>
   <GENERATE>
   <ALL_FUNCTIONS/>
  </GENERATE>
  <DEPLOY/>
  <RETIRE/>
  <ACTIVATE/>
 </SOAP_ACTIONS>
</INTERFACE>
```

### Other Usages

In addition to performing design time activities, this `$JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml` script can be used for the following purposes:

- Deploying Generic XML Gateway Services, page 3-63

- Obtaining Argument irepNames Usage Information, page 3-63
- Running Diagnostic Tests, page 3-64

#### Deploying Generic XML Gateway Services

To deploy a generic XML Gateway service for the current environment, invoke this script with target *deployGenericXMLG*

```
ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml
deployGenericXMLG
```

For more information on deploying generic XML Gateway services, see *Installing Oracle E-Business Suite Integrated SOA Gateway, Release 12.2*, My Oracle Support Knowledge Document 1311068.1 for details.

#### Obtaining Argument irepNames Usage Information

To know how to pass argument `irepNames`, invoke this script with target *irepNamehelp*

```
ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml
irepNamehelp
```

This prints the following information on console window:

Each interface name for the `irepNames` argument should be given in one of the following way:

- `interface_name[{function1:<interactionPattern1>}{function2:<interactionPattern2>}{function3...}]`
- `interface_name`

Usage Example: FND_USER_PKG [{TESTUSERNAME:SYNC}{CHANGE_USER_NAME:ASYNC}],FND_GLOBAL

> **Note:** Patterns supported here are described in the following:
>
> - **SYNC:** This is for synchronous generation.
> - **ASYNC:** This is for asynchronous generation.
> - **BOTH:** This is for both synchronous and asynchronous generations.

**interface_name[{function1:pattern1}{function2:pattern2}]**

- Function `function1` of interface `interface_name` will be generated with pattern `Pattern1`.
- Function `function2` of interface `interface_name` will be generated with pattern `Pattern2.`

**interface_name**

All functions of the interface `interface_name` will be generated with old pattern. If

the interface is a new API or has been reset, then all the functions will be generated with SYNC interaction pattern.

#### Running Diagnostic Tests

Oracle E-Business Suite Integrated SOA Gateway provides a suite of diagnostic tests to help determine specific causes or issues with installation steps. When a test suite is run, multiple tests would be processed on both Oracle E-Business Suite and Oracle SOA Suite environments for diagnosing issues on various categories.

To know how to run different diagnostic tests, invoke this script with *diagnosticshelp*

```
ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml
diagnosticshelp
```

Additionally, you can run different diagnostics through the backend script with different targets. For more information on how to run these diagnostic tests, see Oracle E-Business Suite Integrated SOA Gateway Diagnostic Tests, page A-1.

## Managing REST Service Lifecycle Activities Using an Ant Script

Similar to SOAP services, the administrator can use an Ant script `$JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml` to implement the design-time activities for REST services such as deploy and undeploy services from command line.

**Usage of $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml:**

```
ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml usage
```

> **Note:** Script creates log file at the script location; hence, it is suggested to copy `isgDesigner.xml` to some `<TEMP_DIRECTORY>` and then use the script present in `<TEMP_DIRECTORY>`.

#### Usage Related to Design Activities

You can use the `isgDesigner.xml` script in one of the following ways:

- Run the script without arguments

  For example, enter `ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml`

  When prompted, enter the arguments.

  > **Note:** Do not enclose any input between double quotes.

- Run the script with arguments, along with the commands

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
  Dactions=<comma separated list of operations> -
  DserviceType=REST -DirepNames=<interface_name[{function1:
  <interactionPattern>:<VerbList>}{function2...}]> -
  ```

```
Dverbose=<ON|OFF> -Dalias=<Alias>
```

For example:

- ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -Dactions=deploy -DserviceType=REST -DirepNames=FND_USER_PKG[{TESTUSERNAME:SYNC:POST}] -Dverbose=ON -Dalias=FndUserPkgSvc
  ```

- ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -Dactions=deploy -DserviceType=REST -DirepNames="FND_USER_PKG[{TESTUSERNAME:SYNC:POST}],FND_MESSAGE[{GET_TEXT_NUMBER::POST}]" -Dverbose=ON -Dalias="FndUserPkgSvc,FndMessageSvc"
  ```

While passing actions and irepNames using this method, be aware of the following conditions:

- If more than one action or irepName is passed as command line argument, enclose them between double quotes. For example,

  ```
  -Dactions="method1, method2,.."
  ```

  ```
  -DirepNames="FND_USER_PKG[{function1:SYNC:POST}{function2::}],HR_APPRAISALS_API"
  ```

- If only one action or irepName is passed as command line argument, then there is no need to enclose between double quotes.

- Run the script by providing a descriptor file, along with the commands, as an input

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -Dfile=<absolute path of service descriptor file> -Dverbose=<ON|OFF>
  ```

  The argument values required for processing the design-time activities are provided in the file.

  See Using the Script with an Input Descriptor File for REST Services, page 3-69.

**Argument Description**

Valid arguments for `isgDesigner.xml` are described as follows:

- **actions**: Comma separated list of actions to be performed. Supported operations are listed as follows:

  - deploy: It generates the REST service artifacts and deploys the generated service.

  - undeploy: It undeploys the deployed service and resets the service status to its initial state - 'Not Deployed'. This also deletes the service artifacts from the Oracle E-Business Suite managed server.

  - postclone: It carries out postclone steps including redeploying the services on

the Release 12.2 cloned environment.

While passing the action names, ensure that they have been given in the order of their life cycle. For example,

```
-Dactions="deploy,undeploy"
```

Action 'postclone' should be given independently. For example:

```
-Dactions="postclone"
```

Note that when passing the action 'postclone', only 'actions' and 'verbose' arguments will be used. For example,

```
ant -f  $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
Dactions="postclone" -Dverbose=ON
```

- **serviceType**: (SOAP, [REST], BOTH, GRANT): "`REST`" is the default value for Service Type. Press the **Enter** key or choose the value "`REST`". To clone both SOAP and REST services, select "`BOTH`" as the value.

  "`GRANT`" is used to manage security grants. See: Managing Security Grants Using an Ant Script, page 3-72.

- **irepNames**: Comma separated list of interface names.

  Use either one of the following syntax for the interface name:

  - `interface_name`
  - `interface_name[{function1:<interactionPattern>:<VerbList>}{function2:<interactionPattern>:<VerbList>}{funcntion3...}]`
    - Angle bracket "`< >`" represents an optional place value holder.
      - `<VerbList>` represents the list of verbs separated by "`+`", such as `interface_name[{function1:SYNC:GET+POST}]`.

        **Note:** The colon "`:`" is mandatory in these examples.

        For example, four HTTP verbs (GET, POST, PUT, and DELETE) can be supported for an open interface table 'RAXMTR' with `Inbound` direction:

        ```
        -DirepNames=RAXMTR[{RA_INTERFACE_LINES_ALL:SYNC:
        GET+POST+PUT+DELETE}]
        ```

        If `<VerbList>` is not included, such as `interface_name [{function1:SYNC:}]`, then the verb list will be defaulted to the supported verbs for the interface.

        Please note that the supported verb for Concurrent Programs is POST only; the supported verbs for Java Bean Services and Application Module Services are all the annotated verbs and POST. For PL/SQL

APIs and Business Service Object interfaces, the supported verbs are POST and GET. For Open Interface Tables with `Outbound` direction and Open Interface Views, the supported verb is GET only.

- `<interactionPattern>` represents interaction pattern. For REST services, the supported interaction pattern is SYNC (synchronous) only.

  If `<interactionPattern>` is not included, such as `interface_name[{function1::POST}]`, then the interaction pattern will be defaulted to SYNC.

  If both optional place value holders are not included, such as `interface_name[{function1::}]`, then the interaction pattern will be defaulted to SYNC and the verb list will be defaulted to the supported verbs for the interface.

  If only `interface_name` is mentioned, then the all the functions will be generated with the synchronous interaction pattern with the supported verbs.

  Passing an unsupported interaction pattern or verb will result in an error.

Multiple interfaces can be separated by comma and passed as irepName.

For example, `-DirepNames="oracle.apps.fnd.rep.ws.service.EbsRestLocator[{function1:SYNC:GET+POST}],FND_USER_PKG[{function1::POST}]"`

- **file**: Absolute path of the (service descriptor) XML file containing interfaces and actions to be performed on these interfaces.

  For example, `-Dfile=/u01/oracle/isg_service.xml`

- **verbose**: [ON|OFF] Default value is OFF.

  For example, `-Dverbose=OFF`

- **alias**: It is mandatory for REST services. If multiple services are deployed, use comma separated alias names.

  For example, `-Dalias="FndUserPkgSvc,FndMessageSvc"`

- **policy**: (BASIC, TOKEN, [BOTH])

  "`BOTH`" is the default value for security policy. Press the **Enter** key or choose the value "`BOTH`" for both the HTTP Basic and Token based security authentication types. To deploy a REST service with the HTTP Basic authentication type or Token based security type, select "`BASIC`" or "`TOKEN`" respectively.

**Usage Examples**

- Sample command for actions (actions and interface names are being passed):
  - Deploy 'TESTUSERNAME' contained in the PL/SQL API 'FND_USER_PK' as a REST service operation called `FndUserPkgSvc` with the POST HTTP verb:

    ```
    ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
    Dactions=deploy -DserviceType=REST -DirepNames=FND_USER_PKG
    [{TESTUSERNAME:SYNC:POST}] -Dverbose=ON -
    Dalias=FndUserPkgSvc
    ```

    ```
    Enter Authentication Type(s): (BASIC, TOKEN, [BOTH])
    ```

    "`BOTH`" is the default value for the authentication type. To deploy a REST service with both the HTTP Basic and Token Based security types, select "`BOTH`" as the value.

  - Deploy the following PL/SQL APIs as REST services using one command:
    - 'TESTUSERNAME' contained in the PL/SQL API 'FND_USER_PK' as a REST service operation called `FndUserPkgSvc` with the POST HTTP verb
    - 'GET_TEXT_NUMBER' contained in the PL/SQL API 'FND_MESSAGE' as a REST service operation called `FndMessageSvc` with the POST HTTP verb

    ```
    ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
    Dactions=deploy -DserviceType=REST -DirepNames="
    FND_USER_PKG[{TESTUSERNAME:SYNC:}],FND_MESSAGE
    [{GET_TEXT_NUMBER::POST}]" -Dverbose=ON -Dalias="
    FndUserPkgSvc,FndMessageSvc"
    ```

    ```
    Enter Authentication Type(s): (BASIC, TOKEN, [BOTH])
    ```

    "`BOTH`" is the default value for the authentication type. To deploy a REST service with both the HTTP Basic and Token Based security types, select "`BOTH`" as the value.

  - Undeploy and then deploy an open interface table 'RA_INTERFACE_LINES_ALL' contained in the 'RAXMTR' open interface as a REST service operation called `raxmtr` with four supported HTTP verbs:

    ```
    ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
    DirepNames=RAXMTR[{RA_INTERFACE_LINES_ALL:SYNC:
    GET+POST+PUT+DELETE}] -DserviceType=REST -Dalias=raxmtr -
    Dactions="undeploy,deploy" -Dverbose=ON
    ```

  - Deploy an open interface view 'EGO_ITEM_SYNC_V' as a REST service called `ego` with the GET HTTP verb:

    ```
    ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
    DirepNames=EGO_ITEM_SYNC_V[{EGO_ITEM_SYNC_V:SYNC:GET}] -
    DserviceType=REST -Dalias=ego -Dactions="deploy" -
    Dverbose=ON
    ```

    ```
    Enter Authentication Type(s): (BASIC, TOKEN, [BOTH])
    ```

    "`BOTH`" is the default value for the authentication type. To deploy a REST

service with the HTTP Basic security type, select "BASIC" as the value.

- Clone REST services from an existing 12.2.x environment

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
  Dactions="postclone" -Dverbose=ON
  ```

  ```
  Enter Service Type : (SOAP, [REST], BOTH)
  ```

  "REST" is the default value for service type. To clone both SOAP and REST services, select "BOTH" as the value.

- Sample command for performing design time actions from XML file:

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
  Dfile=/u01/oracle/isg_service.xml
  ```

**Using the Script with an Input Descriptor File for REST Services**

This section describes how to use a descriptor file with the required argument values to manage the design-time activities for REST services.

**Example 1 - Deploying Concurrent Programs as REST Service Operations**

The following descriptor file for a concurrent program provides required argument values highlighted in bold text, such as `<REST_ACTIONS>`, `<ALIAS>shipment</ALIAS>`, `<ALL_FUNCTIONS/>`, and `<POLICY>`, to deploy the method contained in this `WSHDSNO` concurrent program as a REST service with the POST HTTP method and the HTTP Basic authentication type.

Note that by default Concurrent Programs can be exposed as REST services only with the POST HTTP method and synchronous interaction pattern. Therefore, the argument `<ALL_FUNCTIONS/>` works the same as the argument `<ALL_FUNCTIONS pattern="SYNC"/>`, `<ALL_FUNCTIONS verb="POST"/>`, or `<ALL_FUNCTIONS pattern="SYNC" verb="POST"/>` in the descriptor file.

```
<INTERFACE>
  <NAME>WSHDSNO</NAME>
  <TYPE>CONCURRENT</TYPE>
  <REST_ACTIONS>
     <DEPLOY>
       <ALIAS>shipment</ALIAS>
       <!-- GENERATES METHOD WITH DEFAULT VERB "POST" AND DEFAULT
INTERACTION PATTERN "SYNC" FOR CONCURRENT PROGRAM-->
      <ALL_FUNCTIONS/><POLICY>BASIC</POLICY>
    </DEPLOY>
    <UNDEPLOY/>
  </REST_ACTIONS>
  <SOAP_ACTIONS>
   ...
  </SOAP_ACTIONS>
</INTERFACE>
```

Because Concurrent Programs can be exposed as both SOAP and REST services, you can use the same descriptor file to include required argument values for the SOAP service design-time activities as well. See: Using the Script with an Input Descriptor File for SOAP Services, page 3-60.

**Example 2 - Deploying PL/SQL APIs, Java Bean Services, Application Module Services, and Business Service Objects as REST Service Operations with Both the POST and GET HTTP Methods**

This section explains the descriptor files for PL/SQL APIs, Java Bean Services, Application Module Services, and Business Service Objects REST services with the support for both the POST and GET methods.

Specifically, the REST service alias name is provided in the `<ALIAS>` argument. All service operations that need to be deployed with both the GET and POST HTTP methods are indicated in the argument `<ALL_FUNCTIONS pattern="SYNC" verb="GET,POST"/>`, as shown in the following samples:

- PL/SQL APIs and Business Service Objects (Both SOAP and REST Services)

  Similar to Concurrent Programs that can be exposed as both REST and SOAP services, you can use the same descriptor files as explained below for these two interface types to include required argument values for both REST and SOAP service design-time activities. See: Using the Script with an Input Descriptor File for SOAP Services, page 3-60.

  - Example for a PL/SQL API

    Use the following descriptor file that provides required argument values highlighted in bold text, such as `<REST_ACTIONS>`, `<ALIAS>USER</ALIAS>`, `<ALL_FUNCTIONS/>`, and `<POLICY>`, to deploy all functions contained in this PL/SQL API `FND_USER_PKG` as a REST service with the POST and GET HTTP methods and Security Token based security.

```
<INTERFACE>
  <NAME>FND_USER_PKG</NAME>
  <TYPE>PLSQL</TYPE>
  <REST_ACTIONS>
     <DEPLOY>
       <ALIAS>USER</ALIAS>
       <!-- GENERATES ALL FUNCTIONS WITH DEFAULT VERB "POST" AND
DEFAULT ITERACTION PATTERN "SYNC" FOR PLSQL-->
      <ALL_FUNCTIONS pattern="SYNC" verb="GET,POST">
<POLICY>TOKEN</POLICY>
    </DEPLOY>
    <UNDEPLOY/>
  </REST_ACTIONS>
  <SOAP_ACTIONS>
   ...
  </SOAP_ACTIONS>
</INTERFACE>
```

  - Example for a Business Service Object

    The following sample descriptor file is used to deploy a BSO interface `INTERFACE6` (with `<TYPE>SERVICEBEAN</TYPE>`) as a REST service with both the POST and GET HTTP methods as well as Security Token based security. **`ALIAS6`** representing the REST service alias is provided in the `<ALIAS>` argument.

```
<INTERFACE>
  <NAME>INTERFACE6</NAME>
  <TYPE>SERVICEBEAN</TYPE>
  <REST_ACTIONS>
     <DEPLOY>
       <ALIAS>ALIAS6</ALIAS>
       <!-- GENERATES ALL FUNCTIONS WITH VERB "GET,POST" AND
ITERACTION PATTERN "SYNC" FOR BSO-->
      <ALL_FUNCTIONS pattern="SYNC" verb="GET,POST">
<POLICY>TOKEN</POLICY>
    </DEPLOY>
    <UNDEPLOY/>
  </REST_ACTIONS>
  <SOAP_ACTIONS>
   ...
  </SOAP_ACTIONS>
</INTERFACE>
```

- Java Bean Services and Application Module Services (REST Services Only)

  The following example uses a descriptor file to deploy an interface type of Java Bean Services called `INTERFACE4` (with `<TYPE>JAVA</TYPE>`) as a REST service with both the POST and GET HTTP methods as well as Security Token based security.

  Specifically, the REST service alias name is provided in the `<ALIAS>ALIAS4</ALIAS>` argument. All service operations that need to be deployed with the GET and POST HTTP methods are indicated in the argument `<ALL_FUNCTIONS pattern="SYNC" verb="GET,POST"/>`, as shown below:

```
<INTERFACE>
  <NAME>INTERFACE4</NAME>
  <TYPE>JAVA</TYPE>
  <REST_ACTIONS>
     <DEPLOY>
       <ALIAS>ALIAS4</ALIAS>
       <!-- GENERATES ALL FUNCTIONS WITH VERB "GET,POST" AND
ITERACTION PATTERN "SYNC" FOR POJO SERTVICES-->
      <ALL_FUNCTIONS pattern="SYNC" verb="GET,POST">
<POLICY>TOKEN</POLICY>
    </DEPLOY>
    <UNDEPLOY/>
  </REST_ACTIONS>
</INTERFACE>
```

**Other Usages**

The `$JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml` script is a multipurpose script. You can also use it to run the diagnostic tests or download the configuration file from the instance.


- Deploying Generic XML Gateway Services, page 3-63
- Obtaining Argument irepNames Usage Information, page 3-63
- Running Diagnostic Tests, page 3-64


To manage lifecycle activities for SOAP services, see: Managing SOAP Service Lifecycle

Activities Using An Ant Script, page 3-56.

## Managing Security Grants Using an Ant Script

You can use the same `isgDesigner.xml` script to manage security grants in one of the following ways:

- Run the script without arguments

  For example, enter `ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml`

  When prompted, enter the arguments.

  > **Note:** Do not enclose any input between double quotes.

- Run the script with arguments, along with the commands

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
  Dactions=<create | revoke> -DserviceType=GRANT -
  DirepNames=<interface_name[function1:function2:..]> -
  Dverbose=<ON|OFF> -DgranteeType=<USER | GROUP | GLOBAL> -
  DgranteeKey=<Grantee Key>
  ```

  See Using the Script with Arguments for the Grant, page 3-73.

- Run the script by providing a descriptor file, along with the commands, as an input

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
  Dfile=<absolute path of service descriptor file> -
  Dverbose=<ON|OFF>
  ```

  The argument values required for processing the design-time activities are provided in the file.

  See Using the Script with an Input Descriptor File for the Grant, page 3-74.

**Argument Description**

In addition to some common arguments, such as -DirepNames, -Dverbose, described earlier in Managing REST Service Lifecycle Activities Using An Ant Script, page 3-64, the following arguments are specifically used in `isgDesigner.xml` for managing security grants:

- **actions:** Comma separated list of actions to be performed. Supported operations are listed as follows:
  - create: It creates a security grant for a selected interface or service.
  - revoke: It removes a grant created earlier, including the privileges of a grantee of any type (such as user, group, or global) assigned to the grant.
- **serviceType** (SOAP, [REST], BOTH, GRANT): "`REST`" is the default value for

Service Type. Select the value "`GRANT`" to create or revoke a grant.

- **granteeType:** Supported values are:
  - USER: It grants the access privilege of a selected interface or service to a specific user only.
  - GROUP: It grants the access privilege of a selected interface or service to a specific group only.
  - GLOBAL (default): This is the default value for the argument. It grants the access privilege of a selected interface or service to all Oracle E-Business Suite users.
- **granteeKey:** A required argument to provide a specific user or group value when granteeType is USER or GROUP. It is not required when the granteeType value is GLOBAL.
  - If granteeType is USER, provide user code (user name) as granteeKey.
  - If granteeType is GROUP, provide responsibility code as granteeKey.

**Using the Script with Arguments for the Grant**

The following examples explain how to use the script with arguments to create and revoke security grants:

- Create a grant to a specific user "OPERATIONS" with the access privileges of CHANGE_USER_NAME and TESTUSERNAME service operations within the FND_USER_PKG interface:

  > **Note:** `OPERATIONS` in the `DgranteeKey` argument is the user name (user code) value for an Oracle E-Business Suite user.

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
  Dactions=create -DserviceType=GRANT -DirepNames=FND_USER_PKG
  [CHANGE_USER_NAME:TESTUSERNAME] -Dverbose=OFF -
  DgranteeType=USER -DgranteeKey=OPERATIONS
  ```

- Revoke the privileges from the group SYSTEM_ADMINISTRATION that has given the access of all service operations in the FND_MESSAGE interface and the CHANGE_USER_NAME operations within the FND_USER_PKG interface:

  > **Note:** `FND_RESP|ICX|SYSTEM_ADMINISTRATION|STANDARD` in the `DgranteeKey` argument is the responsibility code value for the System Administration responsibility.

  ```
  ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml -
  ```

```
Dactions=revoke -DserviceType=GRANT -DirepNames=FND_MESSAGE,
FND_USER_PKG[CHANGE_USER_NAME] -Dverbose=ON -
DgranteeType=GROUP -
DgranteeKey=FND_RESP|ICX|SYSTEM_ADMINISTRATION|STANDARD
```

**Using the Script with an Input Descriptor File for the Grant**

This section describes how to use a descriptor file with the required argument values to manage security grants.

For file mode, you can provide grant information either using `granteeKey` or using `granteeName` and `granteeSource`. `granteeName` and `granteeSource` are not available when running the script in console mode.

For example, use the following descriptor file to:

- Create the following three grants.
  - Grant all Oracle E-Business Suite users the access privileges of PROG_APPL_ID, CONC_LOGIN_ID, LOGIN_ID, and APPS_INITIALIZE service operations within a selected interface.
  - Grant the access privilege of CONC_REQUEST_ID service operation to the user group who has the `System Administration` responsibility only.
  - Grant the access privilege of CONC_REQUEST_ID service operation to the Oracle E-Business Suite user `OPERATIONS` only.
- Remove the access privileges of the selected interface from all Oracle E-Business Suite users.

```
<INTEGRATION_REPOSITORY>
  <INTERFACE>
   ...
    <GRANT_ACTIONS>
      <CREATE>
         <FUNCTIONS_LIST grantType="GLOBAL">
          <FUNCTION>PROG_APPL_ID</FUNCTION>
          <FUNCTION>CONC_LOGIN_ID</FUNCTION>
          <FUNCTION>LOGIN_ID</FUNCTION>
          <FUNCTION>APPS_INITIALIZE</FUNCTION>
         </FUNCTIONS_LIST>
         <FUNCTIONS_LIST grantType="GROUP" granteeName="System
Administration" granteeSource="FND_RESP">
          <FUNCTION>CONC_REQUEST_ID</FUNCTION>
         </FUNCTIONS_LIST>
         <FUNCTIONS_LIST grantType="grantType="USER" granteeKey="
OPERATIONS">
          <FUNCTION>CONC_REQUEST_ID</FUNCTION>
         </FUNCTIONS_LIST>
      </CREATE>
      <REVOKE>
         <ALL_FUNCTIONS grantType="GLOBAL"/>
      </REVOKE>
    </GRANT_ACTIONS>
  </INTERFACE>
</INTEGRATION_REPOSITORY>
```

# 4

# Administering Composite Services - BPEL

## Overview

A composite service is a set of specifications that define a way of assembling SOA-based application. It may consist of one or more services to describe a complex business process requirement. For example, a composite service - BPEL type can be used for service orchestration to manage more complex business processes (such as Order-to-Receipt) which may be handled by various applications.

A composite service - BPEL type contains its own WSDL definition and service endpoints allowing external web service clients to invoke the services at runtime.

In Oracle SOA Suite 11g and Oracle SOA Suite 12c, BPEL process is managed and deployed together with the associated SOA composite application. In Oracle SOA Suite 10g, it is developed and deployed as a separate component. Integration Repository displays 'Composite Services - BPEL' of Oracle SOA Suite 10g as catalogue in this release.

This chapter includes the following topics:



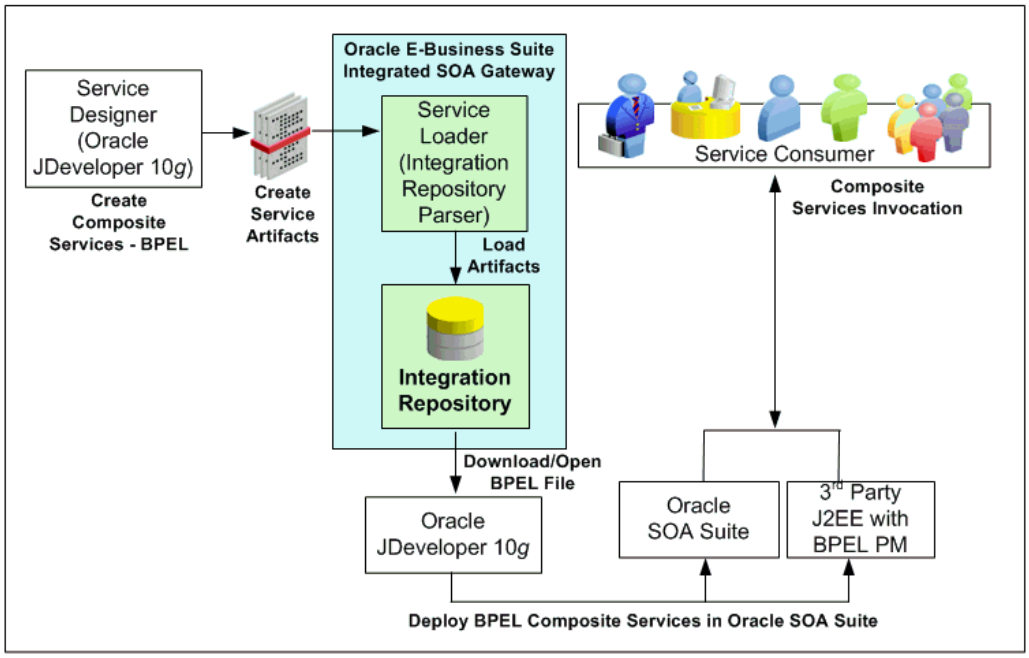
## Understanding the Enablement Process for Composite Services - BPEL

To design a composite service, an integration developer uses BPEL process component in Oracle JDeveloper 10g (Service Designer) to assemble a series of service components together for a business function. The newly created composite service - BPEL definition needs to be annotated first based on the Integration Repository annotation standards. Users who have the Integration Administrator role need to validate the annotated files using a standalone design time tool called Integration Repository Parser. An Integration Repository loader (iLDT) file is generated after the validation and then uploaded to the Integration Repository using the FNDLOAD command. The composite service - BPEL

type can be displayed and searched from the Integration Repository user interface.

The following diagram illustrates the high level enablement process:

***Enablement Process for Composite Services - BPEL***




Users granted the download composite service privilege through Integration Repository Download Composite Service Permission Set (FND_REP_DOWNLOAD_PERM_SET) can download the composite - BPEL file to their local directories. An integration developer can open the downloaded BPEL file using Oracle JDeveloper 10g and modify it if necessary before deploying it to a BPEL server in Oracle SOA Suite 10g for service consumption.

> **Note:** Composite services - BPEL type is supported in Oracle SOA Suite 10g. For example, a composite - BPEL type can be deployed through Oracle JDeveloper to a BPEL server in Oracle SOA Suite 10g BPEL Process Manager or a third party BPEL PM in a J2EE environment.

For detailed information on how to upload composite - BPEL definitions to the Integration Repository, see Enabling Custom Integration Interface Process Flow, page 5-2.

For information on Integration Repository annotation standards, see Composite Service - BPEL Annotations, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

# Administering Composite Services - BPEL

Oracle E-Business Suite Integrated SOA Gateway allows you to perform the following tasks on composite services:

- Viewing Composite Services - BPEL, page 4-3

  Similar to all other users, integration administrators can view composite service - BPEL details, including the abstract WSDL file and BPEL file of the composite service.

- Downloading Composite Services - BPEL, page 4-4

  Apart from viewing the composite service - BPEL details, the administrators can download the .ZIP file for a composite service - BPEL type if it is available for download.

## Viewing Composite Services - BPEL

Once annotated custom composite - BPEL definitions are uploaded to the Integration Repository, 'Composite - BPEL' option can be listed when searching by Interface Type and visible to all users.

Integration administrators can view composite details for a selected composite service including service name, description, BPEL file, WSDL file, and other annotated information.

To locate a composite service - BPEL, navigate to the Composite Service interface type from the Oracle Integration Repository browser window with View By 'Interface Type' or perform a search by selecting Composite service (such as 'Composite - BPEL') interface type in the Search page. Click your desired composite service name link from the browser tree or the search result to display the composite service - BPEL interface details page where you can:

- View the composite service - BPEL details.
- View the composite service - BPEL abstract WSDL file by clicking the **View Abstract WSDL** link.
- View the BPEL file by clicking the **View BPEL File** link in the BPEL Files region.
- Download a corresponding composite service - BPEL project file to your local directory.

For information on Integration Repository annotation standards, see Composite Service - BPEL Annotations, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

For detailed information on how to upload composite - BPEL definitions to the Integration Repository, see Enabling Custom Integration Interface Process Flow, page 5-

.

## Downloading Composite Services - BPEL

In addition to viewing composite service - BPEL details, a WSDL file, and BPEL file, users who have the Integration Administrator role can download a BPEL .JAR file containing relevant composite service files to their local machines by clicking **Download Service** in the composite service - BPEL details page.

> **Important:** In general, only users with the Integration Developer role and the Integration Administrator role can download the composite services - BPEL. However, users who are granted the download composite service privilege through Integration Repository Download Composite Service Permission Set (FND_REP_DOWNLOAD_PERM_SET) can also perform the download action. Otherwise, **Download Service** may not appear in the details page by default.
>
> For more information about how to grant the download composite service privilege, see Role-Based Access Control (RBAC) Security, page 6-3.

**To download a composite service - BPEL:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility and the Integration Repository link.
2. In the Integration Repository tab, select 'Interface Type' from the View By drop-down list.
3. Expand the Composite - BPEL interface type node to locate your desired composite service.
4. Click the composite service - BPEL that you want to download it to open the Composite Service- BPEL interface details page.
5. Click **Download Service** to download the selected composite - BPEL file to your local directory.

# 5

# Administering Custom Integration Interfaces and Services

## Overview

Oracle E-Business Suite Integrated SOA Gateway supports custom integration interfaces and allows them to be published along with Oracle seeded ones through the Oracle Integration Repository where they can be exposed to all users.

Custom interface definitions can be created for various interface types, including custom interface definitions for XML Gateway Map, Business Event, PL/SQL, Concurrent Program, Business Service Object, Java APIs, Java Bean Services, Application Module Services, and Composite Service for BPEL type. Depending on your business needs, integration developers can create and annotate custom interface definitions based on Integration Repository Annotation Standards. The annotated definitions can then be validated and uploaded to Oracle Integration Repository.

> **Note:** Custom interface types of EDI, Open Interface Tables, Open Interface Views, and Java APIs for Forms interfaces are not supported in this release.
>
> Oracle Integration Repository currently does not support the creation of custom Product Family and custom Business Entity.

After the upload, these custom integration interfaces are displayed in the Integration Repository based on the interface types they belong to. To easily distinguish them from Oracle integration interfaces, Interface Source "Custom" is used to categorize those custom integration interfaces in contrast to Interface Source "Oracle" for Oracle seeded interfaces in Oracle E-Business Suite. Custom integration interfaces can now seamlessly leverage the Oracle E-Business Suite Integrated SOA Gateway capabilities. Custom integration interfaces of service enabled interface types can be exposed as web services. Integration administrators perform the same administrative tasks for custom integration interfaces as they do for native integration interfaces. These tasks include

creating security grants, as well as generating and managing services throughout the deployment life cycle.

### Usage Guidelines for Custom Web Services

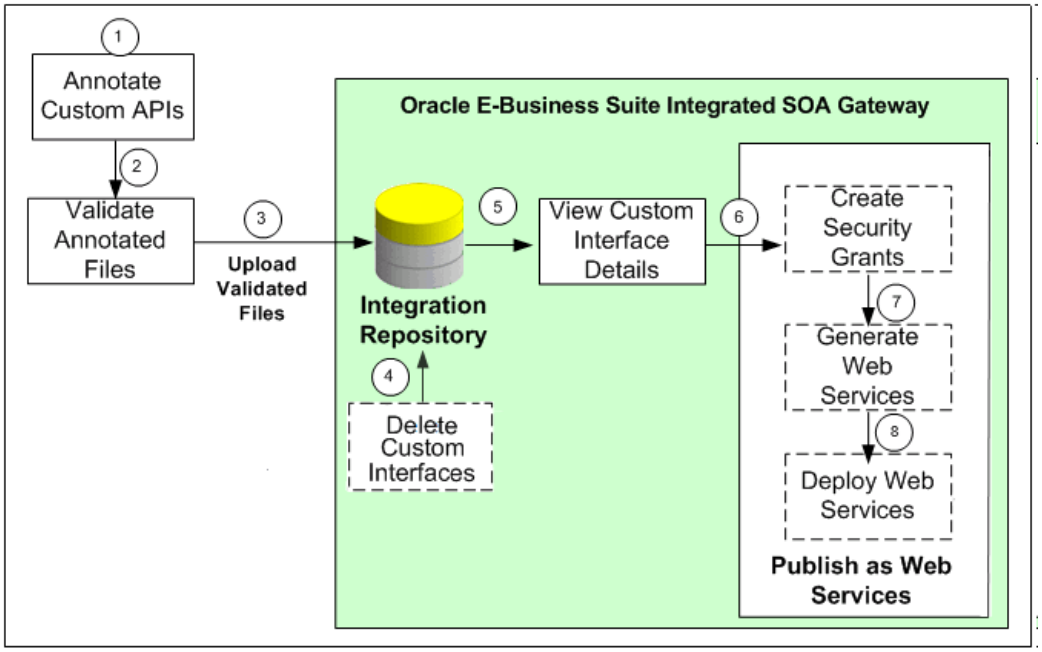
While creating or developing custom services for your business needs, consider the following conditions:

| Requirement | Use |
|---|---|
| To enable existing or new Oracle E-Business Suite customizations built on native Oracle E-Business Suite technologies (such as PL/SQL, Business Service Objects, and other supported custom integration interface types described earlier) as web services | Oracle E-Business Suite Integrated SOA Gateway |
| To integrate Oracle E-Business Suite with SOA application that requires rich service infrastructure and integration capabilities such as Business Rules, Business Activity Monitoring (BAM), web service development and orchestration | Oracle SOA Suite in conjunction with Oracle E-Business Suite Integrated SOA Gateway |
| To develop custom services that are not associated with Oracle E-Business Suite | Oracle WebLogic web service stack |

### Enabling Custom Integration Interface Process Flow

The following diagram illustrates the entire process flow of enabling custom integration interfaces:

***Custom Integration Interfaces Development Process Flow***




1. Users with the Integration Developer role annotate custom integration interface definition based on the Integration Repository annotation standards for the supported interface types.

   See: Integration Repository Annotation Standards, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

   **Note:** For custom PL/SQL APIs (simple data types only) that are created with a custom schema, you can publish such custom APIs in Oracle Integration Repository. Additionally, perform the following tasks for such APIs with a custom schema:

   1. Grant access to APPS schema.

      1. Connect to a custom schema as `EBS_SYSTEM` if your instance is on AD and TXK Delta 13 release update packs (RUPs) or later, or as `SYSTEM` if your instance is on an earlier AD and TXK RUP:

         ```
         sqlplus '/ as EBS_SYSTEM'
         ```

         **Note:** The R12.AD.C.Delta.13 and R12.TXK.C.Delta.13 RUPs introduce the `EBS_SYSTEM` schema. If you are running

Release Update Packs for AD and TXK Delta 13 or later, database privileges are granted to the Oracle E-Business Suite administration account, `EBS_SYSTEM`. Only the minimally required database privileges required to run Oracle E-Business Suite are granted to APPS by `EBS_SYSTEM`. For more information, refer to:

- Document 2755875.1, *Oracle E-Business Suite Release 12.2 System Schema Migration*
- Document 2758993.1, *Managing Database Privileges in Oracle E-Business Suite Release 12.2 (Running adgrants.sql)*

2. Use the following command to grant access:

```
GRANT EXECUTE on <custom_schema>.
<custom_package> TO APPS;
```

2. Create a synonym for the custom stored procedure.

1. Connect to APPS schema using the following command:

```
sqlplus <APPS Username>
Enter password: password
```

2. Use the following command to create a synonym:

```
CREATE SYNONYM <custom_package> FOR
<custom_schema>.<custom_package>;
```

2. Users who have the Integration Administrator role validate the annotated custom interface definitions against the annotation standards. This validation is performed by running the Integration Repository Parser (IREP Parser), a design time tool, to read the annotated files and then generate an Integration Repository loader file (iLDT ) if no error occurred. For more information, see:

- Setting Up and Using the Integration Repository Parser, page 5-6
- Generating ILDT Files, page 5-11

3. Users who have the Integration Administrator role upload the generated iLDT file

to Oracle Integration Repository.

See: Uploading ILDT Files to Integration Repository, page 5-16.

4. (Optional) Users who have the Integration Administrator role can delete the custom integration interfaces if needed.

   Before starting to use a custom integration interface from the Integration Repository, users who have the Integration Administrator role can delete the custom interface if it is not yet generated or deployed as a web service. The administrators can first locate the custom interface from the Integration Repository user interface, and then click **Delete Interface** in the Overview tab of the custom interface details page.

   If a custom interface has been generated or deployed, it must be reset or undeployed to its initial state before it can be deleted. See: Deleting Custom Integration Interfaces, page 5-23.

5. All users can view the uploaded custom interfaces from the Integration Repository user interface.

6. (Optional) Users who have the Integration Administrator role then create necessary security grants for the custom integration interfaces if needed.

   This is achieved by first locating the custom interface from the Integration Repository, and then selecting methods contained in the selected custom interface before clicking **Create Grant**. The Create Grants page is displayed where the administrators can grant the selected method access permissions to a user, user group, or all users.

7. (Optional) Users who have the Integration Administrator role can generate SOAP services if the custom interfaces can be service enabled.

   This is achieved by first locating the custom interface, and then specifying the interaction pattern either at the interface level or the method level before clicking **Generate** in the selected custom interface details page. See: Generating Custom SOAP Web Services, page 5-25.

8. (Optional) Users who have the Integration Administrator role deploy the services from Oracle Integration Repository to the application server.

   To deploy generated SOAP services, the administrators must first select one authentication type (Username Token or SAML Token) for each selected service and then click **Deploy** in the selected interface details page. This deploys the generated service with 'Active' state to Oracle SOA Suite where Oracle E-Business Suite services can be exposed as standard services for service invocation at runtime. See: Deploying and Undeploying SOAP Custom Web Services, page 5-25.

   If the custom interfaces can be exposed as REST services, the administrators must enter a unique service alias for each selected custom interface, and specify the

desired service operations before deploying the service. Additionally, the administrators need to specify HTTP methods for the service operations contained in the selected interface if it is an interface type of PL/SQL, Java Bean Services, Application Module Services, or Business Service Object.

> **Note:** Although open interface tables and views can be exposed as REST services, custom open interface tables and custom open interface views are not supported in this release.

REST services are deployed to an Oracle E-Business Suite environment. For more information on how to deploy custom REST services, see Deploying Custom REST Web Services, page 5-27.

To better understand how to use Integration Repository Parser to validate and upload annotated custom interface definitions to Integration Repository, as well as perform administrative tasks on these uploaded custom integration interfaces, the following topics are discussed in this chapter:

- Setting Up and Using Integration Repository Parser, page 5-6
- Administering Custom Integration Interfaces and Services, page 5-21

For information on how to create and annotate custom integration interfaces, see Creating and Annotating Custom Integration Interfaces, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

# Setting Up and Using the Integration Repository Parser

## Setup Tasks

Integration Repository Parser is a standalone design-time tool used by the integration administrator to validate annotated custom interface definitions against the annotation standards and generate an Integration Repository loader file (iLDT). The generated iLDT files are uploaded to the Integration Repository using the FNDLOAD command so that the custom interfaces can be searched, generated, and deployed from the Integration Repository user interface.

> **Note:** Integration Repository Parser does not support the integration interfaces registered under custom applications.

Before running the Integration Repository Parser, you need to install `Perl` modules with the following steps:

> **Note:** It is required to obtain a native C compiler for the platform and

operating system version that you are running on to build the `Perl` modules. The following are the minimum versions of compilers certified for Oracle E-Business Suite platforms:

- Linux x86-64: Intel C/C++ Compiler (icc) version 7.1.032
- Linux x86-64: GNU Compiler Collection (GCC) version 4.1.2
- Oracle Solaris on SPARC (64-bit): Oracle Studio 12
- HP-UX Itanium: HP ANSI C B3910B A.0.06.05
- IBM AIX on Power Systems (64-bit): XL C Enterprise 8.0

Microsoft Windows platform is currently not supported in this release.

**Installing Perl Modules on Linux**

Perform the following steps to install Perl modules on Linux:

1. Establish the Oracle E-Business Suite application environment.

   From the Oracle E-Business Suite APPS_BASE, establish the run file system APPL_TOP environment by running the EBSapps.env script.

2. Download Patch 13602850 (p13602850_R12_GENERIC.zip) into a temporary directory.

3. Run the following command:

   ```
   perl $FND_TOP/patch/115/bin/IREPParserSetup.pl
   ```

4. When prompted, provide the temporary directory path where patch p13602850_R12_GENERIC.zip is located.

   ```
   Enter patch 13602850 location (<patch
   location>/p13602850_R12_GENERIC.zip file should exist and the
   default location is default location is <current directory>):
   ```

   The `IREPParserSetup.pl` script will display the status of each step on the terminal. At the end, you may see messages like:

   ```
   [Final Status] Integration Repository parser setup completed
   successfully [Info] For more information check <current
   directory>/IREPParserSetup.log
   ```

**Installing Perl Modules on Oracle Solaris, AIX, and HP-UX Itanium**

Perform the following steps to install `Perl` modules on Oracle Solaris, AIX, and HP-UX Itanium:

1. Establish an Oracle E-Business Suite application environment.

From Oracle E-Business Suite APPS_BASE, establish the run file system `APPL_TOP` environment by running the `EBSapps.env` script.

2. On both the run and patch file systems, locate the `Perl` configuration files that need to be modified and back up these files.

   For example, on Oracle Solaris, the `Config.pm` is located in the following directory:

   ```
   $FMW_HOME/webtier/perl/lib/5.10.0/sun4-solaris-thread-multi-64
   ```

3. On both the run and patch file systems, modify the `Perl` configuration file `Config.pm` to point to the `Perl` directory in `$FMW_HOME/webtier`.

   For example, on Oracle Solaris, these are the statements that need to be modified with the absolute path of `$FMW_HOME/webtier/perl`:

   > **Note:** `<FMW_HOME>` is the value of `$FMW_HOME`.

   - `archlibexp =>relocate_inc('<FMW_HOME>/webtier/perl/lib/5.10.0/sun4-solaris-thread-multi-64')`
   - `privlibexp =>relocate_inc('<FMW_HOME>/webtier/perl/lib/5.10.0')`
   - `sitearchexp =>relocate_inc ('<FMW_HOME>/webtier/perl/lib/site_perl/5.10.0/sun4-solaris-thread-multi-64')`
   - `sitelibexp =>relocate_inc ('<FMW_HOME>/webtier/perl/lib/site_perl/5.10.0')`

4. If your system is on Oracle Solaris, modify the `Config.pm` and `Config_heavy.pl` files to point to the C compiler installed as a requirement of the Integration Repository Parser. For example:

   Config.pm

   ```
   cc =>'/opt/SunProd/studio12u3/solarisstudio12.3/bin/cc',
   ```

   ```
   libpth =>'/opt/SunProd/studio12u3/solarisstudio12.3/lib
   /opt/SUNWspro/WS6U1/lib/v9 /usr/lib/sparcv9
   /usr/ccs/lib/sparcv9 /usr/local/lib/usr/lib /usr/ccs/lib,
   ```

   Config_heavy.pl

   ```
   cc='/opt/SunProd/studio12u3/solarisstudio12.3/bin/cc'
   ```

   ```
   ld='/opt/SunProd/studio12u3/solarisstudio12.3/bin/cc'
   ```

5. Create a directory '`perl`' in `$APPL_TOP_NE` where the new `Perl` modules will be installed. For example:

   ```
   mkdir $APPL_TOP_NE/perl
   ```

   ```
   chmod 755 $APPL_TOP_NE/perl
   ```

6. On the run file system, set the following environment variables in the `APPL_TOP` environment:

   1. Prepend `PATH` with the path to the C compiler installed as a requirement of the Integration Repository Parser.

   2. Prepend `PERL5LIB` with `$FND_TOP/perl` and `$APPL_TOP_NE/perl` in that order.

      For example, export `PERL5LIB=$FND_TOP/perl:$APPL_TOP_NE/perl:$PERL5LIB`.

   3. Add `$FMW_HOME/webtier/lib` to `LIBPATH` if it is not present.

      For example, export `LIBPATH=$LIBPATH:$FMW_HOME/webtier/lib`.

   4. Set `$FMW_HOME/webtier` as `ORACLE_HOME`.

      For example, export `ORACLE_HOME=$FMW_HOME/webtier`.

   5. Prepend `LD_LIBRARY_PATH` with `$ORACLE_HOME/lib32` and `$ORACLE_HOME/lib`.

      For example, export `LD_LIBRARY_PATH=$ORACLE_HOME/lib32:$ORACLE_HOME/lib:$LD_LIBRARY_PATH`.

   6. Set `JAVA_HOME` to the JDK top directory.

      Obtain the path returned by 'which java' and set `JAVA_HOME` to the current JDK top directory.

      For example, on Oracle Solaris:

      ```
      which java
         /prod/EBS122/fs1/FMW_Home/jdk/jre/bin/java
      export JAVA_HOME=/prod/EBS122/fs1/FMW_Home/jdk
      ```

7. Download and unzip Patch 13602850 (`p13602850_R12_GENERIC.zip`) to a temporary area.

   Patch 13602850 contains the following `Perl` modules:

   - `Compress-Raw-Zlib-2.009`
   - `Compress-Zlib-2.009`
   - `Class-MethodMaker-1.12`

   Install these modules in the order shown above using the following commands:

   > **Note:** If `Perl` command is not found, invoke `Perl` in `$FMW_HOME/webtier/perl/bin/perl`.

After installing the `Compress-Raw-Zlib-2.009` Perl module but before installing `Compress-Zlib-2.009`, prepend `PERL5LIB` with `$APPL_TOP_NE/perl/lib/5.10.0/<platform thread-multi directory>`.

For example, on Oracle Solaris:

export `PERL5LIB=$APPL_TOP_NE/perl/lib/5.10.0/sun4-solaris-thread-multi-64:$PERL5LIB`.

1. `cd $APPL_TOP_NE/perl`
2. Copy the module to be installed into `$APPL_TOP_NE/perl`.

   For example: `cp -r /temp/Compress-Raw-Zlib-2.009`
3. `cd <Perl module name>`

   For example: `cd Compress-Raw-Zlib-2.009`
4. `perl Makefile.PL`

   **Note:** On HP-UX Itanium, the option `CC=cc` may be needed when installing `Compress-Raw-Zlib-2.009`. For example, `perl Makefile.PL CC=cc`.

   If errors occur, verify your setup and remove the Perl module being installed from `$APPL_TOP_NE/perl` before copying it into `$APPL_TOP_NE/perl` to try again.
5. `make`

   **Note:** If the 'cc' compiler is not found, verify the LD parameter in the `Makefile` that contains the correct path to the C compiler executable file.

   If the following warning appears on Oracle Solaris, replace `-xarch=v9` with `-m64` throughout the `Makefile`, and run `make` again.

   ```
   cc: Warning: -xarch=v9 is deprecated, use -m64
   to create 64-bit programs
   ```
6. `make install`

## Using the Integration Repository Parser

Once the Integration Repository Parser has been installed and set up properly, you can

run the parser to generate iLDT files and then upload the files to the Integration Repository if no error occurs.

> **Note:** For an object (or class) which is present in the Integration Repository, the Integration Repository Loader program reloads the new definition of that object ONLY if the new version is a later version than the current version present in the Integration Repository. If the new file version is the same or earlier than the current one in the repository, then the new file will not be uploaded.
>
> Therefore, before running the parser, you need to increment the Header version of the target source file so that the modifications to the object defined in the source file can take effect in the Integration Repository.

The following sections explain the use of Integration Repository Parser and FNDLOAD utilities in greater detail.

## Generating ILDT Files

**Prerequisites - Setting Up Environment Variables**

Before running the Integration Repository Parser to generate iLDT files, set the following environment variables which may affect parser operation:

1. From the Oracle E-Business Suite APPS_BASE, establish the run file system `APPL_TOP` environment by running the `EBSapps.env` script.

2. The following environment variables affect parser operation:

   - `LIBPATH`: Add the `$FMW_HOME/webtier/lib` to `LIBPATH` variable if it is not present. For example,

     export `LIBPATH=$LIBPATH:$FMW_HOME/webtier/lib`

   - `CLASSPATH`: It is used when parsing Java files. This is required to be properly set up (as if for a compile) when performing `-generate` with such files.

     If parser is not able to find a particular class, check for its availability in `CLASSPATH`.

     On a Linux machine, `CLASSPATH` can be set like `setenv CLASSPATH classpath1:classpath2`.

     For others, refer to your platform documentation on how to set `classpath` variable.

   - `JAVA_HOME`: It is used to find the Java runtime.

     If `JAVA_HOME` is not set, obtain the path returned by 'which java' from the `APPL_TOP` environment, and set `JAVA_TOP` to the JDK top directory. For

example,

- On AIX:

  export `JAVA_HOME=$COMMON_TOP/util/jdk32`

- On Oracle Solaris:

  export `JAVA_HOME=$COMMON_TOP/util/jdk`

- export `PERL5LIB=$APPL_TOP_NE/perl/lib/5.10.0:$APPL_TOP_NE/perl/lib/site_perl/5.10.0:$FND_TOP/perl:$PERL5LIB`

- For HP-UX Itanium Only

  Prepend `LD_LIBRARY_PATH` with `$FMW_HOME/webtier/lib` as follows:

  export `LD_LIBRARY_PATH=$FMW_HOME/webtier/lib:$LD_LIBRARY_PATH`

**Running the Integration Repository Parser**

To generate an iLDT (`*.ildt`) file, run the Integration Repository Parser using the following syntax:

```
$IAS_ORACLE_HOME/perl/bin/perl $FND_TOP/bin/irep_parser.pl -g -v
-username=<a fnd username> <product>:<relative path from product
top>:<fileName>:<version>=<Complete File Path, if not in correct
directory>
```

Examples of generating iLDT files for custom PL/SQL APIs and custom composites of BPEL type:

- `$IAS_ORACLE_HOME/perl/bin/perl $FND_TOP/bin/irep_parser.pl -g -v -username=sysadmin fnd:patch/115/sql:SOATest1S.pls:12.0=SOATest1S.pls`

- `$IAS_ORACLE_HOME/perl/bin/perl $FND_TOP/bin/irep_parser.pl -g -v -username=sysadmin fnd:<path>:ONT_POI_R121XB7A.bpel:12.0=<Path>/ONT_POI_R121XB7A.bpel`

  **Note:** If an error message "Java runtime not found" appears while running the Integration Repository Parser, then set the JRE location to variable `OA_JRE_TOP`. JRE location could be located at `$JAVA_HOME/jre`, If `JAVA_HOME` is not set, source `$FMW_HOME/wlserver_10.3/server/bin/setWLSEnv.sh` file.

While running the parser, you need to pay attention to any error messages on the console. These errors would be due to incorrect annotation or some syntax errors in the annotated file. Ensure that the annotations are correct and the file has proper syntax.

If no error occurs in the annotated interface file, an iLDT (`*.ildt`) file would be

generated. This generated iLDT file needs to be uploaded to the Integration Repository.

See: Uploading ILDT Files to Integration Repository, page 5-16.

### Integration Repository Parser (irep_parser.pl) Usage Details

The usage for the Integration Repository Parser can be seen from the command prompt using the `-manual` option:

```
$IAS_ORACLE_HOME/perl/bin/perl $FND_TOP/bin/irep_parser.pl -
manual
```

**Name** `irep_parser.pl` Interface Repository Annotation Processor

**Synopsis** `irep_parser.pl [-verbose] [-logfile=file ? -append-logfile=file] [-generate] [-force] [-outdir=directory] [-java-source=version] [-cache-java=oper] [-cache-file=file] [-imports=file] [-username=username] <filespec>...`

**Description** The `irep_parser` reads interface annotation documentation in program source files and validates it according to its file type.

If the `-generate` flag is supplied (and other conditions met), then it will generate iLDT files. For more information, see `-generate` option, page 5-14.

Any validation errors will be reported, usually along with file name and line number, like the result of `grep -n`.

**File Types**

The `irep_parser` can handle almost all types of application source files. While validating the annotated files against the annotation standards of the supported interface types, files that do not match will be ignored.

Here is the list of supported file types:

> **Note:** Integration Repository Parser supports custom interface definitions for XML Gateway Map, Business Event, PL/SQL, Concurrent Program, Business Service Object, Java APIs, Java Bean Services, Application Module Services, and Composite Service for BPEL type.
>
> Custom interface types of EDI, Open Interface Tables, Interface Views, and Java APIs for Forms interfaces are not supported in this release.

- `.java`: All Java files are completely parsed.
- `.p(kh/ls)`: PL/SQL package specifications are processed.

  If and when a package body is detected, the parser aborts processing and the file is ignored.
- `.ldt`: It processes the LDT file for annotated concurrent programs. Most LDT files will fail and be ignored right away because they are not concurrent program loader

files (i.e. not created with `afcpprog.lct`).

- `.xgm`: It processes the XML Gateway map file, looking for an annotated map.
- `.xml`: It processes the XML file, scanning for signature contents indicating various kinds of Business Service Object data since the filename pattern is generic.
- `.wfx`: It processes the Business Event file, looking for annotated events.

**Files Specifications**

Argument `filespec` tokens have the following formats:

- `pathname`: A simple `pathname` argument directly indicates the file to be processed. Since path information is not included, the output iLDT can not be generated. For example, only validation is supported. See `-development` flag, page 5-15 (This is backward compatible with previous validation only usage.)
- `product:relative_path[:name[:version]]=pathname`: Specify the product and relative path from product top (and optionally file name and version) in addition to the physical location of the file to process.

  Please note that the source file information on the left-hand side of the "`=`" sign is imported verbatim into the output iLDT, and otherwise not examined. The `pathname` on the right-hand side must refer to a real file, which can be located anywhere.

  The `product` and `relative_path` correspond to file location on `APPL_TOP`.

**Options**

Options can be abbreviated by the smallest significant number of characters. Often this can be just the first character. Options cannot be combined. Here are the supported options:

- `-generate`: It generates iLDT (Interface Repository Seed Data) files. The file is created in either the current directory or the directory designated by `-outdir`.

  The generated file name is derived from the file name by replacing all periods with underscores, and then appending the suffix "`.ildt`".

  > **Note:** Use of the `-generate` flag requires that the command line filespecs to have (at least) the source product and path. For more information, see `prod:path[:name[:version]]=pathname`, page 5-14 and the `-development` flag, page 5-15.

- `-force`: If the `-generate` flag is used to request iLDT generation, and if the file is an incorrect file type for annotations or has no significant annotation contents (no annotation at all, or no `@rep:scope` tag in any primary-level annotation), then an

empty file is created anyway. If a file of the same name existed from a previous run, it is forced to be overwritten with a zero-length file.

The net effect is that only files that had actual errors (parsing, validation, and incomplete for generation) will not be represented in the creation of (at least) in an empty iLDT file.

- `-development`: It is a special flag for developers to quickly verify syntax of annotations in a file. It is equivalent to using both `-generate` and `-verbose` flags with sample values of fields, such as 'product', 'relative path from product top' and 'version'. For example, `-d TestFileName` is equivalent to `-g -v nul: relative/path/unknown:TestFileName:1.0=TestFileName`.

  This allows you to generate test iLDTs using a simple list of filenames.

- `-outdir=directory`: It designates an alternate directory (other than the working directory) for generated output to be placed in.

- `-username=username`: A valid FND user name (other than the default SEED user name) which marks this interface as custom service.

  If tag `-username` is missed, it is considered as a seeded interface. A custom interface is identified on the Integration Repository user interface by the label 'Custom' and can be searched by selecting 'Custom' in the Interface Source field after clicking **Show More Search Options** in the Search page.

- `-logfile=file`: It writes all verbose tracing and validation error messages in a log file instead of printing to standard output. It is mutually exclusive with `-append-logfile`.

- `-append-logfile=file`: It is similar to `-logfile`, append all verbose tracing and validation error messages in a log file instead of printing to standard output. It is mutually exclusive with `-logfile`.

- `-verbose`: It provides chatty information about files processed and other internals, non-fatal warning messages, and so on. This is in addition to any error messages generated.

  It is useful for querying the parser version, if it's used without any filespec arguments.

- `-java-source=version`: It informs the parser what language version (via JDK version number) to support for Java parses. A minor change was introduced in 1.4 (the assert facility), and major changes were introduced in 1.5 (generics, enhanced for loop, autoboxing/unboxing, enums, varargs, static import and annotations). If it is not supplied, then 1.5 is assumed.

**Return Value**

The parser will return an exit value of 0 if no errors occurred during processing.

Otherwise, it will return a count of the number of files that had errors.

Files with incomplete information for generation (class resolution) are considered errors only if the `-generate` flag is used.

**Quick Validation Examples**

Use the following statements in validating annotation in PL/SQL specification files during development:

- `$IAS_ORACLE_HOME/perl/bin/perl $FND_TOP/bin/irep_parser.pl *s.pls`
- `$IAS_ORACLE_HOME/perl/bin/perl $FND_TOP/bin/irep_parser.pl -v -g itg:patch/115/sql:12.0=fndav.pls`

## Uploading ILDT Files to Integration Repository

After validation is completed and iLDT files are generated, the integration administrator can upload the generated iLDT files to the Integration Repository using the FNDLOAD command. The custom interfaces can be displayed in the repository and exposed to all users.

**Manual Steps for Uploading the iLDT File**

Perform the following steps to upload the iLDT file to the Integration Repository:

1. Log in to the Oracle E-Business Suite Release 12 instance.
2. Set the Oracle E-Business Suite application environment.

   From the Oracle E-Business Suite APPS_BASE, establish the run file system `APPL_TOP` environment by running the `EBSapps.env` script.
3. Use the following command to upload the iLDT file:

   ```
   $FND_TOP/bin/FNDLOAD <APPS username> 0 Y UPLOAD
   $fnd/patch/115/import/wfirep.lct <ildt file>
   ```

   ```
   ORACLE Password:
   ```

   Examples of uploading iLDT files for custom PL/SQL APIs and custom composites of BPEL type:

   - ```
     $FND_TOP/bin/FNDLOAD apps @isg122d 0 Y UPLOAD
     $FND_TOP/patch/115/import/wfirep.lct SOATest1S_pls.ildt
     ```

     ```
     ORACLE Password: password
     ```
   - ```
     $FND_TOP/bin/FNDLOAD apps @$TWO_TASK 0 Y UPLOAD
     $FND_TOP/patch/115/import/wfirep.lct .
     /ONT_POI_R121XB7A_bpel.ildt
     ```

     ```
     ORACLE Password: password
     ```
4. Pay attention to any error messages in the generated log file. Error messages mostly

would be due to incorrect database connect string or incorrect `lct` file.

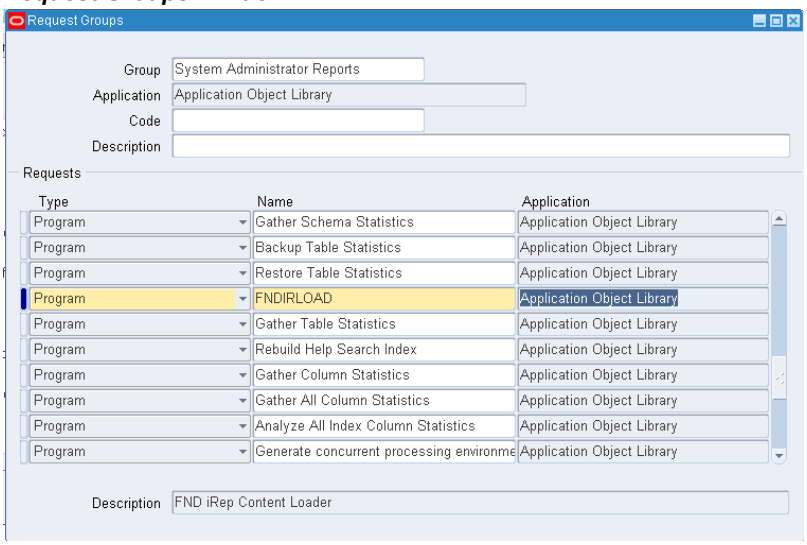
Look for string "Concurrent request completed successfully" to determine whether the iLDT file was correctly uploaded.

5. For Business Service Object only - submit a concurrent program called FNDIRLOAD which loads all the iLDT files related to Business Service Object interfaces present on various product tops of the instances.

   **Note:** Ensure that the FNDIRLOAD concurrent program is associated with the user who will run the concurrent request.

   For example, if the request will be run by a user who has the system administrator responsibility, FNDIRLOAD should be listed as part of the requests for System Administrator Reports group in the Request Groups window.

***Request Groups Window***




   If you cannot find FNDIRLOAD from the name list, use the following steps to register it with the system administrator responsibility.

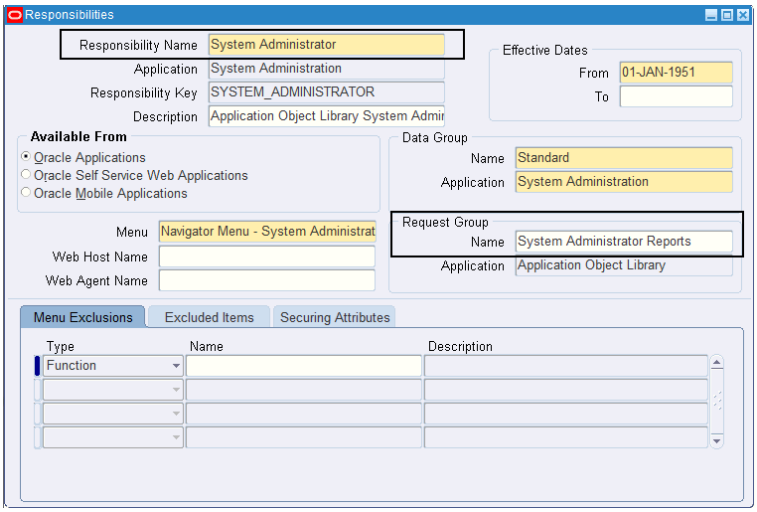
   1. Log in to Oracle E-Business Suite with the System Administrator responsibility. Select **System Administrator > Security > Responsibility > Define** from the navigation menu.

   2. In the Responsibilities window, locate 'System Administrator' as the value in the Responsibility Name field through a search.

      Ensure 'System Administrator Reports' is selected as the

Request Group Name.

***Responsibilities Window with Responsibility Name and Request Group Name Highlighted***



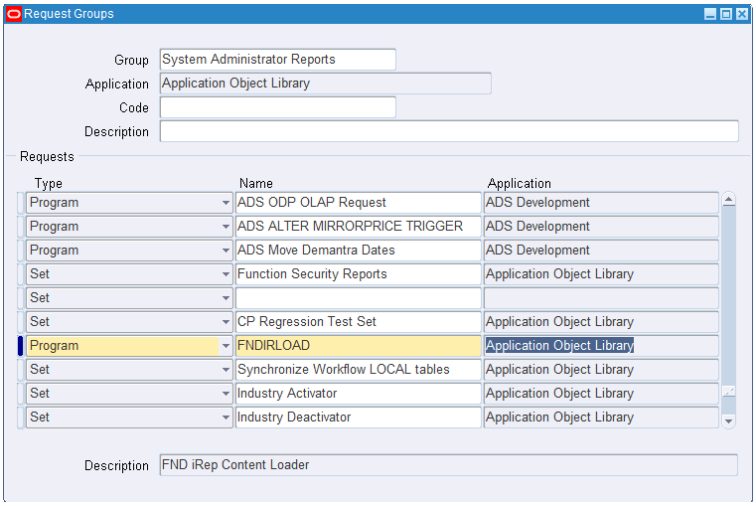

Save the change and close the window.

3. Select **System Administrator > Security > Responsibility > Requests** from the navigation menu.

   In the Request Group window, locate 'System Administrator Reports' as the value in the Group field through a search.

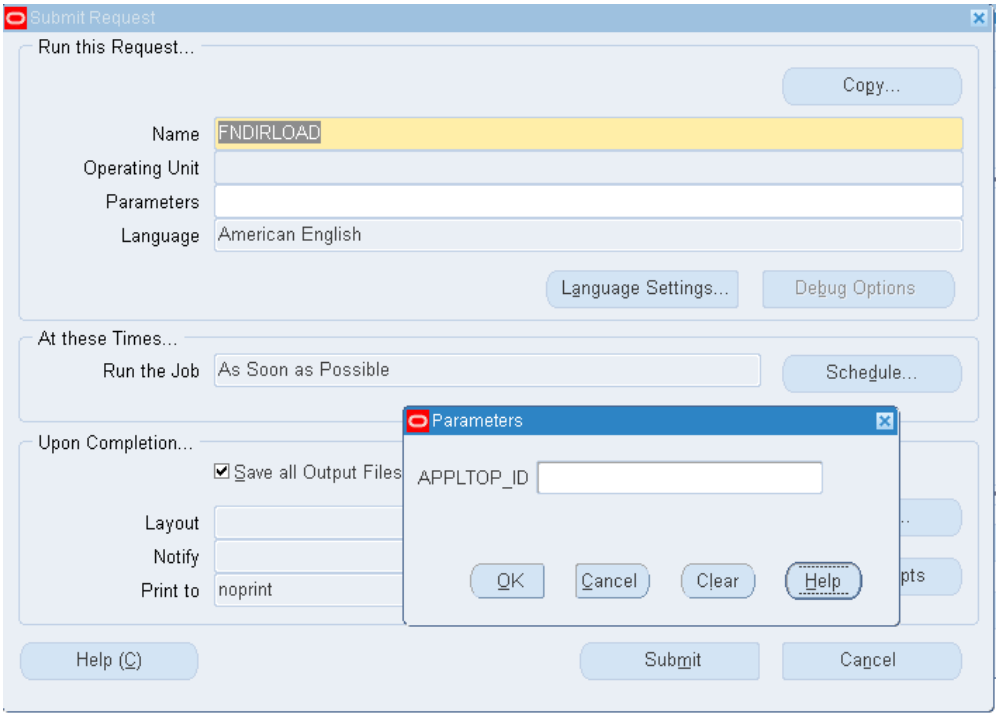
   In the Requests region, add FNDIRLOAD program to the list and save your entry.

***Request Groups Window***




In the Parameters window, enter an appropriate value for APPLTOP_ID.

***Submit Request Window with Parameters Pop-up Window***




**Note:** To obtain the APPLTOP_ID parameter value, your system administrator can run the following query:

```
SELECT max(appl_TOP_id)
FROM ad_appl_tops
WHERE active_flag = 'Y'
```

Click **Submit** to process the request.

Examine the request log file to see if any issues occur while running the concurrent request.

Once these annotated source files have been successfully uploaded, they will appear in the Integration Repository based on the interface types they belong to. The administrator can perform administrative tasks on these custom integration interfaces.

If the upload is for an updated version of iLDT file, after the upload the administrator needs to perform the following additional tasks to ensure successful invocation of the updated API included in the file:

- For SOAP services, stop and restart Oracle SOA Suite managed server where data sources are deployed.
- For REST services, clear existing cache, and stop and restart the `oafm` Oracle E-Business Suite managed server where data sources are deployed.

## Performing Administrative Tasks for Custom Integration Interfaces and Services

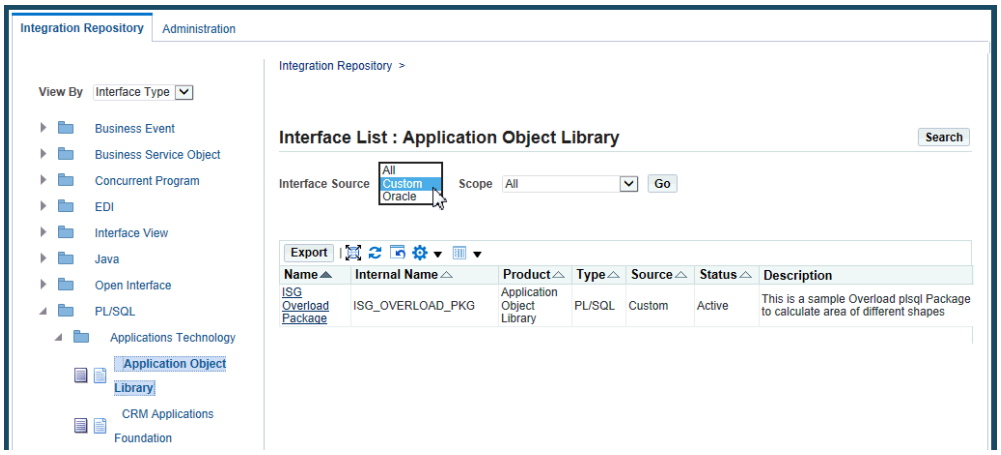
Custom integration interfaces are annotated based on Integration Repository annotation standards for the supported interface types. The behavior of these interfaces is the same as Oracle seeded interfaces except they are not native packaged, but custom ones. As a result, an integration administrator uses the same approach of managing native interfaces to manage custom interfaces and services.

**Viewing Uploaded Custom Integration Interfaces from the Integration Repository**

Before starting to perform any administrative task, the administrator needs to first locate a desired custom interface or service through either of the following ways:

- From the Interface List page, select 'Custom' from the Interface Source drop-down list along with a value for the Scope field to restrict the custom integration interface display. The search criteria 'Oracle' in the drop-down list is used for searching seeded interfaces.

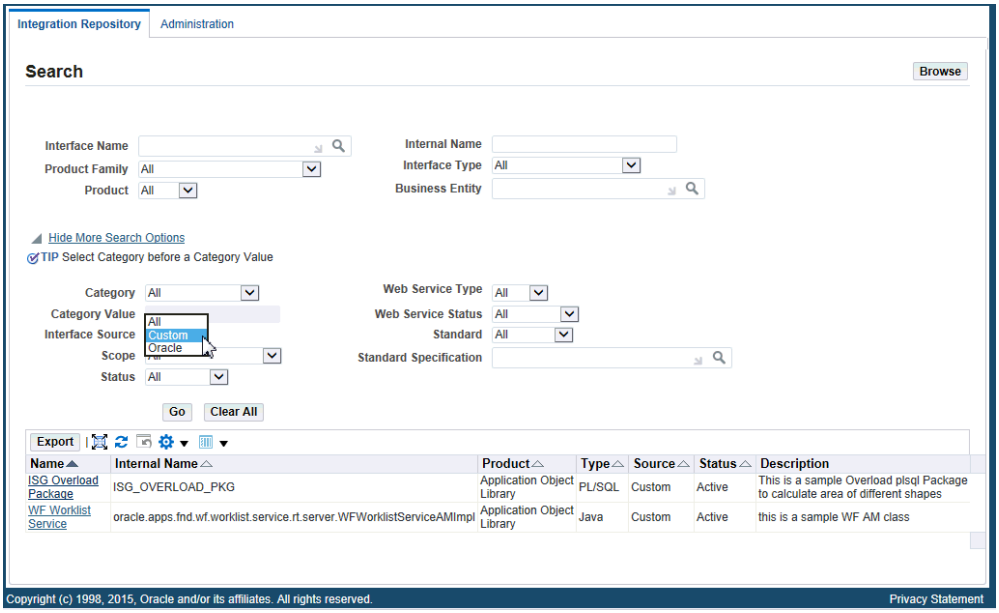
***Interface List Page with Interface Source "Custom" Selected***




- From the Search page, click **Show More Search Options** and select 'Custom' from the Interface Source drop-down list along with any interface type, product family, or scope if needed as the search criteria.

  For example, select 'Custom' as the Interface Source and 'PL/SQL' as the Interface Type to locate the custom interfaces for PL/SQL type.

***Search Page with Interface Source "Custom" Selected***




For more information on how to search for custom integration interfaces, see the *Oracle E-Business Suite Integrated SOA Gateway User's Guide*.

After locating a desired custom interface, the administrator can perform the following administrative tasks:

> **Important:** If you update a custom service, you must set the `ISG_CLEAR_JPUB_CACHE` property to reflect the changes in the updated custom service:
>
> - For a custom SOAP service, set `<SID>.ISG_CLEAR_JPUB_CACHE=YES` in `isg.properties`.
>
> - For a custom REST service, set `<SID>.ISG_CLEAR_JPUB_CACHE=YES` in `isgagent.properties`.
>
> Set this property to `Yes` only when there are updates in the custom service. Otherwise, set it to `No` or it should be commented out.

- **Managing Custom Integration Interfaces**
  - Deleting Custom Integration Interfaces, page 5-23

- **Managing Custom Web Service Lifecycle Activities**

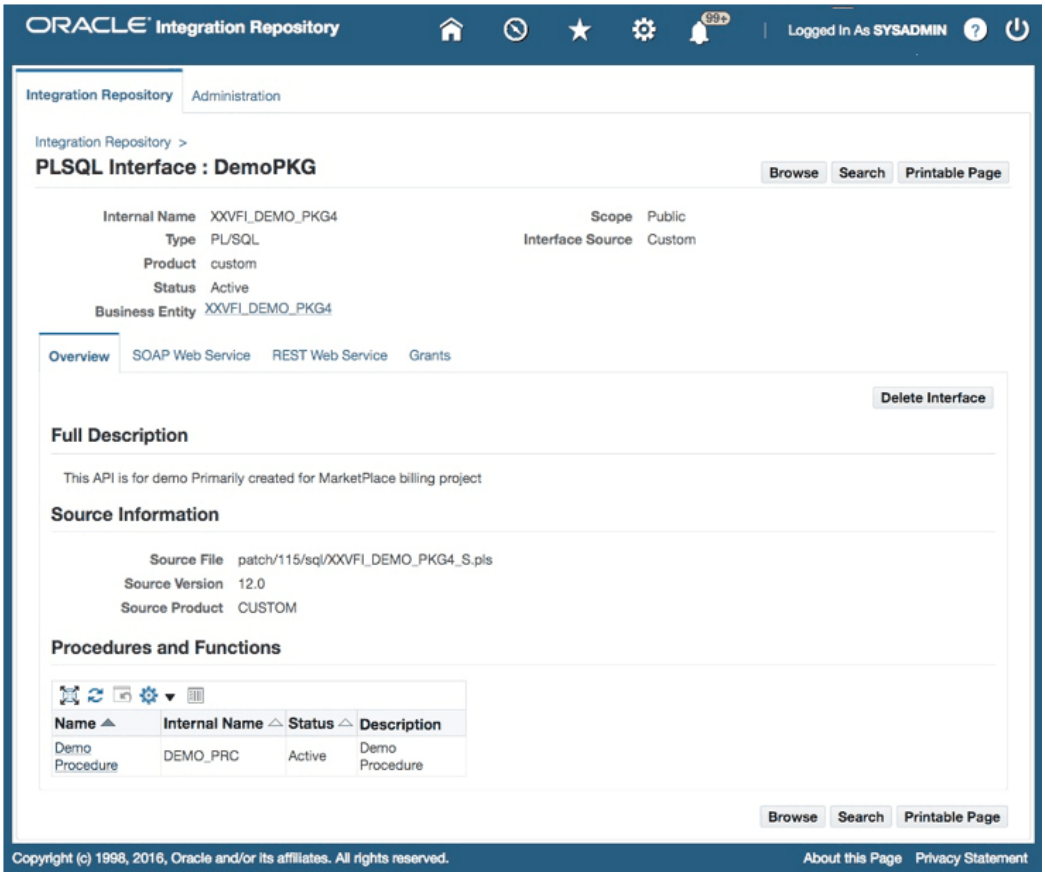
  *For Custom SOAP Web Services*

## Deleting Custom Integration Interfaces

Once a custom integration interface is validated and uploaded to the Integration Repository, integration administrators can delete the custom interface from the repository if the custom interface is not yet generated or deployed and it is no longer used or needed.

To delete a custom interface, first locate the custom interface from the repository and then click **Delete Interface** in the Overview tab of the selected custom interface details page. This action removes the selected custom interface from the integration repository.

*Overview Tab with the "Delete Interface" Button Shown for a Custom Interface*




If a custom interface has been generated or deployed, it must be reset or undeployed to its initial state ('Not Generated' for a SOAP service or 'Not Deployed' for a REST service) before it can be deleted through the Overview tab. Otherwise, a warning message appears indicating that you cannot delete a generated or deployed service.

For information on resetting a SOAP service, see Resetting SOAP Web Services, page 3-15. For information on undeploying a REST service, see Undeploying REST Web Services, page 3-46.

## Managing Security Grants for SOAP Services Only

To let appropriate users use these newly uploaded custom integration interfaces, the administrators can select one or more methods contained in a given custom interface and then grant the selected method access permissions to a user, user group, or all users. The administrators can revoke existing grants by removing the privileges from the grantee who can be a specific user, user group, or all users if needed.

For information about managing grants for interfaces with the support for SOAP services only, see Managing Security Grants for SOAP Web Services Only, page 3-21.

## Generating Custom SOAP Web Services

Once custom integration interfaces have been uploaded to Oracle Integration Repository, an integration administrator or an integration developer can transform these interface definitions into WSDL descriptions if the interface types they belong to can be service enabled.

To generate a web service, the administrator must first locate a custom interface, and then specify the interaction pattern either at the interface level or the method level before clicking **Generate** in the interface details page.

If the web service has been successfully generated, a WSDL link appears along with the 'Generated' web service status information displayed in the Web Service region (or the SOAP Web Service tab if the interface can be exposed as both SOAP and REST services). The selected interaction pattern information ('Synchronous', 'Asynchronous', or both Synchronous and Asynchronous) for the selected custom service is also displayed.

For detailed information on how to generate SOAP services on native integration interfaces, see Generating SOAP Web Services, page 3-4.

## Deploying and Undeploying Custom SOAP Web Services

Once a web service has been successfully generated for a custom interface, like native packaged interfaces, the administrator will perform the same deployment activity to deploy the generated service to an Oracle SOA Suite WebLogic environment with Active state. Before deploying the custom service, the administrator must select one authentication type to authenticate the web service.

The administrator can undeploy the service if needed.

> **Note:** Similar to the native Oracle E-Business Suite services, the deployed WSDL URL for the custom service shows the physical location of service endpoint where the service is hosted in `soa-infra` in this release. If your system is upgraded from a previous Oracle E-Business Suite release, after the upgrade to Release 12.2, the deployed WSDL URL information for the custom service has already been changed. Therefore, you may need to replace it with the new WSDL URL and service location or address accordingly in web service clients while invoking the deployed custom service.

For detailed information on how to deploy or undeploy SOAP web services, see Deploying and Undeploying SOAP Web Services, page 3-11.

## Resetting Custom SOAP Web Services

Once a custom service has been successfully generated or deployed, **Reset** appears in the Web Service region (or the SOAP Web Service tab if the interface can be exposed as

both SOAP and REST services) allowing you to reset the 'Generated' or 'Deployed' web service status to its initial state - 'Not Generated' if needed. This feature clears up the custom service artifact for a given service regardless of its current state.

For more information, see Resetting SOAP Web Services, page 3-15.

## Retiring Custom SOAP Web Services

When a custom service has been successfully deployed to Oracle SOA Suite with active state, this deployed custom service is ready to accept new requests.

The administrator can change the active state of a deployed custom service by clicking **Retire** in the Web Service region (or the SOAP Web Service tab if the interface can be exposed as both SOAP and REST services). This retires a deployed custom service and it will no longer accept new requests.

For a retired custom service, the administrator can activate the retired service so that it can become active again.

For more information on retiring SOAP web services, see Retiring SOAP Web Services, page 3-17.

## Activating Custom SOAP Web Services

For a custom service that has been retired, you can activate it by clicking **Activate** in the interface details page. This action allows a retired custom service to become active again.

For more information on activating web services, see Activating SOAP Web Services, page 3-18.

## Subscribing to Custom Business Events

Similar to the native business events, the administrator can subscribe to a custom business event by clicking **Subscribe** from the business event interface details page. Internally, an event subscription is created for that selected event with `WF_BPEL_QAGENT` Out Agent.

Once an event subscription for that custom event has been successfully created, **Unsubscribe** appears instead. Clicking **Unsubscribe** removes the event subscription from the `WF_BPEL_Q` queue.

For more information on subscribing to business events, see Subscribing to Business Events, page 3-20.

## Enabling Log Configurations and Viewing Log Messages for SOAP Services

In addition to managing design-time lifecycle activities through the Integration Repository user interface, the administrator can access the **Configuration** tab to perform additional administrative tasks:

- Enable design-time and runtime logs for SOAP service of an interface through the Log & Audit Setup Details page
  - Enabling Design-Time Log Configuration for SOAP Services, page 3-22
  - Adding a New Configuration, page 7-6
- Monitor and view design-time and runtime logs recorded for SOAP messages if the design-time log and runtime log are enabled for the SOAP service of a specified interface respectively
  - Viewing Design-Time Logs for SOAP Services, page 3-24
  - Viewing Service Processing Logs, page 7-15
  - Viewing SOAP and REST Request and Response Details, page 8-6

## Managing Security Grants for Custom REST Services

Similar to managing grants for the interfaces with the support for SOAP services only, the administrators can create grants by selecting one or more methods contained in a given custom interface and then grant the selected method access permissions to a user, user group, or all users.

Once an access permission to a procedure is authorized to a grantee, it grants the permission to access the associated SOAP and REST service operations simultaneously. For more information about managing grants for interfaces with the support for SOAP and REST services, see Managing Security Grants for SOAP and REST Web Services, page 3-48.

## Deploying Custom REST Web Services

After custom interfaces that can be exposed as REST services are uploaded to the Integration Repository, the administrator can deploy the custom REST services.

Before deploying a custom interface as a REST service, the administrator must specify service alias for the selected interface, select one or more methods from the Service Operations table, and ensure that at least one authentication type is selected in the REST Service Security region. Additionally, if the selected interface type is PL/SQL, Business Service Object, Java Bean Services, or Application Module Services, the administrator needs to specify HTTP verbs for desired methods contained in the selected interface before deployment.

If the service has been successfully deployed, the REST Service Status field is updated to 'Deployed' from 'Not Deployed' indicating that the deployed REST service is ready to accept new service requests.

For more information on deploying REST services, see Deploying REST Web Services,

page 3-30.

## Undeploying Custom REST Web Services

If a custom REST service has been successfully deployed to an Oracle E-Business Suite managed server, **Undeploy** appears in the REST Web Service tab. Undeploying a REST service not only brings the deployed REST service back to the Integration Repository, but also resets its status to its initial state - 'Not Deployed'.

For more information on undeploying REST services, see Undeploying REST Web Services, page 3-46.

## Enabling Log Configurations and Viewing Log Messages for REST Services

Similar to managing SOAP services, the administrator can perform the following additional tasks through the **Configuration** tab to enable and view log messages for REST services:

- Enable design-time and runtime logs for REST service of an interface through the Log & Audit Setup Details page
  - Enabling Design-Time Log Configuration for REST Services, page 3-52
  - Adding a New Configuration, page 7-6
- Monitor and view design-time and runtime logs recorded for REST messages if the design-time log and runtime log are enabled for the REST service of a specified interface respectively
  - Viewing Design-Time Logs for REST Services, page 3-53
  - Viewing Service Processing Logs, page 7-15
  - Viewing SOAP and REST Request and Response Details, page 8-6

## Viewing and Downloading Custom Composite Services

**Viewing Custom Composite Services**

To view a custom composite service, from the Search page select 'Composite' from the Interface Type field. Click **Show More Search Options** and select 'Custom' from the Interface Source drop-down list along with any product family or scope as the search criteria.

click a custom composite service from the search result to display the composite service details.

**Downloading Custom Composite Services**

The administrators can click **Download Service** in the interface details page to

download the relevant custom composite files aggregated in a .JAR file to your local directory.

For more information on how to view and download a composite service, see:

- Viewing Composite Services - BPEL, page 4-3
- Downloading Composite Services - BPEL, page 4-4

# 6

# Securing Web Services

## Overview

Security is the most critical feature that is designed to guard service content from unauthorized access.

To ensure the secure access to web service content, Oracle E-Business Suite integrated SOA Gateway uses the following security models to authenticate and authorize users to invoke a specific service or operation:


- Function Security and Data Security, page 6-1
- Role-Based Access Control (RBAC) Security, page 6-3
- Multiple Organization Access Control Security (MOAC Security), page 6-5
- WS-Service Security (Web Service Security), page 6-8


## Managing Function Security and Data Security

By leveraging Oracle User Management function security and data security, Oracle E-Business Suite Integrated SOA Gateway provides a security feature which allows authorized users to invoke certain methods of an integration interface exposed through Oracle Integration Repository. This security protects application data from unauthorized access or process of the methods or functions within an API.

Function security is the basic access control in Oracle E-Business Suite. It restricts user access to individual menus and menu options within the system. Regardless of the interface types, APIs enable you to insert and update data in Oracle E-Business Suite. When an API has the function security layer enforced, it implicitly restricts user access to the application.

Building on function security, data security provides another layer of security control. In other words, data security further restricts user access to the application at the data

level.

To allow users with appropriate privileges to access certain methods within an API, the concept of security grant is used to reinforce the security. This approach enables the data access privileges to be granted to a user, user group, or all users. To accomplish this goal, an integration administrator can select one or more methods contained in an API and then grant the selected methods to users.

The administrator can create security grants in the following ways:

- If an interface has only one method, then this single method should be selected in creating security grants.

  Concurrent Program interface type contains only one method. User security for XML Gateway interface is not managed in the Methods region, but in Oracle XML Gateway instead.

- If there is more than one method contained in an interface, then multiple methods can be selected simultaneously in creating security grants.

  Interface types containing multiple methods are PL/SQL, Business Service Object, Java interface, and Open Interface Table and View.

  > **Note:** For PL/SQL interfaces that can be service enabled with the support for both synchronous and asynchronous interaction patterns, the security grants given for the selected method names for a PL/SQL interface would be applicable to the generated synchronous and asynchronous operations of the service if both interaction patterns are selected during service generation.

**Creating Security Grants**

Only integration repository administrators (or users who have the Integration Repository Administrator role) can create security grants by authorizing the access permission of a selected interface method or procedure and function to an appropriate user, user group, or all users.

Security grants are managed in the Grants tab for the interface types that can be exposed as REST services. These interfaces include PL/SQL APIs, Concurrent Programs, Business Service Objects, Java Bean Services, Application Module Services, and Open Interface Tables and Views.

See: Managing Grants for Interfaces with Support for SOAP and REST Services, page 3-48.

> **Note:** XML Gateway interfaces can be exposed as SOAP services only, but the user security is managed in the Oracle XML Gateway user interface through the Trading Partner User Setup form. See: Managing XML Gateway User Security in the Trading Partner User Setup Form,

page 3-22.

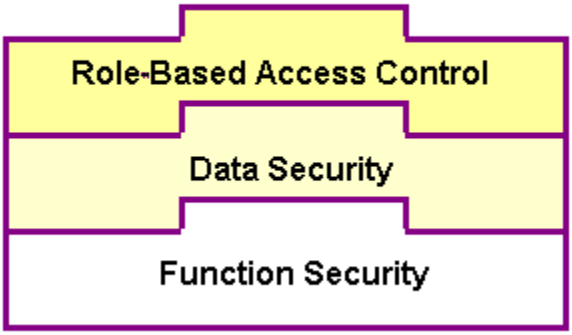
For more information on function security and data security, refer to the Oracle Application Object Library Security chapter, *Oracle E-Business Suite Security Guide*.

## Managing Role-Based Access Control Security

To allow only authorized users to perform certain administrative tasks, Oracle E-Business Suite Integrated SOA Gateway leverages Oracle User Management Role-Based Access Control (RBAC) security to build another layer of security. This RBAC security is enforced through user roles. As a result, whether a user can perform certain tasks, such as downloading a composite service from the application server, is determined by the roles granted to the user.

This approach builds upon Data Security and Function Security, but it goes beyond both of them.

***Role-Based Access Control Security***




As described earlier, function security is the base layer of access control in Oracle E-Business Suite. It restricts user access to individual menus and menu options within the system, but it does not restrict the access to the data contained within those menus. Data security provides access control on the application data, and the actions a user can perform on the data.

With RBAC, access control is defined through roles, and a role can be configured to consolidate the responsibilities, permissions, permission sets, and function security policies that users require to perform a specific function. This simplifies mass updates of user permissions because changes can be done through roles which will inherit the new sets of permissions automatically. Based on the job functions, each role can be assigned a specific permission or permission set if needed. For example, an organization may include 'Analyst', 'Developer', and 'Administrator' roles. The 'Administrator' role

would include a permission set that contains all administrative related tasks or functions allowing the administrator role to perform a job function while the Analyst and Developer roles may not have the access privileges.

## Role-Based Access Control (RBAC) Security for Oracle E-Business Suite Integrated SOA Gateway

In Oracle E-Business Suite Integrated SOA Gateway, each administrative function is considered as a permission. Relevant permissions are grouped into a permission set that will then be associated with appropriate function roles and assigned to appropriate users through security grants.

Oracle E-Business Suite Integrated SOA Gateway uses the following seeded permission sets to restrict administrative privileges only to authorized users:

- Integration Administrator Permission Set (FND_REP_ADMIN_PERM_SET)
- Integration Repository Download Composite Service (FND_REP_DOWNLOAD_PERM_SET)

### Integration Administrator Permission Set

The Integration Administrator Permission Set (FND_REP_ADMIN_PERM_SET) contains almost all administrative tasks performed by the users who have the Integration Administrator role. It consists of the following administrative permissions:

***Integration Administrator Permission Set***

| Privilege | Permission | Permission Display Name |
|---|---|---|
| Generate/Regenerate | FND_REP_GENERATE | Generate Web Service |
| Deploy | FND_REP_DEPLOY | Deploy Web Service |
| Undeploy | FND_REP_UNDEPLOY | Undeploy Web Service |
| Subscribe to Agent | FND_REP_SUBSCRIBE | Subscribe to Agent |
| Create Grants | FND_REP_METHOD_GRNT | Grant access privileges to methods |

### Integration Repository Download Composite Service Permission Set

Users with an appropriate privilege can download composite services and that privilege is associated with a permission set called Integration Repository Download Composite Service Permission Set (FND_REP_DOWNLOAD_PERM_SET) which is

separated from the Integration Administrator Permission Set described earlier. This approach allows the download feature to be granted separately to users through the Integration Administrator role, the Integration Developer role, or the Integration Analyst role if necessary.

***Integration Repository Download Composite Service Permission Set***

| Privilege | Permission | Permission Display Name |
|---|---|---|
| Download Composite Service | FND_REP_DOWNLOAD_CS | Download Composite Service |

## Managing MOAC Security

Multiple organizations can be sets of books, business groups, legal entities, operating units, or inventory organizations. You can define multiple organizations and the relationships between them in a single installation of Oracle E-Business Suite.

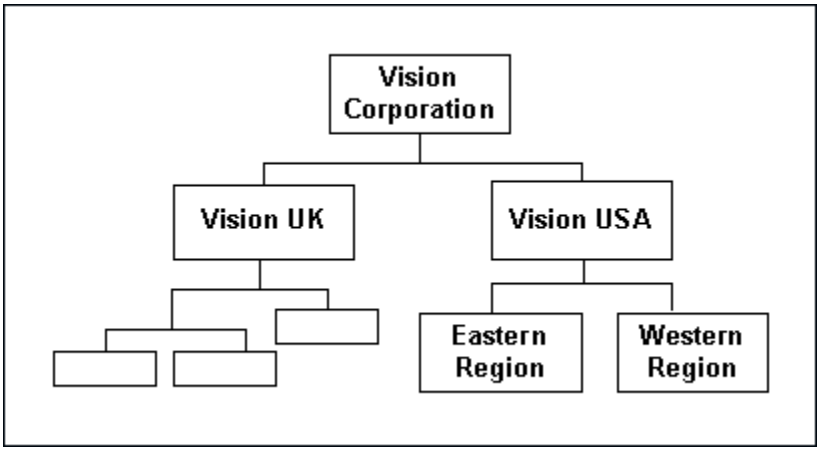
Oracle E-Business Suite Integrated SOA Gateway leverages the MOAC security feature to ensure that only authorized users have data access privilege within an operating unit.

With MOAC, a system administrator can predefine the scope of access privileges as a security profile, and then use the profile option *MO: Security Profile* to associate the security profile with a responsibility. By using this approach, multiple operating units are associated with a security profile and the security profile is assigned to a responsibility. Therefore, through the access control of security profiles, users can access to data in multiple operating units without changing responsibilities.

For example, a sales company consists of USA and UK operating units; the USA operating unit has Western Region Sales and East Region Sales. Sales managers are responsible for both USA and UK sales. Supervisors are responsible for either USA or UK. Sales representatives are only responsible for their designated sales regions.

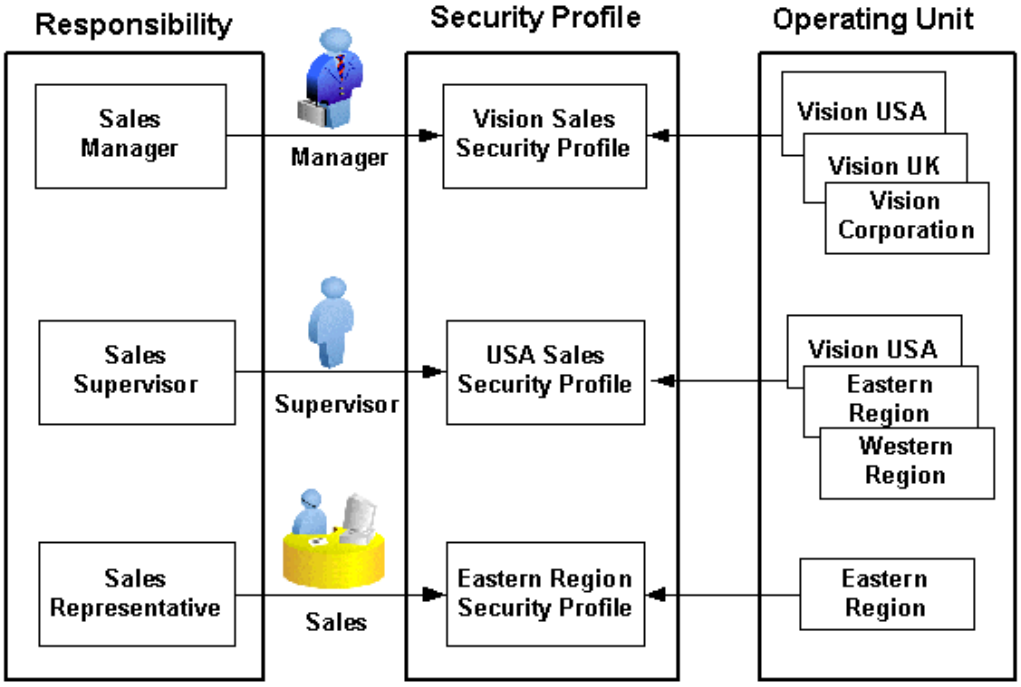
The following diagram illustrates the Sales organization hierarchy:

***Sales Organization Hierarchy***




To secure sales data within the company, relevant operating units can be associated with predefined security profiles. For example, all sales data access privileges are grouped into the Vision Sales security profile. A USA Sales security profile is for USA related data, and a regional security profile is for designated regional data. The system administrator can associate these security profiles containing multiple operating units with users through appropriate *responsibilities*. Therefore, sales supervisors can easily access sales data in the Eastern or Western region without changing their responsibilities. The following diagram illustrates the relationship between security profiles, responsibilities, and operating units for this sales company:

*Relationship Diagram Between Security Profiles, Responsibilities, and Operating Units*




**Responsibility Determines Operating Units**

Because responsibilities are associated with security profiles that are linked to operating units, your responsibility is the key to determine which operating units you will have the access privileges.

1. When integrating with Oracle E-Business Suite using PL/SQL and Concurrent Program interfaces, applications context values passed in `SOAHeader` elements for SOAP requests are Responsibility, RespApplication, SecurityGroup, NLSLanguage, and Org_Id. The same context values are passed for PL/SQL and Java Bean Services in `RESTHeader` element as part of the HTTP body for REST requests.

   For integrating with Oracle E-Business Suite using Business Service Object interfaces, applications context values passed in `ServiceBean_Header` elements for SOAP requests are RESPONSIBILITY_NAME, RESPONSIBILITY_APPL_NAME, SECURITY_GROUP_NAME, NLS_LANGUAGE, and ORG_ID.

2. MOAC setup is done based on the RespApplication or RESPONSIBILITY_APPL_NAME for Business Service Object interfaces to which the user belongs. If Org_Id is passed, the Organization access would be set to the passed Organization.

3. If the NLS Language element is specified, SOAP requests can be consumed in the language passed. All corresponding SOAP responses and error messages can also

be returned in the same language. If no language is identified, then the default language of the user will be used.

For more information on multiple organizations setup and implementation, see the *Oracle E-Business Suite Multiple Organizations Implementation Guide*.

## Managing Web Service Security

Web service security (WS-Security) is a specification to enable applications to conduct secure message exchanges. It proposes a standard set of extensions that can be used when building secure web services to implement message content integrity and confidentiality. It also provides support for multiple security tokens, the details of which are defined in the associated profile documents.

To secure web service content and authenticate service operation, Oracle E-Business Suite Integrated SOA Gateway supports the following authentication security models for inbound service requests:

- For SOAP Services
  - UsernameToken Based Security, page 6-10
  - SAML Sender-Vouches Token Based Security, page 6-12

  At design time, an integration administrator must select one authentication type before deploying a service. If no authentication type is identified for the service, then a validation error occurs.

  If the authentication type of a deployed SOAP service needs to be changed, the administrator must first undeploy the SOAP service, make appropriate changes, regenerate the SOAP service, and then deploy it again. For more information on how to deploy and undeploy SOAP services, see: Deploying and Undeploying SOAP Web Services, page 3-11.

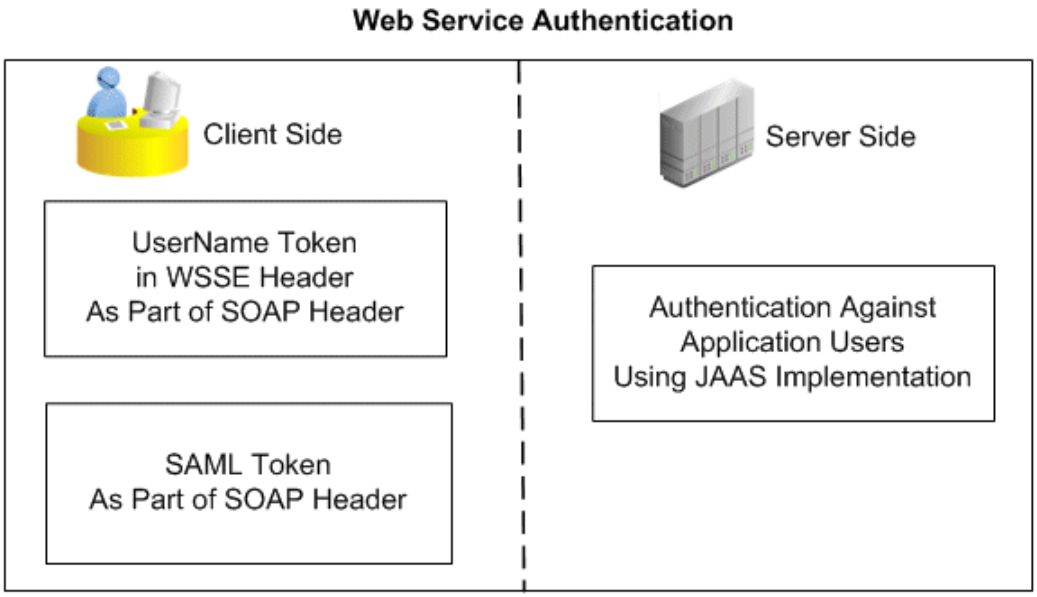
- For REST Services
  - HTTP Basic Authentication, page 6-15
  - Token Based Authentication, page 6-16

  All REST services are secured by either HTTP Basic Authentication (user name and password) or Token Based Authentication (user name and a valid token, such as Oracle E-Business Suite session ID).

**Subject Authentication to Establish User's Identity**

At runtime, when SOAP requests are received through Oracle SOA Suite for the deployed SOA Composites in an Oracle WebLogic managed server, each message is authenticated, depending on the selected authentication type, by a JAAS (Java Authentication and Authorization Service)-based login module for Oracle E-Business

Suite.




> **Note:** JAAS (Java Authentication and Authorization Service) is a Java security framework that can be used for **authentication of users** (user login) to securely determine who is currently invoking Java code, and for **authorization of users** to ensure that they have appropriate access control privileges required to access or perform certain operations.
>
> To authenticate users, the JAAS-based login module for Oracle E-Business Suite will be deployed into the Oracle WebLogic Server containing Oracle SOA Suite.

For REST services, when users are authenticating based on provided user name and password in REST requests, security Login service is used to validate user credentials and return a unique access token (such as Oracle E-Business Suite session ID). The token may be sent to LoginModule and used in subsequent requests for token based authentication.

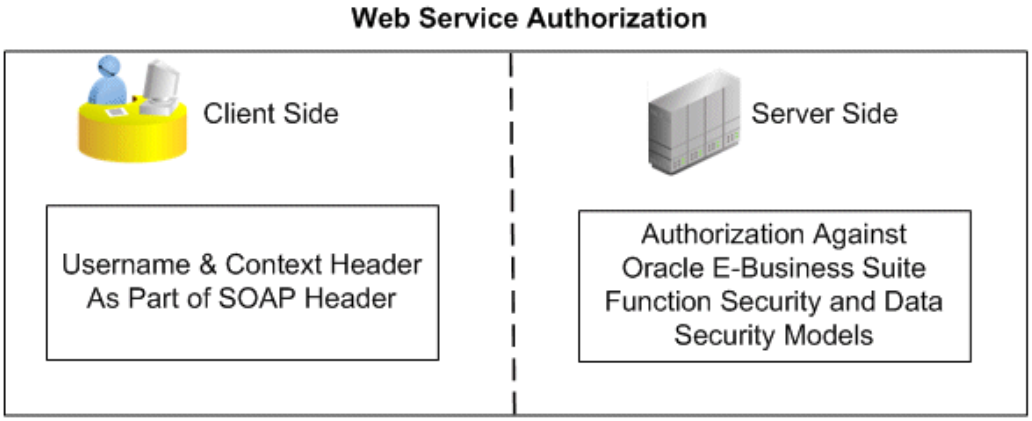
**Subject Authorization to Verify Access Privileges**

At design time, users are given appropriate privileges to access certain functions or APIs through security grants and RBAC-based function security.

***Authorization for SOAP Services***

At runtime, SOAP message header information is used to determine whether the current context has access to the operation that is invoked. For example, Oracle E-Business Suite application context contains many crucial elements that are used in passing values required in proper functioning of Oracle E-Business Suite services. This context header information is required for an API transaction or a concurrent program in order for an Oracle E-Business Suite user who has sufficient privileges to run the

program.




The following code snippet shows the sample header of application context:

```
<soapenv:Header>
 ..
 <!--wsse Header-->
 <fnd:SOAHeader>
 <fnd:Responsibility>SYSTEM_ADMINISTRATOR</fnd:Responsibility>
 <fnd:RespApplication>FND</fnd:RespApplication>
 <fnd:SecurityGroup>STANDARD</fnd:SecurityGroup>
 <fnd:NLSLanguage>AMERICAN</fnd:NLSLanguage>
 <fnd:Org_Id>204</fnd:Org_Id>
</fnd:SOAHeader>
</soapenv:Header>
```

For more information about SOAP header elements used for authorization, see SOAP Header for Application Context, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

***Authorization for REST Services***

After authentication, the LoginModule verifies the authenticated user's role and access privilege and then authorizes the authenticated user of accessing or invoking the underlying API, only if the user has the required privilege.

## UsernameToken Based Security

In the UsernameToken based security, the user name and password sent in the SOAP header for authentication is associated with the user created in Oracle E-Business Suite.

User name is a clear text; password is the most sensitive part of the UsernameToken profile. In this security model, the supported password type is plain text password (or PasswordText).

> **Note:** The PasswordText password type is the password written in clear text. SOAP requests invoking the web services should include security header consisting of a user name and plain text password. The

password received as part of the SOAP request at runtime will be validated against the encrypted password stored in Oracle E-Business Suite. After validation, the plain text password from the SOAP request will be discarded.

At runtime, SOAP request messages received through Oracle SOA Suite are passed on to a JAAS based login module for Oracle E-Business Suite for authentication based on the `wsse:security` Web Security headers.

A basic UsernameToken security header can be explained as follows:

```
<S11:Envelope xmlns:S11="..." xmlns:wsse="...">
 <S11:Header>
...
   <wsse:Security>
   <wsse:UsernameToken>
     <wsse:Username>sysadmin</wsse:Username>
     <wsse:Password>password</wsse:Password>
     </wsse:UsernameToken>
   </wsse:Security>
...
  </S11:Header>
...
</S11:Envelope>
```

**Important: Authorization Check at Both the Trading Partner Level and WS-Security Header Level for XML Gateway Interfaces**

In Oracle XML Gateway, each trading partner is configured with Oracle E-Business Suite users. Only these authorized users defined in the Trading Partner Setup form are allowed to perform XML transactions. External clients can pass such user names in the `<USERNAME>` and `<PASSWORD>` elements defined within the `<ECX:SOAHeader>` element (or `<XMLGateway_Header>` element for generic XML Gateway services) in the SOAP body. These username parameters are validated by Oracle XML Gateway against the user name defined in the trading partner setup before initiating a transaction.

Therefore, for XML Gateway interface type, the authorization check is performed at both the trading partner level, as well as on the user name passed in the `wsse:security` header in the SOAP request. For information on trading partner setup and how to associate users with trading partners, see the *Oracle XML Gateway User's Guide*.

A WS-Security header in the SOAP message from Oracle E-Business Suite can be as follows:

```
<xml version="1.0" encoding="UTF-8">
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <env:Header>

 <wsse:Security xmlns:wsse="http://docs.oasis-open.
org/wss/2004/01/oasis-200401-wsswssecurity-secext-1.0.xsd>
  <wsse:UsernameToken>
     <wsse:Username>sysadmin</wsse:Username>
     <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-username-token-profile-1.0#PasswordText">password</wsse:
Password>
     </wsse:UsernameToken>
  </wsse:Security>
 </env:Header>

 <env:Body>
...
 </env:Body>
</evn:Envelope>
```

## SAML Sender-Vouches Token Based Security

To authenticate web services relying on sending a user name only through SAML assertion, Oracle E-Business Suite Integrated SOA Gateway supports SAML Token (Sender Vouches) based web service security.

Security Assertion Markup Language (SAML) is an XML-based standard for exchanging authentication and authorization data between security domains, that is, between an identity provider and a service provider.

**How to Authenticate Users through a Trusted Sender-Vouches SAML Token**

A SAML token uses SAML assertions as security tokens. One type of SAML token is the sender-vouches SAML token. This token uses a sender-vouches method to establish the correspondence between a SOAP message and the SAML assertions added to the SOAP message.

When a web application invokes a service that uses SAML token as its authentication type, this SOAP request message containing or referencing SAML assertions is received through Oracle SOA Suite and passed on to a JAAS based login module for Oracle E-Business Suite to authenticate the service based on the `wsse:security` Web Security headers. As part of the validation and processing of the assertions, the receiver or the login module for Oracle E-Business Suite must establish the relationship between the subject, claims of the referenced SAML assertions, and the entity providing the evidence to satisfy the confirmation method defined for the statements.

In other words, in order to validate and authenticate a user who logs on to the enterprise information system, a trusted sender-vouches SAML token security must be used to establish the correspondence between the SOAP message and the SAML assertions added to the SOAP message.

**Note:** Since everyone can send a SAML Token with valid conditions,

the authentication framework only trusts certain SAML token sources and stores the public key of each of these sources in a common key store. This Public Key Infrastructure (PKI) based security provides more sophisticated trusted rules to authenticate web services.

Please note that the following algorithms have been certified for SAML Token security in this release:

- Symmetric Encoding Algorithm: `http://www.w3.org/2001/04/xmlenc#aes128-cbc`
- Key Encryption Algorithm: `http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p`

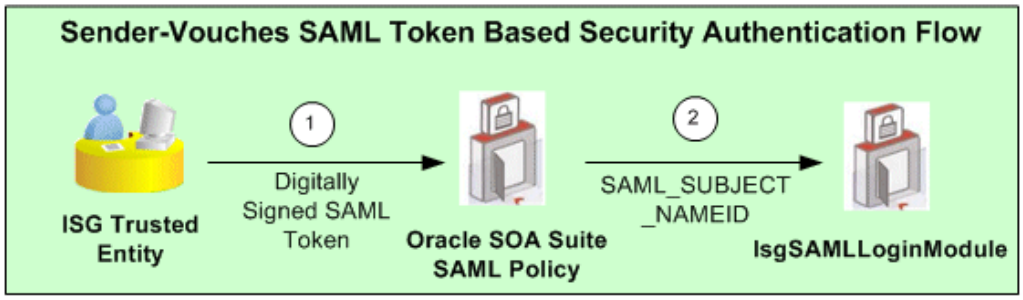
**Important:** To ensure SAML Token security works properly, necessary setup steps need to be performed. For setup information on SAML Token security, see *Setting Up SAML Token Security for Oracle E-Business Suite Integrated SOA Gateway Release 12.2*, My Oracle Support Knowledge Document 1332262.1 for details.

To authenticate users, any entity that establishes a PKI trust with Oracle E-Business Suite Integrated SOA Gateway can send the SAML Assertion with a valid user name. A PKI trusted entity will send a SAML token profile with the user name embedded with it and that must be digitally signed. The SAML Token policy attached to the web service verifies attributes like "Issuer", "Conditions", and so on. After the verification, the login module (`IsgSAMLLoginModule`) extracts the SAML principal (user name in `NameIdentifier`) through a NameCallback. This is verified against LDAP for Single Sign-On (SSO) users or against Oracle E-Business Suite `FND_USER` for non-SSO users.

Please note that for Oracle E-Business Suite Integrated SOA Gateway, it is mandatory that all users must be valid Oracle E-Business Suite users. If SSO is used, then the user in LDAP server for SSO should be in synchronous with Oracle E-Business Suite `FND_USER` table. Otherwise, the user authorization check will fail when looking up the application responsibilities for user authorization against entries in the `FND_USER` table. For more information on integrating Oracle E-Business Suite in an enterprise single sign-on environment, see the *Oracle E-Business Suite Security Guide*.

**Note:** The login module `IsgSAMLLoginModule` gets invoked through the Authentication Provider `IsgAuthenticator`.

The following diagram illustrates the sender-vouches SAML Token based security authentication process flow:




1. A trusted application authenticates a user and creates a digitally signed SOAP request, containing a SAML Sender-Vouches Token.

   Please note that a trusted application can be any application whose Public Key is known to Oracle E-Business Suite Integrated SOA Gateway and which can send digitally signed SAML Assertions in SOAP requests using that public key.

2. SAML Token Policy attached to the web service verifies signature and SAML conditions.

3. `IsgSAMLLoginModule` in Oracle SOA Suite extracts the SAML principal (user name in `NameIdentifier`) through a NameCallback. This is verified against LDAP for Single Sign-On (SSO) users or against Oracle E-Business Suite `FND_USER` for non-SSO users.

   The format of the `NameIdentifier` indicates if the user has been authenticated against LDAP (for a SSO user) or Oracle E-Business Suite `FND_USER` (for a non-SSO user). If the format is `dn=xxxx`, then this is a SSO user who has been authenticated against LDAP. Otherwise, this is a non-SSO user who has been authenticated against Oracle E-Business Suite `FND_USER`.

A sample sender-vouches SAML assertion for a non-SSO environment can be as follows:

```
<Assertion AssertionID="xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx" IssueInstant="
2010-02-27T17:26:21.241Z" Issuer="www.oracle.com" MajorVersion="1"
MinorVersion="1" xmlns="urn:oasis:names:tc:SAML:1.0:assertion"  xmlns:
samlp="urn:oasis:names:tc:SAML:1.0:protocol"><Conditions NotBefore="
2010-02-27T17:26:21.241Z" NotOnOrAfter="2011-02-27T17:26:21.241Z"/>
 <AuthenticationStatement AuthenticationInstant="2010-02-27T17:26:21.241
Z" AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password">
  <Subject>
    <NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-format:
unspecified" NameQualifier="notRelevant">SYSADMIN</NameIdentifier>
    <SubjectConfirmation>
    <ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:sender-
vouches</ConfirmationMethod>
    </SubjectConfirmation>
  </Subject>
 </AuthenticationStatement>
</Assertion>
```

A sample sender-vouches SAML assertion for a SSO environment can be as follows:

```
<Assertion
IssueInstant="2010-02-27T17:26:21.241Z" Issuer="www.oracle.com"
MajorVersion="1" MinorVersion="1"
xmlns="urn:oasis:names:tc:SAML:1.0:assertion"
xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"><Conditions
NotBefore="2010-02-27T17:26:21.241Z"
NotOnOrAfter="2011-02-27T17:26:21.241Z"/>
<AuthenticationStatement
AuthenticationInstant="2010-02-27T17:26:21.241Z"
AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password">
  <Subject>
    <NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-format:
unspecified"
     NameQualifier="notRelevant">orclApplicationCommonName=PROD1,
cn=EBusiness,cn=Products,cn=OracleContext,dc=us,dc=oracle,
dc=com</NameIdentifier>
    <SubjectConfirmation>
    <ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:sender-
vouches</ConfirmationMethod>
    </SubjectConfirmation>
  </Subject>
 </AuthenticationStatement>
</Assertion>
```

- `Issuer`: The value of this attribute is defined through Oracle SOA Suite. It will appear in `jps-config.xml`. For information on how to add `Issuer`, see *Setting Up SAML Token Security for Oracle E-Business Suite Integrated SOA Gateway Release 12.2*, My Oracle Support Knowledge Document 1332262.1.

- `Conditions`: This tag defines the time limit in which this SAML Assertion is valid.

- `NameIdentifier`: The value of this tag contains the user name.

  If the user name is of the form of LDAP DN, then the user name is verified in the registered OID for a SSO user. Otherwise, the user name is verified in `FND_USER` table for a non-SSO user.

- `SubjectConfirmation`: It should be sender-vouches.

For information on how the sender-vouches SAML Token is used in SOAP security header to authenticate web services, see SAML Token-based SOAP Security Header, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

## HTTP Basic Authentication

Oracle E-Business Suite Integrated SOA Gateway supports HTTP Basic Authentication security to authenticate the users who invoke REST services over secure transport protocol – HTTPS.

When an HTTP client application tries to access an Oracle E-Business Suite REST service, user security credentials (user name/password) should be provided as input data in HTTP header as part of the REST request message. The user name and password will be routed to LoginModule for authentication and authorization.

The LoginModule in turn extracts the credentials from HTTP header, authenticates the user against Oracle E-Business Suite user table, and establishes identity for the

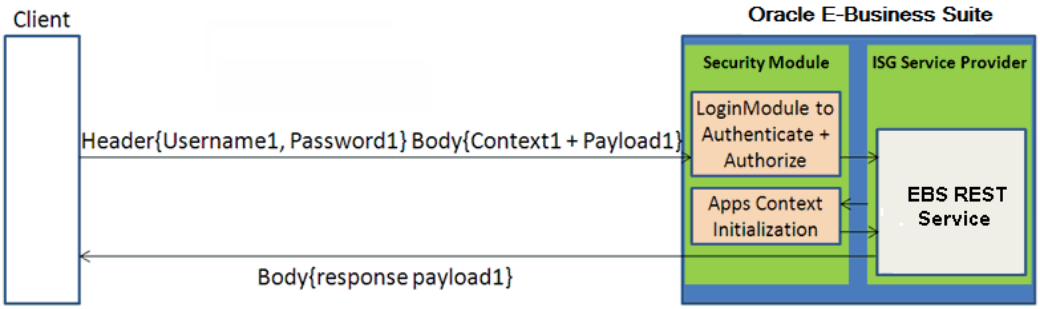
authenticated user. The LoginModule will then send the response to ISG Service Provider framework.

- For the authenticated and authorized user request, the Service Provider framework invokes a security service to initialize the application context, and then invokes the REST service.

- For the unauthenticated or unauthorized user request, the Service Provider framework returns system fault to the client.

The following diagram illustrates the authentication process flow of HTTP Basic Authentication security:

***Authentication Process Flow for HTTP Basic Authentication Security***




Based on HTTP Basic Authentication defined by W3C, the HTTP client application should use the following header field to send user credentials:

```
Authorization: Basic <base64 encoded version of username:
password>
```

Please note that if your Oracle E-Business Suite environment is configured for single sign-on (SSO), user authentication should be delegated to SSO which performs authentication against information stored in Oracle Directory Services (an LDAP server).

## Token Based Authentication

Token based security authenticates users using security tokens provided by the server. When a user tries to log on to a server with multiple requests, instead of authenticating the user each time with a given user name and password, a unique access token (such as Oracle E-Business Suite session ID) may be sent as `Cookie` in HTTP header.

For example, when an Oracle E-Business Suite user has initially authenticated on a given user name and password, after successful login, the security Login service creates an Oracle E-Business Suite user session and returns the session ID, as shown in the following:

```
<response>
<data>
<accessToken>xxxxxxxxxxxxxxxxxxxxxxxxxx</accessToken>
<accessTokenName>myEbsInstance</accessTokenName>
<ebsVersion>12.2.0</ebsVersion>
<userName>SYSADMIN</userName>
</data>
</response>
```

The session ID that points to the user session will be passed as `Cookie` to HTTP headers of all subsequent web service calls for user authentication.

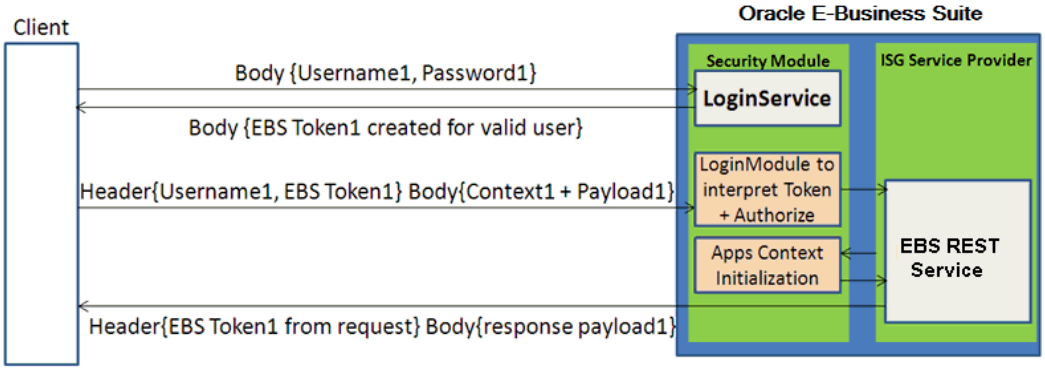
```
POST /webservices/rest/Invoice/create_invoice
Cookie: <accessTokenName>=<accessToken>
Content-Type: application/xml
```

The LoginModule will interpret and extract the token (session ID) from HTTP headers, and validate the subject or user name with token, not password, in the subsequent requests for authentication.

Similar to the HTTP Basic Authentication security, if the request passes the authentication and authorization, the Service Provider framework invokes a security service to initialize the application context, and then invokes the REST service. Otherwise, system fault will be returned.

The following diagram illustrates the authentication process flow of Token Based Authentication security:

***Authentication Process for Token Based Authentication Security***




In this diagram, user name and password are provided and validated in the initial request. A unique token (EBS Token1) is obtained through the Login Service for the valid user. In case a different service is requested in the subsequent call, user name along with the token, instead of the password, are provided in the header this time.

In this subsequent request, application context information that may be required in initializing Oracle E-Business Suite session is also provided in the request. Security LoginModule will be used to interpret and extract the token from the header to authenticate the user and then authorize the request. The application context session will also be initialized before invoking the REST service. After a successful service invocation, a response message will be sent along with the response payload if it is available.

### Advantages of Using Token Based Security

Note that when token based security is used, the application context information mentioned above does not have to be passed in every request. If the context values are not provided in the consecutive requests, the previously passed values will be used.

This will reduce the size of the payload included in HTTP headers and thus less data bandwidth is required. It is particularly useful for mobile data networks.

For more information on application context in REST header, see REST Header for Application Context, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

For more information about how to use context values to initialize or re-initialize the Oracle E-Business Suite session, see the Oracle Application Object Library REST Security Services section, *Oracle E-Business Suite Security Guide*.

# 7

# Logging for Web Services

## Overview

To extend logging support to more granular level and provide inside-out views for web service activities, Oracle E-Business Suite Integrated SOA Gateway provides an enhanced, flexible web service logging mechanism. An integration administrator can configure log settings at the service type (SOAP or REST) level of an interface. This includes selecting a desired interface name and service type that the logging feature should be set, enabling or disabling the design-time log, and selecting an appropriate runtime log severity level. Additionally, the web service auditing feature can be enabled or disabled through the same logging user interface at the service type level.

With proper logging setups and configuration, you can easily monitor and audit SOAP and REST service activities provided through Oracle E-Business Suite Integrated SOA Gateway. You can track log messages and troubleshoot any issues occurred at design time and runtime. Moreover, the administrator can delete existing log settings, and purge audit information through Service Monitor if needed.

> **Important:** In this release, logging feature is supported for both SOAP and REST services.

### Key Features

The enhanced web service logging feature includes the following features:

- It provides centralized, user-friendly user interface for logging and audit configuration for Oracle E-Business Suite SOAP and REST services.
- It allows logging and audit setups to be configured at the service type level of an integration interface.
- It lets you enable or disable the design-time log, runtime logs, and auditing.
- All design-time and runtime SOAP and REST service activities within Oracle E-

Business Suite can be logged and audited if the SOAP and REST service types of an interface have the logging feature enabled.

- It provides integrated log view allowing you to view SOAP and REST service generation and deployment logs through Integration Repository, and view service processing logs through Service Monitor if the design-time and runtime logs are enabled for the SOAP and REST service types of an interface.

- Auditing information for a specified service type within certain date range can be purged from the database tables through Service Monitor.

**Design-time logs** capture each stage of service generation and development lifecycle activities only if the design-time log for a desired service type (SOAP or REST) is enabled for an interface. For example, if the selected interface can be exposed as both SOAP and REST service types, you must select either SOAP or REST from the Service Type drop-down list for the logging to be enabled in one configuration.

> **Note:** For interfaces that can be available as both SOAP and REST services, to enable the logging and auditing features for both service types of the same interface, you need to create two separate configurations for each service type.

- If an interface has the design-time log enabled for a service type, such as SOAP service type, 'Enabled' is shown as the Log Configuration value in the SOAP Web Service tab of that interface.

  If there is any design-time log available for the enabled SOAP service type, **View Log** appears in the SOAP Web Service tab for that interface allowing you to view the log details in the Log & Error Details page. If any errors occurred during the design-time activities, the error details are also displayed in the Log & Error Details page.

- If the design-time log is not enabled for a service type of an interface, 'Disabled' is shown as the Log Configuration value for that selected service type. For example, if the design-time log is not enabled for the REST service type of an interface, then 'Disabled' is shown in the REST Web Service tab. If the SOAP service type of the same interface does not have the log enabled either, then 'Disabled' is also shown in the SOAP Web Service tab. If errors occurred while performing the design-time activities, then **View Error** appears instead for that interface allowing you to view the error and exception details only.

**Runtime logs** record service processing details during the invocation of an Oracle E-Business Suite service if the service with a desired service type has runtime log enabled. If log information is available for a given instance, the **Log** icon appears for that instance in Service Monitor. The administrator can view the log messages.

**Auditing** allows you to monitor and track SOAP service activities passed through

Oracle SOA Suite and REST service activities if the auditing feature for a desired service type of an interface is enabled. All messages for the specified service type of an interface including the associated payloads and fault messages can be saved and audited through Service Monitor.

To better understand the logging feature, the following topics are discussed in this chapter:



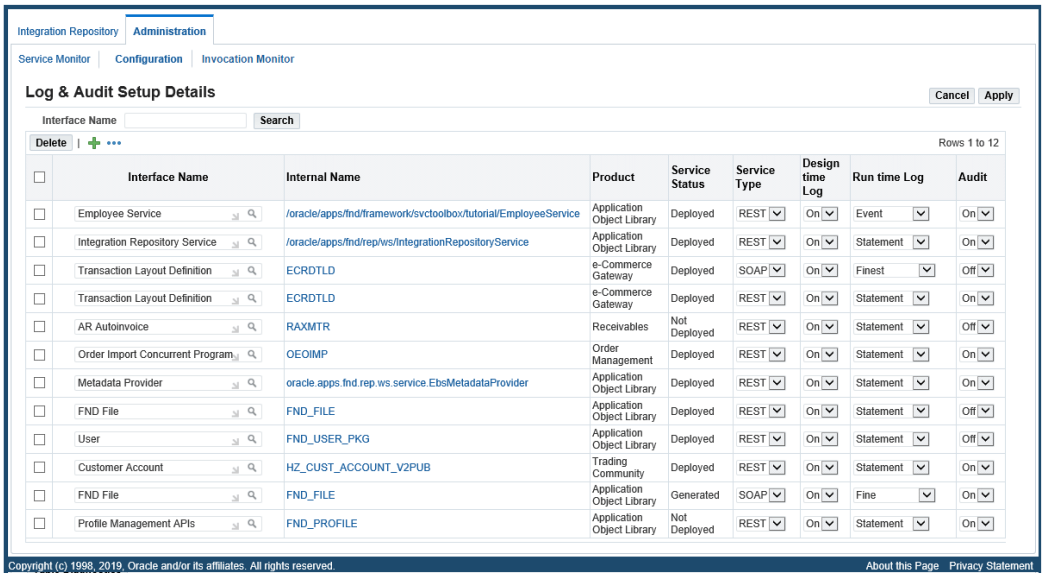
## Accessing the Logging Configuration User Interface

To access the log and audit setup page, log in to Oracle E-Business Suite as a user who has the Integration Administrator role.

Select the **Integrated SOA Gateway** responsibility from the navigation menu, and then select **Administration > Configuration**. The Administration tab appears with the **Configuration** subtab.

> **Note:** The **Administration** selection from the navigation menu appears only to the users who have the Integration Administrator role after logging in to Oracle E-Business Suite with the Integrated SOA Gateway responsibility.
>
> All administrative tasks performed outside the Integration Repository user interface are grouped and displayed under the **Administration** tab. These tasks include managing log and audit setups in the Configuration subtab, monitoring inbound invocations for Oracle E-Business Suite services in the Service Monitor subtab, and monitoring outbound service invocations through Service Invocation Framework in the Invocation Monitor subtab.

***Log & Audit Setup Details Page***




| Interface Name | Internal Name | Product | Service Status | Service Type | Design time Log | Run time Log | Audit |
|---|---|---|---|---|---|---|---|
| Employee Service | /oracle/apps/fnd/framework/svctoolbox/tutorial/EmployeeService | Application Object Library | Deployed | REST | On | Event | On |
| Integration Repository Service | /oracle/apps/fnd/rep/ws/IntegrationRepositoryService | Application Object Library | Deployed | REST | On | Statement | On |
| Transaction Layout Definition | ECRDTLD | e-Commerce Gateway | Deployed | SOAP | On | Finest | Off |
| Transaction Layout Definition | ECRDTLD | e-Commerce Gateway | Deployed | REST | On | Statement | On |
| AR Autoinvoice | RAXMTR | Receivables | Not Deployed | REST | On | Statement | Off |
| Order Import Concurrent Program | OEOIMP | Order Management | Deployed | REST | On | Statement | On |
| Metadata Provider | oracle.apps.fnd.rep.ws.service.EbsMetadataProvider | Application Object Library | Deployed | REST | On | Statement | On |
| FND File | FND_FILE | Application Object Library | Deployed | REST | On | Statement | Off |
| User | FND_USER_PKG | Application Object Library | Deployed | REST | On | Statement | Off |
| Customer Account | HZ_CUST_ACCOUNT_V2PUB | Trading Community | Deployed | REST | On | Statement | On |
| FND File | FND_FILE | Application Object Library | Generated | SOAP | On | Fine | On |
| Profile Management APIs | FND_PROFILE | Application Object Library | Not Deployed | REST | On | Statement | On |

The Log & Audit Setup Details page is the entry page to perform all the following logging setup and management activities:

- Viewing and Searching Existing Configurations, page 7-5

  All existing logging and audit settings listed by interfaces are displayed in the configuration table once the Log & Audit Setup Details page appears. Each entry in the table includes interface name, internal name, product name, service status, service type (SOAP or REST), design-time log status (On or Off), runtime log severity level, and audit feature status (On or Off).

  Clicking the Internal Name link from the table takes you to the interface details page for the selected interface in the Integration Repository.

- Adding a New Configuration, page 7-6

  To add a new log configuration for an interface, click '+' (Add Another Row: Log/Audit Configuration) icon in the Log & Audit Setup Details page. An empty row appears allowing you to add a new configuration. It includes specifying a desired interface and service type, enabling or disabling the design-time log and the service auditing feature, and selecting runtime log information.

- Updating an Existing Configuration, page 7-12

  From the configuration table, you can directly update an existing configuration by selecting a desired value for the log setting that you want to change. This setting includes service type, design-time log, log severity level, and audit feature status.

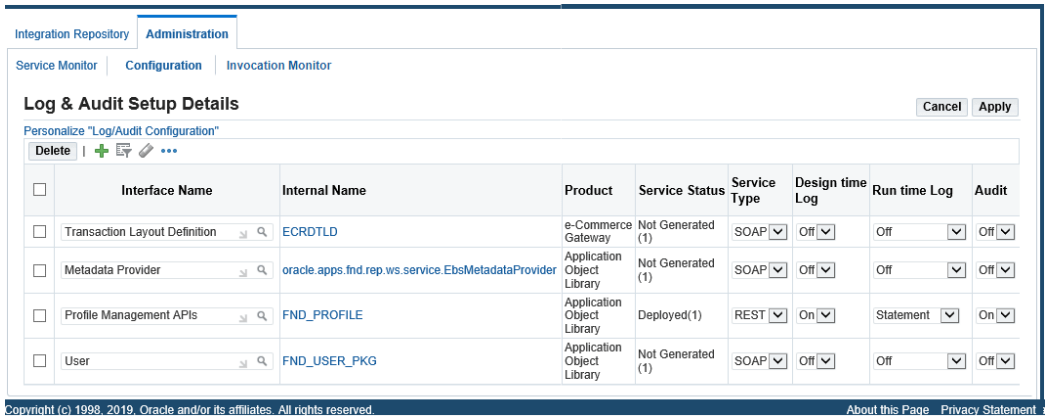
- Deleting an Existing Configuration, page 7-12

You can delete an existing configuration by selecting an interface with log settings that you want to remove and then clicking **Delete** from the Log & Audit Setup Details page.

## Viewing and Searching Existing Configurations

Logging is enabled at the service type level of an interface. When an integration administrator logs in to Oracle E-Business Suite, then selects the Integrated SOA Gateway responsibility, and then selects the **Administration > Configuration** link from the navigation menu, the Log & Audit Setup Details page appears. All existing log configurations by interface are automatically displayed in the configuration table.

***Log & Audit Setup Details Page with Existing Configurations Displayed***




Each log entry listed in the table contains interface name, internal name, product name, service type (REST or SOAP), service status, web service status, design-time log status (On or Off), runtime log severity level, and audit feature status (On or Off).

### Searching Existing Configurations

Search feature is available only if there are more than 10 interfaces that have log settings configured. In this situation, the Interface Name field appears on the top of this page letting you filter or search the configurations by interface name. After specifying the desired interface name (such as 'Order%') that you want to view the configuration details, click **Search** to run the query. All interface names that match your search criteria will be listed in the table.

If no log configuration has been defined, then an empty table with message 'No interface level logging configuration is defined.' appears.

From the configuration table, you can perform the following tasks:

- Add a new log configuration by clicking the **+** (Add Another Row) icon. See: Adding a New Configuration, page 7-6.

- Search the configuration list by Interface Name if there are more than 10 configurations in the table.
- View the selected interface by clicking the Internal Name link. This takes you to the interface details page in the Integration Repository.
- Update an existing configuration for a selected interface. This includes changing service type, enabling or disabling the design-time log and the service auditing feature, and changing runtime log severity level or disabling the runtime log. See: Updating an Existing Configuration, page 7-12.
- Delete an existing configuration by clicking **Delete** for a desired log configuration.

  See: Deleting an Existing Configuration, page 7-13.

**To view and search existing configurations:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility.

   From the navigation menu, select the **Administration > Configuration** link from the menu selection. The Log & Audit Setup Details page is displayed.

2. All existing log and audit configurations are automatically displayed by interface name in the table.

3. If there are more than 10 configurations listed in the table, you can perform a search by entering interface name and click **Search** to run the query. All matched interfaces will be listed in the table.

4. To delete existing configurations, select desired settings that you want to delete and click **Delete** to remove them from the database.

5. To add a new configuration, click **+** (Add Another Row) icon to add a new setting.

## Adding a New Configuration

With the support for both SOAP and REST services in the logging mechanism, you can configure new log settings at the service type level of an interface. After you click the **+** (Add Another Row) icon in the Log & Audit Setup Details page, an empty row is added to the end of the current configuration table letting you add a new configuration for a desired interface. This includes selecting a desired service type, specifying runtime log severity information (disabling it with value 'Off'), and enabling or disabling the design-time log and the service auditing feature for the selected interface.

> **Note:** Only if the design-time log is enabled for a specified service type of an interface, design-time activities will then be captured and logged.

These activities include the Generate, Deploy, Undeploy, Reset, Retire, and Activate actions for SOAP services and the Deploy and Undeploy actions for REST services. Without enabling the design-time log, the logs will not be written.

Enter information in the following fields to add a new configuration:

- **Interface Name:** Search and select a desired interface name that you want the logging to be enabled.

  Once you select a desired interface for the new configuration, the interface associated Internal Name, Product, and Service Status fields are automatically populated. The rest of the configuration fields including Service Type, Design Time Log, Run Time Log Level, and Audit fields are also displayed with default values. You can change them if needed.

- **Service Type:** Depending on the interface you choose for the configuration, you can have the following options:

  **Note:** Logging is configured at the service type (SOAP or REST) level of an interface or service. Configuration at the method or operation level is not supported in this release.

  - *PL/SQL APIs, Concurrent Programs, and Business Service Objects (Available for SOAP and REST Services)*

    You can select either 'SOAP' or 'REST' as the service type value for a configuration. By default, no value (blank) is selected.

    If you want to enable both the SOAP and REST service types of the same interface, you need to create one configuration for the SOAP service type and another configuration for the REST service type.

  - *Java Bean Services, Application Module Services, Open Interface Tables, and Open Interface Views (Available for REST Services Only)*

    By default, 'REST' is the only option that appears in this field.

  - *XML Gateway (Available for SOAP Services Only)*

    By default, 'SOAP' is the only option that appears in this field.

- **Design time Log (Optional):** Select a value in this field if you want to enable the design-time log for a selected service type of an interface. Use the design-time logs to troubleshoot any issues or exceptions encountered during the service generation and deployment life cycle.

  By default, the design-time log is turned off initially after you selected an interface.

After selecting a desired service type for an interface, you can enable the design-time log for the selected service type by selecting 'On' from the drop-down list. Once it is enabled, logs can be written for the design-time actions, including 'Generate', 'Deploy', 'Undeploy', 'Reset', 'Retire', and 'Activate' for SOAP services and 'Deploy' and 'Undeploy' for REST services. Without enabling the design-time log for your selected service type, the logs will not be written for that service type.

For example, a PL/SQL interface 'Order Capture' has the design-time log enabled for the SOAP service type. At design time during the SOAP service generation and deployment, logs specific to the selected 'Order Capture' interface as a SOAP service can then be captured through the Integration Repository user interface. **View Log** is shown in the SOAP Web Service tab for the interface 'Order Capture' letting you view log details and error details if occurred during the design-time activities.

> **Note:** In the above example, if the design-time log is not enabled for the SOAP service type, and if any errors occurred while performing the SOAP design-time activities, then **View Error** appears instead for that interface. Clicking **View Error** to access and view only the error and exception details in the Log & Error Details page.

For more information on viewing design-time logs, see Viewing Design-Time Logs for SOAP Services, page 3-24 and Viewing Design-Time Logs for REST Services, page 3-53.

- **Run time Log (Optional):** Select a desired log value if you want to enable the runtime log for a selected service type of an interface. Log level controls logging output for the enabled service type of a service.

  By default, the runtime log is turned off for both the SOAP and REST service types. You can enable the log by changing its default value 'Off' to any other log level from the list for the selected service type.

  The following tables describe the available log levels used for the SOAP and REST services:

  - **SOAP Services**

***Log Levels for SOAP Services***

| Severity | Description |
|---|---|
| Off (default) | This severity level is used to turn off logging. |

| Severity | Description |
| --- | --- |
| Severe | This is a message level indicating a serious failure. |
| Warning | This is a message level indicating a potential problem. |
| Information | This is a message level for informational messages. |
| Configuration | This is a message level for static configuration messages. |
| Fine | This is a message level providing tracing information. |
| Finer | This severity level indicates a fairly detailed tracing message. |
| Finest | This severity level indicates a highly detailed tracing message. |

- **REST Services**

***Log Levels for REST Services***

| Severity | Description |
| --- | --- |
| Off (default) | This severity level is used to turn off logging. |
| Statement | This severity is used for low-level progress reporting. |
| Procedure | This severity is used at integration points for API-level progress reporting. |
| Event | This severity is used for high-level progress reporting, such as starting a new transaction, etc. |

| Severity | Description |
| --- | --- |
| Exception | This severity indicates a handled internal failure which typically requires no fix, such as Java exceptions, etc. |
| Error | This severity indicates an end user error which typically requires a fix from the user. |
| Unexpected | This severity indicates any unrecoverable errors that could occur. |

At runtime during the invocation of Oracle E-Business Suite services, if a service with a desired service type has the runtime log enabled, the associated log messages for the selected service type (SOAP or REST) are captured. You can click the **Log** icon in the Service Monitor search result table to open the Web Service Runtime Logs page where you can view the logs recorded for the service against a specific instance.

> **Important:** Runtime logging for PL/SQL, Concurrent Program, and XML Gateway SOAP services is handled by Oracle SOA Suite; therefore, setting runtime log levels for these services in the Log & Audit Setup Details page will display Oracle SOA Suite logs if the services are deployed on Oracle SOA Suite. Limited runtime log statements from the Oracle E-Business Suite Integrated SOA Gateway code (identified by the package name `oracle.apps.fnd.isg`) will be shown for these services.
>
> Runtime logging for Business Service Object SOAP services is handled by Oracle E-Business Suite Integrated SOA Gateway; therefore, Service Monitor shows Oracle E-Business Suite Integrated SOA Gateway logs for these interfaces based on the log level selected here.

For more information on viewing runtime logs, see Viewing Service Processing Logs, page 7-15.

- **Audit (Optional):** Select a value in this field if you want to enable the auditing feature for a selected service type of an interface.

  By default, the auditing feature is turned off. You can enable the feature by selecting 'On' in this field to create the audit trail for the selected service type of an interface.

For example, if the SOAP service type of an interface is enabled, all SOAP messages for that interface that Oracle SOA Suite processes along with the associated payloads and fault messages can be saved and audited through Service Monitor. For more information about Service Monitor, see Monitoring and Managing Inbound Service Invocation Messages Using Service Monitor, page 8-1.

Note that log messages can be correlated across application and database servers. If a new configuration is added for a service type of an interface that has been deployed, the newly-configured log setting including the runtime log level configured for that deployed service will be added to Oracle SOA Suite if it's for the SOAP service type. When the configuration is deleted for a deployed service, the runtime log level would be reset at the composite level as well in Oracle SOA Suite for SOAP services. The same mechanism applies when an integration administrator updates an existing log level for a deployed service, that is the new log level will be updated in the database.

If a new configuration is added for a service type of an interface that is not deployed, then the runtime log configuration including log level set for that service would be effective after the service with desired service type is deployed.

**To add a new configuration:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility.

   Select **Administration > Configuration** from the navigation menu. The Log & Audit Setup Details page is displayed.

2. To add a new configuration, click **+** (Add Another Row) icon.

   An empty row appears allowing you to enter information in the following fields:

   - Interface Name: Specify an appropriate interface name for the log is configured.

     Once the Interface Name field is selected, the associated Internal Name, Product, and Service Status fields are automatically populated. The rest of configuration fields such as the Design Time Log, Run Time Log, and Audit fields are also displayed with default values. You can change them if needed.

   - Service Type: Select a desired service type available for the selected interface.

   - Design time Log: By default, it is set to "Off". You can enable the design-time log by selecting 'On' from the drop-down list.

   - Run time Log: By default, it is set to "Off" and the runtime log is turned off. You can change the default value by selecting an appropriate value from the drop-down list.

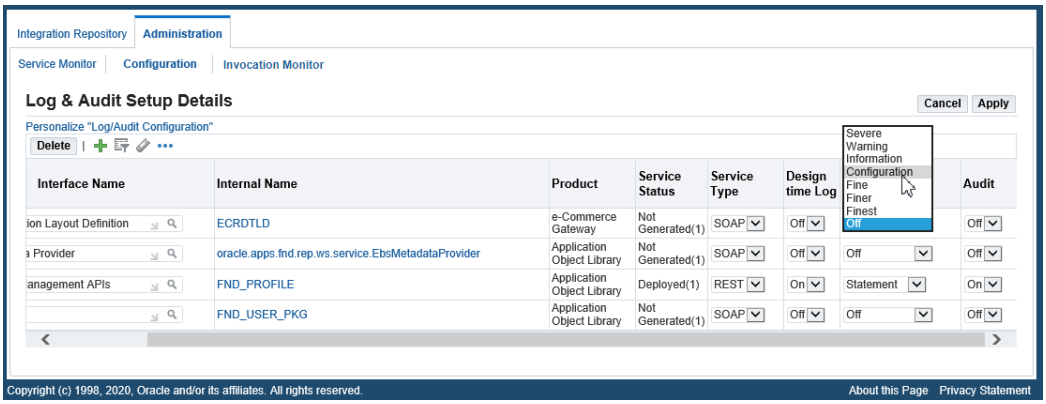
   - Audit: By default, it is set to "Off". You can enable the auditing feature by selecting 'On' from the drop-down list.

3. Click **Apply** to save the information.

## Updating an Existing Configuration

From the Log & Audit Setup Details page, you can modify an existing configuration for a selected interface including changing service type and runtime log severity, and enabling or disabling the design-time log and the auditing feature.

***Log & Audit Setup Details Page with Updating Runtime Log Information Highlighted***




To update the log settings for an interface, select an appropriate value from the drop-down list. For example, to enable the runtime log for the 'Order Capture' REST service, you need to select 'REST' as the service type for the 'Order Capture' interface and change the 'Off' value to 'Information' in the Run Time Log field. All messages during service invocation specific to the 'Order Capture' REST service will be written.

After you modify the existing settings for an interface, click **Apply** to save changes to the database and on Oracle SOA Suite if the changes applied to a SOAP service that has been deployed. Click **Cancel** to display the previous saved details.

**To update an existing configuration:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility.

   Choose **Administration > Configuration** from the navigation menu. The Log & Audit Setup Details page is displayed.

2. Update the basic log settings for an interface by selecting appropriate values from the drop-down lists for the service type, design-time log, runtime log level, and the Audit field.

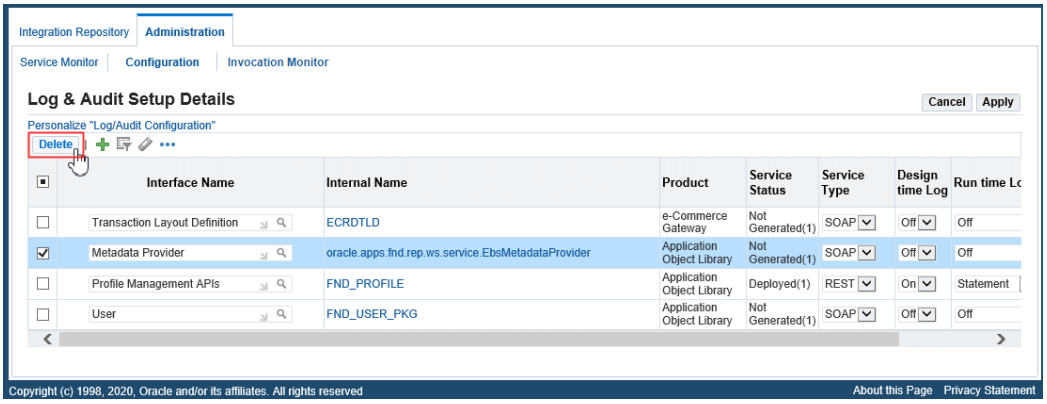
3. After the modification, click **Apply** to save the changes. Click **Cancel** to display the previous saved details.

## Deleting an Existing Configuration

If an existing configuration is no longer needed, you can remove it directly from the Log & Audit Setup Details page.

To delete existing configurations, select at least one setting that you want to remove and then click **Delete**. This removes the records from the existing configuration list and database. A confirmation message appears indicating that the selected log setups have been successfully deleted. This disables the logging and audit features for the selected interfaces.

***Log & Audit Setup Details Page with "Delete" Button Selected***




For a SOAP service that has been deployed to Oracle SOA Suite, once a configuration is deleted for that service, the runtime log level would be reset at the composite level as well in Oracle SOA Suite.

**To delete an existing logging configuration:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility.

   Choose **Administration > Configuration** from the navigation menu. The Log & Audit Setup Details page appears.

2. To delete an existing configuration, select a desired interface level setting that you want to remove and click **Delete**. The configuration for the selected interface is removed from the list and the system.

## Viewing, Deleting, and Exporting Log Messages

To effectively troubleshoot or debug errors if occurred at each stage of the service deployment life cycle, you can view and download log details recorded for an interface or service if it has the logging feature enabled properly.

Note that sensitive information such as passwords and security credentials in unencrypted plain text will not be logged.

- *Viewing Design-Time Logs*

  At design time during the service generation and deployment life cycle, logs can be captured through the Integration Repository user interface if the design-time log is enabled for a specific service type of an interface.

  For example, if an interface has the design-time log enabled for the 'SOAP' service type only, **View Log** appears only in the SOAP Web Service tab. If the 'REST' service type is also enabled for the same interface in another configuration, **View Log** appears in the REST Web Service tab as well for the same interface.

  > **Note:** If an interface does not have the design-time log enabled for any service type and if errors occurred during the design-time activities, such as 'Generate', 'Deploy', 'Undeploy', 'Reset', 'Retire', and 'Activate' for SOAP services and 'Deploy' and 'Undeploy' for REST services, **View Error** appears instead letting you view only the error or exception message details. You will not find log messages recorded because the design-time log is not enabled.

  Clicking **View Log** opens the Log & Error Details page where you can view log and error information the following regions:

  - **Log Details**: You can view log messages compiled in a table in this region.
  - **Error Details**: You view error message details in this region, only if errors occurred during the design-time activities.

  For more information on viewing design-time logs, see:

  - Viewing Design-Time Logs for SOAP Services, page 3-24
  - Viewing Design-Time Logs for REST Services, page 3-53

- *Viewing Service Processing Logs*

  At runtime during the service invocation, log messages can be captured and viewed through Service Monitor. Click the **Log** icon for a request in the search result table to open the Web Service Runtime Logs page where you can view the log details for the request against a specific instance.

  The Web Service Runtime Logs page contains the following regions:

  - **Runtime Middle Tier Logs**: Logs in this region are retrieved from the middle or application tier.
  - **Adapter Logs**: Logs in this region are retrieved from the adapter framework

layer.

After viewing adapter log messages retrieved from the Oracle E-Business Suite table for a service, you can delete them if needed by clicking **Delete Log**.

For more information on viewing log messages recorded while processing service requests, see Viewing Service Processing Logs, page 7-15.

## Viewing Service Processing Logs

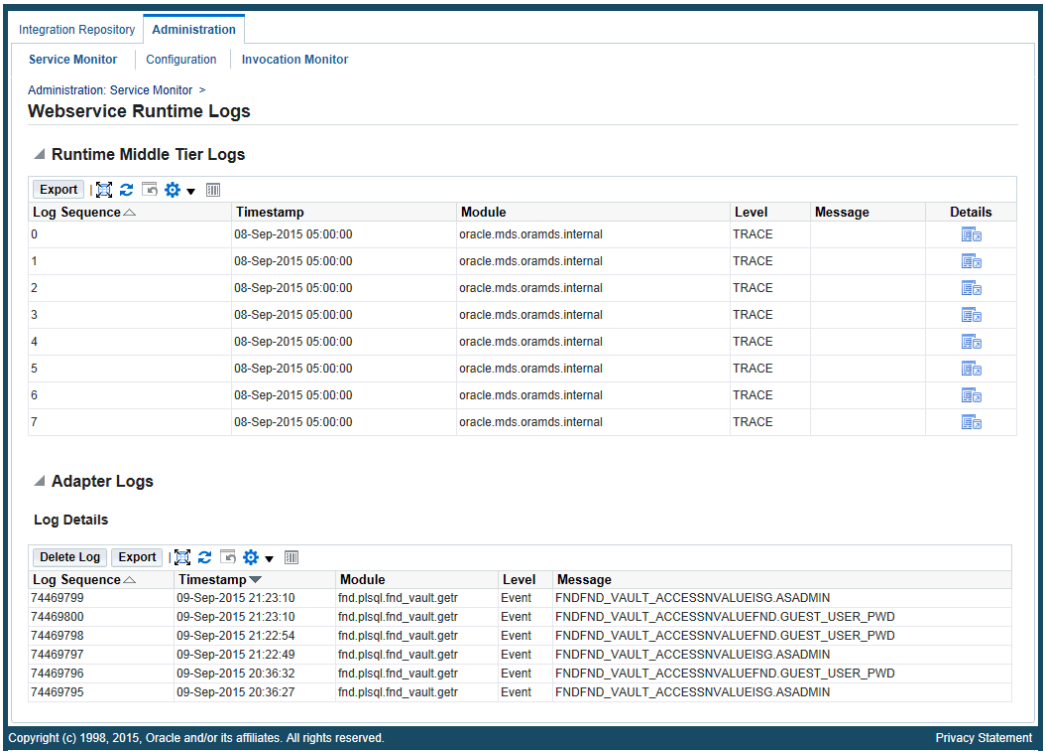
To effectively monitor SOAP and REST messages at runtime during the invocation of Oracle E-Business Suite services, if the runtime logging is enabled for a specific interface with a service type specified in the Log & Audit Setup Details page, log messages can be captured in Service Monitor against that instance for the specified service.

When a SOAP or REST request is received, Service Provider generates a unique numeric instance ID based on a database sequence and passes it to Service Monitor. Therefore, each request in Service Monitor appears with instance ID and the **Log** icon letting you retrieve the log details.

Click the **Log** icon in the search result table to view log messages in the Web Service Runtime Logs page.

> **Important:** For SOAP services, runtime logging for PL/SQL, Concurrent Program, XML Gateway interface types is handled by Oracle SOA Suite; therefore, setting runtime log levels for these services in the Log & Audit Setup Details page will display Oracle SOA Suite logs if the services are deployed in Oracle SOA Suite. Limited runtime log statements from the Oracle E-Business Suite Integrated SOA Gateway code (identified by the package name `oracle.apps.fnd.isg`) will be displayed for these services. Runtime logging for Business Service Object interface type is handled by Oracle E-Business Suite Integrated SOA Gateway; therefore Service Monitor shows Oracle E-Business Suite Integrated SOA Gateway logs for these interfaces based on the log level selected in the Log & Audit Setup Detail page.

***Web Service Runtime Logs Page***




The Web Service Runtime Logs page contains the following log regions:

- **Runtime Middle Tier Logs**: Logs in this region are retrieved from the middle or application tier.
  - For SOAP services, logs are retrieved from the Oracle SOA Suite server for Oracle E-Business Suite integration.

    Logs are compiled in a table for a selected service request. Click the **Details** icon from the table to view the application tier log details.

  - For REST services, logs are retrieved from the Oracle E-Business Suite application tier.

- **Adapter Logs:** Logs in this region are retrieved from the adapter framework layer.

  These log messages are compiled and listed in the table format for the selected service in a given instance. Each entry in the table includes log sequence, log timestamp, module, severity level, and actual message.

  **Deleting and Exporting Logs in the Adapter Logs Region**

  In the Adapter Logs region, after viewing log messages retrieved for a request in a given instance, you can delete them if needed by clicking **Delete Log**. A warning message appears alerting you that this will permanently delete all adapter logs in

the table. Click **Yes** to confirm the action. An empty log table appears after all adapter log messages have been successfully deleted.

Before deleting the logs, you can save a backup copy by clicking **Export**. This exports the records listed in the table to Microsoft Excel and you can use it later.

> **Note:** The log records deleted here are instance specific, whereas the Purge program from Service Monitor where you need to enter specific date range and service type before processing the purge request is not instance specific. The purge concurrent request will delete only the service processing logs for which the service is completed with a 'SUCCESS' status. It does not delete the logs for the service with a 'FAILURE' status.
>
> For more information on purging logs through Service Monitor, see Purging SOAP and REST Messages, Audits, and Logs, page 8-11.

**To view log messages in Service Monitor:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility.

   From the navigation menu, select the **Service Monitor** link from the Administration section to open the Monitor Search page.

2. Perform a search to display the search result. See: Searching SOAP and REST messages, page 8-3.

3. In the search result table, click the **Log** icon for a desired instance. The Web Service Runtime Logs page appears letting you view the log details.

4. In the Adapter Logs region, click **Delete Log** to delete all the logs listed in the table for a given instance if needed. Click **Yes** to confirm the action. Click **No** to return to the Web Service Runtime Logs page.

   Click **Export** to export log list table to Microsoft Excel.

## Configuring File Logging (Optional)

In Oracle E-Business Suite release 12.2, log statements can be captured in either of the following places:

- File system

- Database tables

  By default, log statements are captured in the database if logging is enabled from the Log & Audit Setup Details page.

In comparison to file logging, database logging reduces the performance of design-time operations. Performance can be improved by setting the optional parameter `<sid>.ISG_KEEP_ALIVE_DB_CONN=true` in `$INST_TOP/soa/isgagent.properties` for Oracle E-Business Suite.

This section describes the steps to enable the file logging for both SOAP and REST services.

- Steps to Enable File Logging for SOAP Services, page 7-18
- Steps to Enable File Logging for REST Services, page 7-19

When the file logging is enabled, log statements for design-time and runtime operations are not shown in the Interface Details page and Service Monitor user interface.

### Steps to Enable File Logging for SOAP Services

To capture log statements recorded for SOAP services on the file system, you need to enable the file logging using the following steps:

1. Set the following properties from `$INST_TOP/soa/isgagent.properties` in Oracle E-Business Suite and `<ISGTEMP>/appsutil/<EBS_CONTEXT_NAME>/bpel/isg.properties` in Oracle SOA Suite:

   `<ISGTEMP>` indicates a temporary folder created with *write permission* on the Oracle SOA Suite server. For details about this folder and `isg.properties`, see *Installing Oracle E-Business Suite Integrated SOA Gateway, Release 12.2*, My Oracle Support Knowledge Document 1311068.1.

   > **Note:** Logging mechanism should be the same across Oracle E-Business Suite and Oracle SOA Suite. If file logging is enabled in Oracle E-Business Suite, then it must be enabled in Oracle SOA Suite as well.

   - `<SID>.ISG_GLOBAL_LOG=TRUE`
   - `<SID>.ISG_LOGGER=FILE`

2. Stop and restart the `oafm` and `oacore` managed servers in an Oracle E-Business Suite environment and the Oracle SOA Suite managed server in Oracle SOA Suite.

   > **Note:** If your instance is configured with multiple nodes, stop and restart the `oafm` and `oacore` managed servers on each Oracle E-Business Suite and Oracle SOA Suite node of the multi-node environment.

   The `ISGLog.log` file is created in `<SID>.ISG_TEMP_DIRECTORY_LOCATION`

specified in the `isgagent.properties` file and `isg.properties`.

### Steps to Enable File Logging for REST Services

To capture log statements recorded for REST services on the file system, you need to enable the file logging using the following steps:

1. Set the following properties from `$INST_TOP/soa/isgagent.properties` in Oracle E-Business Suite:
   - `<SID>.ISG_GLOBAL_LOG=TRUE`
   - `<SID>.ISG_LOGGER=FILE`
2. Stop and restart the `oafm` and `oacore` managed servers in an Oracle E-Business Suite environment.

   **Note:** If your instance is configured with multiple nodes, stop and restart the `oafm` and `oacore` managed servers on each Oracle E-Business Suite node.

The `ISGLog.log` file is created in `<SID>.ISG_TEMP_DIRECTORY_LOCATION` specified in the `isgagent.properties` file.

# 8

# Monitoring and Managing Inbound Service Invocation Messages Using Service Monitor

## Service Monitor Overview

Service Monitor, previously known as SOA Monitor, is a centralized, light-weight service invocation monitoring and management tool. It fetches data and statistics for each instance of inbound service invocations and associated messages and provides monitoring capability for Oracle E-Business Suite services.

You can view all SOAP messages received and sent from Oracle SOA Suite and REST messages received and sent directly to Oracle E-Business Suite through Service Monitor. Additionally, Service Monitor provides auditing records for the service invocation details if the auditing feature is enabled.

> **Important:** In this release, both SOAP and REST services are monitored and audited through Service Monitor.

For the monitoring purpose, Service Monitor stores basic information about inbound service invocation for all the services, such as instance ID, integration interface details, header information, start date, end date, status, and other information depending on the service type. It does not store request and response payloads, fault message, or attachments unless the auditing feature is turned on for a desired service type of an interface.

> **Important: Enabling Service Auditing Feature Using the Configuration Subtab**
>
> For the monitoring purpose, Service Monitor is a permanent monitoring tool and is enabled at all times to monitor all Oracle E-Business Suite services. However, its auditing feature needs to be explicitly enabled at the service type level (REST or SOAP) of an

interface through the Log & Audit Setup Details page.

For more information on enabling the auditing feature along with log configuration, see Adding a New Configuration, page 7-6.

When the auditing feature is enabled for a service type of an interface, Service Monitor saves the corresponding payloads, fault messages, and attachments if they are available for an instance. This provides additional audit trails for integration administrators to quickly retrieve service invocation details and identify errors or exceptions if occurred.

**Accessing Service Monitor**

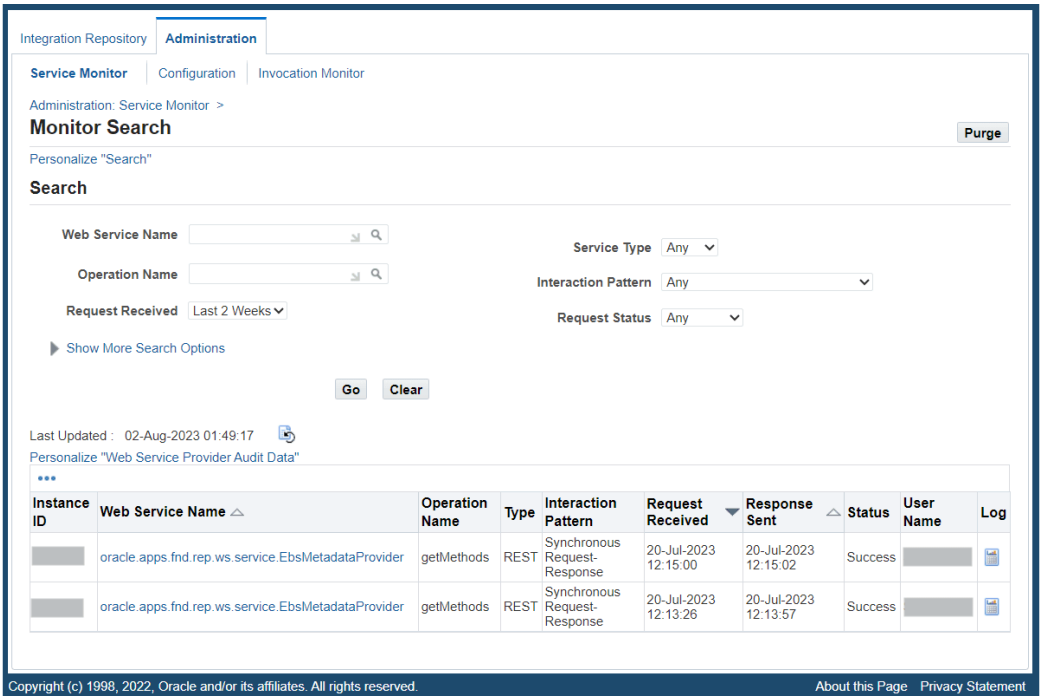
To access Service Monitor, log in to Oracle E-Business Suite as a user who has the Integration Administrator role.

Select the **Integrated SOA Gateway** responsibility from the navigation menu and then select the **Administration > Service Monitor** link. The **Service Monitor** subtab is displayed with the Monitor Search page.

**Note:** The **Administration** selection from the navigation menu appears only to the users who have the Integration Administrator role after logging in to Oracle E-Business Suite with the Integrated SOA Gateway responsibility.

All administrative tasks performed outside the Integration Repository user interface are grouped and displayed under the Administration tab. These tasks include monitoring inbound invocations for Oracle E-Business Suite services in the Service Monitor subtab, managing log and audit setups for inbound services in the Configuration subtab, and monitoring outbound service invocations through Service Invocation Framework in the Invocation Monitor subtab.

***Monitor Search Page***




Integration administrators can perform the following activities through Service Monitor:

- Searching SOAP and REST Requests, page 8-3
- Viewing SOAP and REST Request and Response Details, page 8-6
- Viewing Log Messages, page 8-10
- Purging SOAP and REST Messages, Audits, and Logs, page 8-11
- Enabling Web Service Auditing Using the Configuration Subtab, page 8-13

## Searching SOAP and REST Requests

In the Search region, you can perform searches on the SOAP requests received and sent from Oracle SOA Suite and the REST requests received directly in Oracle E-Business Suite based on the criteria you specified.

Service Monitor allows you to search SOAP and REST requests by interface name, operation name, request received time, service type, interaction pattern, and request status.

Based on the Service Type value your selected, you can select an appropriate value for the Interaction Pattern field:

- **SOAP**: The selection can be 'Any' (default), 'Synchronous Request-only', 'Asynchronous Request-only', 'Synchronous Request-Response', or 'Asynchronous Request-Response'.

- **REST**: The selection can be either 'Any' (default) or 'Synchronous Request-Response'.

- **Any**: The selection can be either 'Any' (default) or 'Synchronous Request-Response'.

The Request Received time can be selected from the list of values. Its value can be 'Any Time', 'Last 2 Weeks', 'Last 30 Days', 'Last 60 Days', 'Last 90 Days', 'This Week', or 'Today'. By default, 'This Week' is selected.

> **Note:** All the list of value selections from the Request Received field will include the requests received day of Today except 'Any Time'. For example, 'This Week' means the last 7 days inclusive of today the requests have been received, and 'Last 30 Days' means the last 30 days inclusive of today the requests have been received.
>
> 'Any Time' means a blind search of requests received regardless of the Request Received date. If this field is left blank, then 'This Week' is the default value for the Request Received time.

You can optionally enter more search criteria by clicking the **Show More Search Options** link in the Search region. These criteria include user name and a selected time frame.

When the search is processed, all entries that match your search criteria will be retrieved and displayed in a table. Each entry in the result table includes the instance ID, web service name, operation name, service type, interaction pattern, date and time the request was received and responded, user name, request status, and log.

If service processing log messages are available for an instance, the **Log** icon is enabled in the result table letting you view the log messages.

From the search result page, you can perform the following tasks:

- View the request and response details in the Service Instance page by clicking the **Instance ID** link for a given request

  See: Viewing SOAP and REST Request and Response Details, page 8-6.

- View the interface details in the corresponding SOAP Web Service tab or REST Web Service tab of the interface by clicking the **Web Service Name** link for a given request

- View the status of each monitored request and response

- View service processing log details by clicking the **Log** icon if log messages are

available for an instance

See: Viewing Log Messages, page 7-15.

- Purge SOAP and REST requests and responses, audits, as well as log messages collected over a period of time by clicking the **Purge** button

  See: Purging SOAP and REST Messages, Audits, and Logs, page 8-11.

The service auditing feature is enabled at the service type level of an integration interface through the Log & Audit Setup Details page. For more information on how to enable or disable the auditing feature, see Adding a New Configuration, page 7-6.

**To perform a search:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility. From the navigation menu, select the **Service Monitor** link from the Administration section to open the Monitor Search page.

2. In the Search region, enter appropriate search criteria including interface name, operation name, request received time, service type, interaction pattern, and request status for your search. Click **Go** to run your search.

3. Optionally, enter more search criteria by clicking the **Show More Search Options** link to enter the following information:

   - From: Enter an appropriate search start date.

   - To: Enter an appropriate search end date.

   - Username: Search and select an appropriate user name.

   Click **Go** to run your search.

4. All requests that match your search criteria appear.

5. Click the **Instance ID** link for a given ID to view the request and response details in the Service Instance page.

6. Click the **Web Service Name** link for a given request to view the interface details.

7. Click the **Log** icon, if service processing logs are available for a given instance ID, to view the log details.

8. Click **Purge** to purge SOAP and REST requests and responses, audits, as well as log messages collected for a period of time.

# Viewing SOAP and REST Request and Response Details

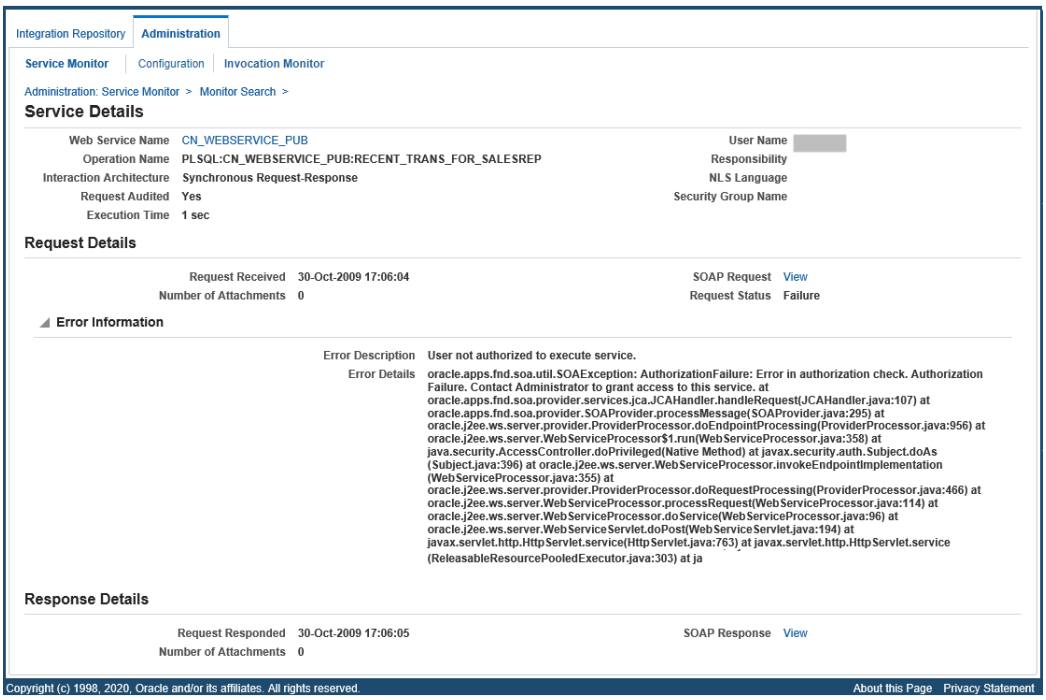
After you perform a search, all SOAP and REST messages that match your criteria are retrieved. To view the request and response details for a given instance, click the **Instance ID** link for a desired instance listed in the search result table. The Service Instance page appears for the selected instance with SOAP or REST service type.


- Viewing SOAP Request and Response Details, page 8-6
- Viewing REST Request and Response Details, page 8-8


## Viewing SOAP Request and Response Details

If you click an instance ID that is for a SOAP message, the Service Details page appears where you can view the selected SOAP service details.

***Service Details Page for Viewing SOAP Request and Response Details***




General interface information is shown at the top of the page. It includes interface name, user name, operation name, interaction pattern, auditing information, execution time, responsibility, NLS language, and security group name.

> **Note:** The application context fields (such as responsibility, NLS language, and security group name) display the values passed that are

required during the service invocation. These context values may be retrieved from the SOAP Header for a SOAP request.

Clicking an interface name link launches the interface details page for the service in Integration Repository. This lets you view the integration interface and service in details.

In addition to the general header, the following regions appear:

- **Request Details:** This region contains the service request related information including request received date and time, request status, SOAP request message allowing you to view the request payload, and number of attachments.

  Click the SOAP request **View** link if available to view the actual file in XML format except the password from the header.

  > **Note:** The **View** link appears only if at the time of processing that request, the auditing feature was enabled for the selected interface or service. If it was disabled at the time of processing that request, the **View** link will not appear. The same theory applies to processing responses as well.

***A Sample SOAP Request Message***


Additionally, the following regions appear in the Request Details region if certain conditions are met:

- **Error Information**: This region appears only if the request has a "Failure" status caused by server fault.

  This region displays error description and details. For error messages, error codes, and possible solutions, see Error Messages, page C-1.

- **Attachment**: If the SOAP request has attachments associated with it, the Attachment region appears with attachment details including all attachment

names and MIME Type information.

- **Service Response:** This region contains the response sent date and time, number of attachments, and the message **View** link if available to view the actual payload of the response in XML.

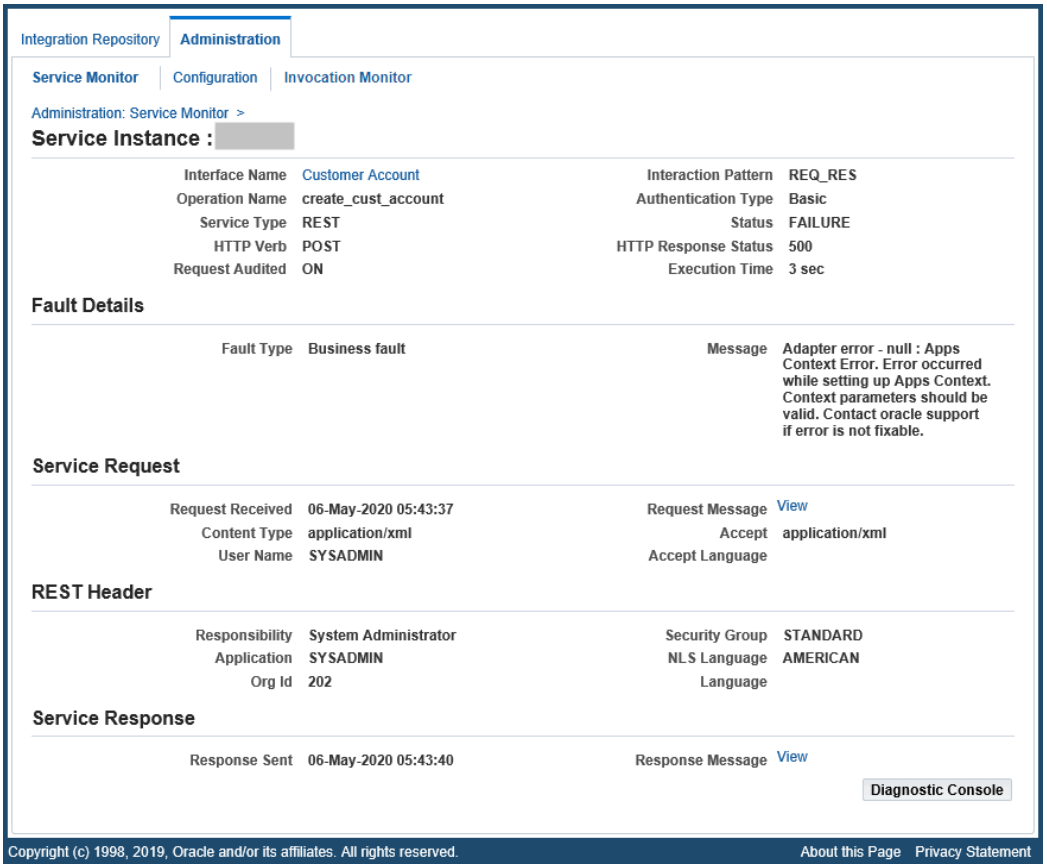
  **Note:** The **View** link appears only if at the time of processing that response, the auditing feature was enabled for the selected interface or service. If the auditing feature was turned off at the time of processing that response, the **View** link will not appear. The same theory applies to processing requests as well.

  Additionally, if the Interaction Pattern is of type 'Request-Only', the **View** link for response payload is not shown.

## Viewing REST Request and Response Details

If the selected instance ID is for a REST message, the Service Instance page appears where you can view the selected REST service details.

***Service Instance Page for Viewing REST Request and Response Details***




General interface information is shown at the top of the page. This header information includes interface name, operation name, interaction pattern, service type, HTTP verb, authentication type, status, HTTP response status, execution time, and auditing information.

Clicking an interface name link launches the interface details page for the REST service in Integration Repository.

In addition to the general header, the following regions appear:

- **Fault Details (Conditional**: This region appears only if there is any error or exception while invoking the web service request.

  *Fault Type*: Depending upon the nature of fault, the value could be one of the following types:

  - Client Error - All Oracle E-Business Suite Integrated SOA Gateway exceptions returning 4xx HTTP status code fall under this category.

  - Business Fault - This type belongs to Oracle E-Business Suite Integrated SOA Gateway exceptions with 500 HTTP status code and exception thrown by an

API.

- Technical Fault - All the remaining 5xx code errors fall under this category, such as ISG_SYSTEM_ERROR.

*Fault Message*: Fault code or the first 150 characters of the error message are shown.

For error messages, error codes, and possible solutions, see Error Messages, page C-1.

- **Service Request**: This region contains the service request related information including request received date and time, request message with the **View** link if available, content type, accept, accept-language, and user name.

  Click the request message **View** link if available to view the actual file (except the password from the REST Header) in XML or JSON format, depending on the message format.

***A Sample REST Request Message in XML***


```
<CREATE_CUST_ACCOUNT_Input
xmlns:nstrgmpr="http://xmlns.oracle.com/apps/hz/rest/CustomerAccount/create_cust_account/">
<RESTHeader> <Responsibility>SYSTEM_ADMINISTRATOR</Responsibility>
<RespApplication>SYSADMIN</RespApplication> <SecurityGroup>STANDARD</SecurityGroup>
<NLSLanguage>AMERICAN</NLSLanguage> <Org_Id>202</Org_Id> </RESTHeader>
<nstrgmpr:InputParameters> <nstrgmpr:P_CUST_ACCOUNT_REC>
<nstrgmpr:CREATED_BY_MODULE>BO_API</nstrgmpr:CREATED_BY_MODULE>
</nstrgmpr:P_CUST_ACCOUNT_REC> <nstrgmpr:P_ORGANIZATION_REC>
<nstrgmpr:ORGANIZATION_NAME>          </nstrgmpr:ORGANIZATION_NAME>
<nstrgmpr:ORGANIZATION_NAME_PHONETIC>          </nstrgmpr:ORGANIZATION_NAME_PHONETIC>
</nstrgmpr:P_ORGANIZATION_REC> <nstrgmpr:P_CUSTOMER_PROFILE_REC>
<nstrgmpr:PERSON_FIRST_NAME>          </nstrgmpr:PERSON_FIRST_NAME>
<nstrgmpr:PERSON_LAST_NAME>          </nstrgmpr:PERSON_LAST_NAME>
</nstrgmpr:P_CUSTOMER_PROFILE_REC> </nstrgmpr:InputParameters>
</CREATE_CUST_ACCOUNT_Input>
```


- **REST Header**: This region displays the values passed for the application context, including responsibility, application, NLS language, security group name, and Org Id, and the value passed for the Language parameter.

  These context values may be retrieved from the Context Parameters element in HTTP body for a REST request (JSON / XML).

- **Service Response**: This region contains the response sent date and time, and the message **View** link if available to view the actual payload of the response in XML or JSON.

## Viewing Log Messages

If the runtime logging is enabled for a specific service type of an interface or service in

the Log & Audit Setup Details page, log messages can be captured in Service Monitor against that instance for the specified service type of the interface.

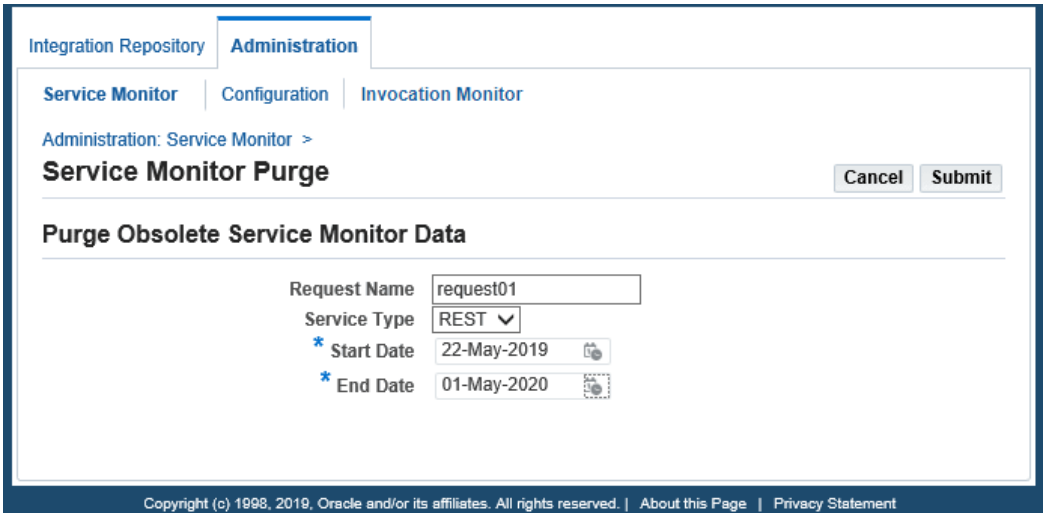
At runtime, each request is received and appears in Service Monitor with an instance ID and the **Log** icon. You can view the runtime log details by clicking the **Log** icon.

For information on viewing the runtime processing log messages in the Web Service Runtime Logs page, refer to Viewing Runtime Processing Logs, page 7-15.

## Purging SOAP and REST Messages, Audits, and Logs

Oracle E-Business Suite Integrated SOA Gateway lets you purge SOAP and REST messages, logs stored in the Oracle E-Business Suite database, and audit records that have been collected through Service Monitor for a period of time. Click **Purge** in the Service Monitor Search page to launch the Service Monitor Purge page.

***Service Monitor Purge Page***




Enter the following purge parameters in the Service Monitor Purge page:

- **Request Name**: Specify the Request Name for your request.
- **Service Type**: Select a desired service type from the drop-down list. It can be 'SOAP', 'REST', or 'Any'. By default, 'Any' is selected, meaning that both SOAP and REST service types are included.
- **Start Date**: Identify the start date of the date range for your purge.
- **End Date**: Identify the end date of the date range for your purge.

Click **Submit**. A request number will be automatically assigned to you for your purge request indicating that your request has been submitted for processing. Service Monitor will purge all monitored data for SOAP services, REST services, or both SOAP and

REST services within your specified date range.

The monitored requests and responses of the selected service type(s) will be purged for the specified date range in the following order of sequence:

1. Request details
2. Request and response payloads
3. Attachments (SOAP only)
4. Log messages from the Oracle E-Business Suite database

   This deletes only the logs for which the service is completed with a status of 'SUCCESS'. This does not delete the logs for the service with a 'FAILURE' status.

   The purge is based on the Completion Date of the service for the specified date range.
5. SOA composite instances from Oracle SOA Suite (SOAP only)

   > **Note:** For SOAP services, log messages retrieved from the server file system of Oracle SOA Suite cannot be purged. However, log messages coming from Oracle E-Business Suite service invocation which are stored in the log message table can be purged from the Oracle E-Business Suite database.

**To purge requests and responses:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility.

   From the navigation menu, select the **Service Monitor** link from the Administration section to open the Monitor Search page.
2. Click **Purge**.
3. Enter the following information in the Service Monitor Purge page:
   1. Enter the request name for your purge request.
   2. Select a desired Service Type value (SOAP, REST, or Any) for your purge.
   3. Enter the Start Date and End Date fields to specify the time range for your purge.
4. Click **Submit** to submit your purge request.

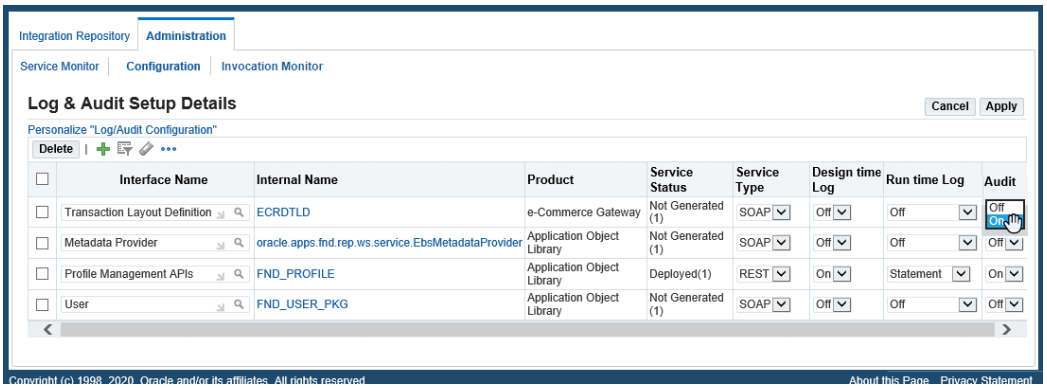
## Enabling Web Service Auditing Using the Configuration Subtab

In addition to searching and viewing service requests and responses, Service Monitor provides auditing feature letting you track message details such as requests, responses, and faults.

If the auditing feature for a specified service type of an interface is enabled, all incoming REST messages sent and received directly to Oracle E-Business Suite or all incoming SOAP messages for the service that Oracle SOA Suite processes along with the associated payloads, and fault messages can be saved and tracked in Service Monitor.

Note that the auditing feature is enabled at the service type level of an interface through the Log & Audit Setup Details page in the Configuration subtab.

***Log & Audit Setup Details Page for Enabling Auditing***




To enable the auditing feature, select a desired service type (SOAP or REST) of an interface that you want the feature to be enabled, and then select 'On' from the Audit drop-down list. Click **Apply** to save and validate the addition.

> **Note:** To enable the auditing feature for both the SOAP and REST service types of the same interface if both types are available for that interface, you need to create two separate configurations for each service type.

For more information on how to enable the auditing feature along with log configuration, see Adding a New Configuration, page 7-6.

# 9

# Implementing SOAP and REST Service Invocation Framework

## Overview

To invoke and consume external services from Oracle E-Business Suite, Oracle E-Business Suite Integrated SOA Gateway uses service invocation framework (SIF) that leverages Oracle Workflow Java Business Event System (JBES) and seeded Java rule functions to invoke SOAP and REST services.

By using this service invocation framework, developers or implementers can interact with SOAP and REST services through service metadata descriptions instead of working directly with APIs. This approach provides service access in a manner that is independent of protocol or location.

This framework allows updated implementations of a binding to be plugged at runtime. As a result, it not only facilitates a stubless or completely dynamic web service invocation, but also allows the calling service to defer choosing a service binding until runtime. More importantly, this enhances the seamless business integration between loosely-coupled applications and accelerates service invocation and consumption.

Service Invocation Framework has the following features:

- It supports various service invocation sources or points from an Oracle E-Business Suite instance. This includes:
  - PL/SQL Layer
    - Workflow Process
    - Any other PL/SQL code
    - Forms
  - Java Layer

    - OA Framework
    - Standalone Java Code

- It supports both SOAP and REST service invocations from Oracle E-Business Suite.

  **Note:** All outbound SOAP and REST service invocation messages through Service Invocation Framework can be monitored using Service Invocation Monitor. See: Monitoring and Managing Outbound Service Invocation Messages Using Service Invocation Monitor, page 10-1.

- It supports the Synchronous Request - Response, and One-way/Notification Only message patterns for both SOAP and REST services.
- It supports TLS-based web service invocation over HTTPS protocol.
- It supports web service (WS) security through UsernameToken-based service authentication for SOAP services and HTTP Basic Authentication for REST services.
- It supports passing values for any header part that may be required to embed applications context into a SOAP envelope or REST HTTP header.
- It supports document-based SOAP service invocation.

  **Note:** The document-based web service uses the form of XML with commonly agreed upon schema between the service provider and consumers as a communication protocol.

  While RPC (remote procedure call)-based web service is to invoke a cross-platform remote procedure call using SOAP, and this approach is not supported in this release.

- It invokes and consumes external REST services using GET or POST HTTP method with XML or JSON payload.
- It provides errors and exception handling, and the invocation retry feature.
- It provides the ability to test business event for service invocation.

To have a better understanding on how the service invocation framework invokes web services, the following topics are described in this chapter:

- Architecture Overview, page 9-3

- Implementing Service Invocation Framework, page 9-9

## Architecture Overview

Oracle Workflow is the primary process management solution within Oracle E-Business Suite; Oracle Workflow Business Event System, an essential component within Oracle Workflow, provides event and subscription features that help identify integration points within Oracle E-Business Suite.

The Business Event System consists of an Event Manager and workflow process event activities. The Event Manager lets you register subscriptions to significant events; event activities representing business events within workflow processes let you model complex business flows or logics within workflow processes.

When an event occurs, the Event Manager processes subscription to the event. Subscription processing can include running custom code on the event information, sending event information to a workflow process, and sending event information to other agents or systems.

For example, to invoke a web service through Oracle Workflow JBES, the description of WSDL URL representing the SOAP service (or the description of service endpoint for a REST service) must be consumed through the event subscription definition so that web service metadata can be parsed and stored as subscription parameters.

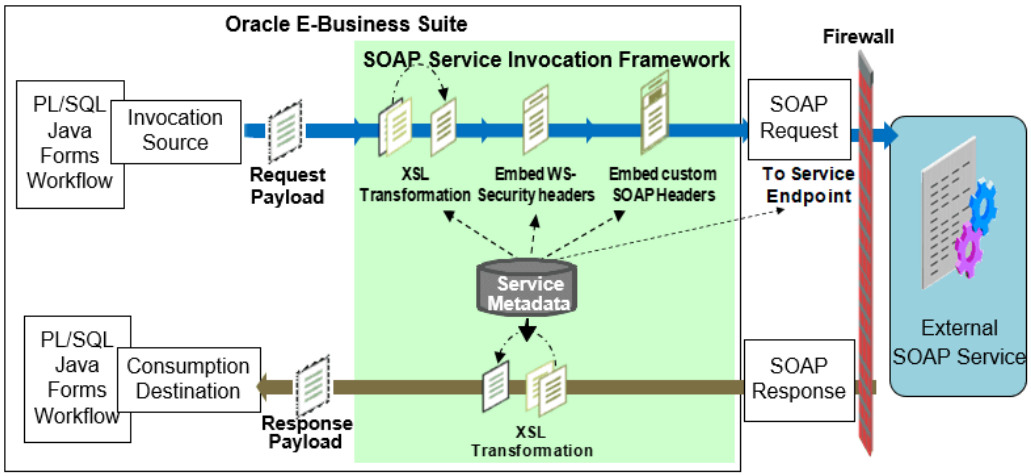
The runtime architecture diagrams for SOAP and REST services are explained respectively in the following sections:

- SOAP Service Invocation Runtime Architecture, page 9-3
- REST Service Invocation Runtime Architecture, page 9-7

### SOAP Service Invocation Runtime Architecture

To better understand the service invocation process for SOAP services, the following diagram illustrates the transactional architecture details of the runtime SOAP service invocation:

*SOAP Service Invocation Transactional Architecture Diagram*

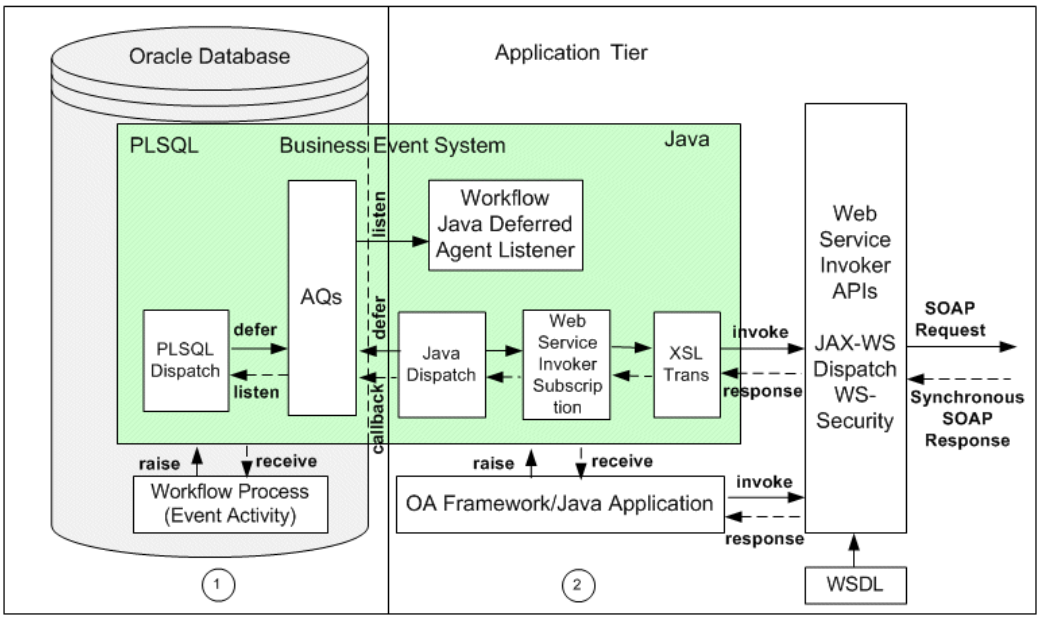



In this diagram, when an invoker event is raised at runtime, the event and subscription parameters are used to invoke a SOAP service by sending a SOAP request message. If this request or output message requires transformation in order to communicate with an external SOAP service, the XSL transformation on the output message is performed before invoking the service. If it is a synchronous request - response operation and the response is available, the XSL transformation on the input message can be performed if necessary in order to communicate or call back to Oracle E-Business Suite.

> **Note:** If event parameters are passed with the same names as the subscription parameters that have been parsed and stored, the event parameter values override the subscription parameters.

Furthermore, the following diagram provides the topology of various components that exchange information during the end-to-end service invocation from Oracle Workflow:

**SOAP Service Invocation Framework Architecture**




Service invocation framework from Oracle E-Business Suite is enabled though Oracle Workflow Java Business Event System and is based on the JAX-WS (Java API for XML-based Web Services) Dispatch from Oracle JRF (Java Required Files).

Oracle Workflow Business Event System is a workflow component that allows events to be raised from both the PL/SQL and Java layers. Therefore, the service invocation from Oracle E-Business Suite can be from both layers.

1. **Service Invocation from PL/SQL**

   1. Application raises a business event using PL/SQL API `WF_EVENT.Raise`.

      The event data can be passed to the Event Manager within the call to the `WF_EVENT.Raise` API, or the Event Manager can obtain the event data or message payload by calling the generate function for the event if the data or payload is required for a subscription.

      > **Note:** See the *Oracle Workflow API Reference* for information about `WF_EVENT.Raise` API.

   2. Oracle Workflow Business Event System (BES) identifies that the event has a subscription with the Java Rule Function `oracle.apps.fnd.wf.bes.WebServiceInvokerSubscription` for SOAP services.

   3. The Business Event System enqueues the event message to the WF_JAVA_DEFERRED queue. The Java Deferred Agent Listener then dequeues and processes the subscription whose Java rule function invokes the

SOAP service.

4. If callback event and agent parameters are mentioned, the SOAP service response is communicated back to Oracle E-Business Suite using the callback information. The Java Deferred Agent Listener process that runs on the Concurrent Manager (CM) tier invokes the service.

2. **Service Invocation from Java**

   1. Java Application raises a business event using Java method `oracle.apps.fnd.wf.bes.BusinessEvent.raise` either from the OA Framework page controller/AMImpl or Java code running on the Concurrent Manager (CM) tier.

   2. Since the event is raised in Java where the subscription's seeded Java Rule Function `oracle.apps.fnd.wf.bes.WebServiceInvokerSubscription` for SOAP services is accessible, whether the Java Rule Function is processed inline or deferred is determined by the phase of the subscription.

      - If the invoker subscription is created with Phase greater than or equal to 100, the event is enqueued to the WF_JAVA_DEFERRED queue.

      - If the invoker subscription is created with Phase less than 100, the event is dispatched inline.

        If the event is raised from the OA Framework page, the dispatch logic processes within `OACORE WebLogic Server`.

After an event is raised either using the PL/SQL API or Java method, the raised event can be processed in the following ways:

- If the raised event is dispatched immediately to the Java Business Event System, then the seeded Java Rule Function and its associated event subscription information will be retrieved and processed to invoke the SOAP service.

- If the raised event is enqueued to the WF_JAVA_DEFERRED queue, then the Java Deferred Agent Listener running on the concurrent tier will dequeue the event message and then dispatch the event to the Java Business Event System. The seeded Java Rule Function and its associated event subscription information will then be retrieved and processed to invoke the service.

While invoking the web service, the seeded Java Rule Function first reads the SOAP service metadata created for the subscription.

If web service input message requires transformation, the Java Rule Function performs XSL transformation on the request message generated during the event creation by using a PL/SQL API `ECX_STANDARD.perform_xslt_transformation`. Next, the Java Rule Function invokes the service.

> **Note:** For detailed information on the XSL transformation PL/SQL API, see Execution Engine APIs, *Oracle XML Gateway User's Guide*.

If it is for the synchronous request - response operation, when the response message is available and XSL transformation is required on the web service output message, XSL transformation on the output (response) message will be performed.

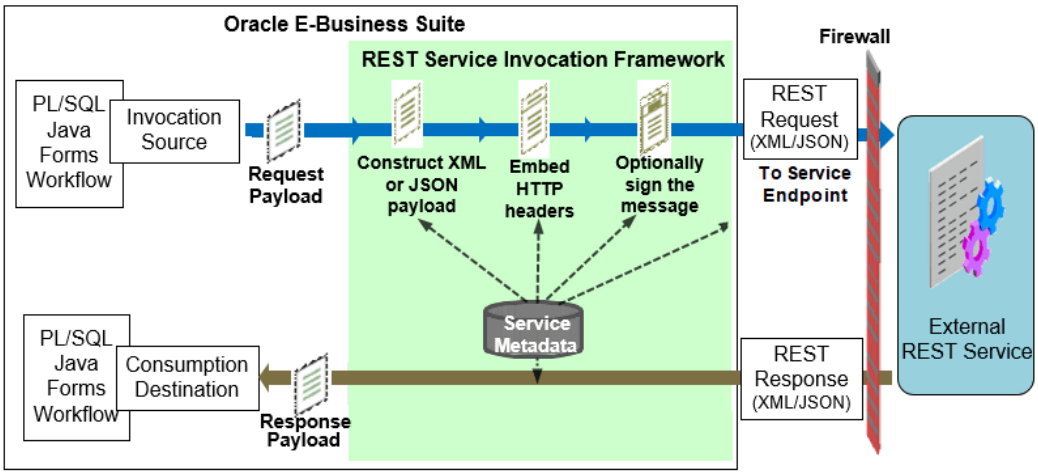
If callback information is provided, perform callback by either raising a business event or by enqueuing the event to a given workflow agent with the response message as payload.

> **Note:** For the service invocation from Java code, if the web service invoker subscription is synchronous with subscription phase less than 100, then the web service is invoked as soon as the event is raised, and if it's successful, the response is available immediately by using the method `getResponseData()` on the `BusinessEvent` object.

## REST Service Invocation Runtime Architecture

The runtime architecture for REST service invocation framework can be depicted in the following diagram:

***REST Service Invocation Transactional Architecture Diagram***




In this diagram, when an invoker event is raised at runtime, event and subscription parameters are used to invoke an external REST service by sending a REST request message with payload. The REST service invocation framework will construct the request payload with XML or JSON format, embed the HTTP headers, and optionally add the signing message as part of the HTTP request before invoking the REST service. If it is a synchronous request - response operation and the response is available, the response message in XML or JSON format communicates or calls back to Oracle E-

Business Suite.

> **Note:** If event parameters are passed with the same names as the subscription parameters that have been parsed and stored, the event parameter values override the subscription parameters.

Similar to the service invocation framework for SOAP services, REST service invocation from Oracle E-Business Suite can be from a PL/SQL or Java layer.

1. **Service Invocation from PL/SQL**

   1. Application raises a business event using PL/SQL API `WF_EVENT.Raise`.

      The event data can be passed to the Event Manager within the call to the `WF_EVENT.Raise` API, or the Event Manager can obtain the event data or message payload by calling the generate function for the event if the data or payload is required for a subscription.

      > **Note:** See the *Oracle Workflow API Reference* for information about `WF_EVENT.Raise` API.

   2. Oracle Workflow Business Event System (BES) identifies that the event has a subscription with the Java Rule Function `oracle.apps.fnd.wf.bes.RESTServiceInvokerSubscription` for REST services.

   3. The Business Event System enqueues the event message to the WF_JAVA_DEFERRED queue. The Java Deferred Agent Listener then dequeues and processes the subscription whose Java rule function invokes the REST service.

   4. If callback event and agent parameters are mentioned, the service response is communicated back to Oracle E-Business Suite using the callback information. The Java Deferred Agent Listener process that runs on the Concurrent Manager (CM) tier invokes the web service.

2. **Service Invocation from Java**

   1. Java Application raises a business event using Java method `oracle.apps.fnd.wf.bes.BusinessEvent.raise` either from the OA Framework page controller/AMImpl or Java code running on the Concurrent Manager (CM) tier.

   2. Since the event is raised in Java where the subscription's seeded Java Rule Function `oracle.apps.fnd.wf.bes.RESTServiceInvokerSubscription` for REST services is accessible, whether the Java Rule Function is processed inline or deferred is determined by the phase of the subscription.

- If the invoker subscription is created with Phase greater than or equal to 100, the event is enqueued to the WF_JAVA_DEFERRED queue.

- If the invoker subscription is created with Phase less than 100, the event is dispatched inline.

  If the event is raised from the OA Framework page, the dispatch logic processes within `OACORE WebLogic Server`.

After an event is raised either using the PL/SQL API or Java method, the raised event can be processed in the following ways:

- If the raised event is dispatched immediately to the Java Business Event System, then the seeded Java Rule Function and its associated event subscription information will be retrieved and processed to invoke the REST service.

- If the raised event is enqueued to the WF_JAVA_DEFERRED queue, then the Java Deferred Agent Listener running on the concurrent tier will dequeue the event message and then dispatch the event to the Java Business Event System. The seeded Java Rule Function and its associated event subscription information will then be retrieved and processed to invoke the service.

While invoking the service, the seeded Java Rule Function first reads the REST service metadata created for the subscription.

If callback information is provided, perform callback by either raising a business event or by enqueuing the event to a given workflow agent with the response message as payload.

> **Note:** For the service invocation from Java code, if the web service invoker subscription is synchronous with subscription phase less than 100, then the web service is invoked as soon as the event is raised, and if it's successful, the response is available immediately by using the method `getResponseData()` on the `BusinessEvent` object.

## Implementing Service Invocation Framework

This section discusses the following topics:

## Setup Tasks

Web services can be invoked from any one of the following tiers:

- **OACORE WebLogic Server**: Web service invocations from the OA Framework page using a synchronous event subscription (phase less than 100) are processed from the OACORE WebLogic Server.

- **Concurrent Manager (CM) Tier JVM**: The following web service invocations are processed from the CM tier JVM within the Java Deferred Agent Listener that runs within Workflow Agent Listener Service:

  - Invocations from PL/SQL either through synchronous or asynchronous event subscriptions

  - Invocations from Java/OA Framework through asynchronous event subscriptions

- **Standalone JVM**: Web service invocations from a Java process that runs outside OACORE or CM using a synchronous event subscription are processed from within that JVM.

**Proxy Host and Port Setups**

If a target web service resides within the firewall and is directly accessible from an Oracle E-Business Suite server, administrators do not need to configure proxy host and port.

However, if a target web service that is invoked resides outside the firewall and thus the request needs to be routed through the proxy, in this circumstance, administrators must set up and configure proxy host and port appropriately for the tiers that web service invocations occur in order to perform the following activities:

- Parse and consume a SOAP WSDL or REST WADL URI during subscription definition

- Invoke web service from subscription definition

### Common Proxy Setup at the WebLogic Server and Concurrent Manger Tier JVM

Use common setup information to configure proxy host and port. This information is applicable to the following conditions:

- **Proxy host and port at the WebLogic Server**

  For a web service invoked from OA Framework, the JBES seeded Java Rule Function would run within the OACORE WebLogic Server.

  The WebLogic Server start script (`<EBSDomain>/bin/startWebLogic.sh`) should have the following system properties setup in the `JAVA_OPTIONS` in order

for it to work:

```
-Dhttp.proxyHost=myproxy.host.name
-Dhttp.proxyPort=80
-Dhttp.nonProxyHosts=*.example.com|localhost
```

- **Proxy host and port at the Concurrent Manger Tier JVM**

  For a web service invoked from a PL/SQL or Java layer using an asynchronous subscription, the event is raised by the application code and then is enqueued to the WF_JAVA_DEFERRED queue by the Event Manager. The event subscription is processed from the CM tier by the Java Deferred Agent Listener.

  If a web service is invoked by the Java Deferred Agent Listener, then the code would run within the CM tier Java service's JVM. If the web service resides outside the firewall, the proxy host and port need to be configured properly.

To configure the proxy host and port for the Oracle WebLogic Server and CM tier JVM, you need to update AutoConfig context file with the following entries and run AutoConfig:

```
<!-- proxy -->
  <proxyhost oa_var="s_proxyhost">myproxyhost</proxyhost>
   <proxyport oa_var="s_proxyport">80</proxyport>
   <nonproxyhosts oa_var="s_nonproxyhosts">any domain that needs to be
by-passed (such as *.us.example.com)</nonproxyhosts>
```

This configuration would apply to all programs and services on the CM tier JVM. If you want to set the proxy only for the Workflow container, perform the following steps:

1. Log in to Oracle E-Business Suite as a user who has the **Workflow Administrator Web Applications** responsibility.

2. Select the **Oracle Applications Manager** link from the Navigator and then click the **Workflow Manager** link.

3. Click the **Agent Listener** radio button.

4. Click the **Workflow Java Deferred Agent Listener** radio button.

5. Click the **Workflow Agent Listener Service** row.

6. Select "Workflow Agent Listener Service" and click **Edit**.

7. Click **Edit Service Parameters**.

8. Specify the following value:

   ```
   SVC_PROXY_SET=true:SVC_PROXY_HOST=<your proxy server>:
   SVC_PROXY_PORT=<your proxy port>
   ```

9. Restart the CM container using the `adcmctl.sh` script in `$ADMIN_SCRIPTS_HOME`

.

#### Proxy Host and Port Setup When Using Standalone Java Class

You must set the following entries:

```
java -Dhttp.proxyHost=myproxyhost -Dhttp.proxyPort=80 classname
```

## Setup Tasks for Invoking TLS-based Web Services Over HTTPS

Service invocation framework supports TLS-based web service invocation using Server Authentication method. When a client connects to a web server via HTTPS, the server sends back its server certificate to the client for verification. Once verified, the client sends the data, encrypted, to the server. Server Authentication allows the client to identify the server. Before invoking a web service from a server over HTTPS (HTTP protocol over TLS), you need to perform manual setup tasks in order to read the TLS-based WSDLs and invoke the TLS service endpoints.

A client may receive one of the following two types of server certificates:

- Public certificate and it is issued by a Certification Authority (CA)
- Self-signed certificate or certificate is not in trusted certificate list

Perform the following setup tasks for the service invocation framework to invoke a TLS-based web service:

- Importing Server TLS Certificate into a SIF JVM's Certificate Store, page 9-12
- Setting Up TLS Proxy Host and Port, page 9-15
- Performing Additional Setup Tasks, page 9-16

#### Importing Server TLS Certificate into a SIF JVM's Certificate Store

***Public Certificate Issued by a Certification Authority (CA)***

If server certificate is a public certificate and is issued by a public CA such as VeriSign, then it is most likely available in a SIF JVM's certificate store or in a trusted certificate list.

***Self-signed Certificate or Certificate is not in Trusted Certificate List***

Perform the following tasks to import the server's TLS certificate into a SIF JVM's certificate store or add it to a trusted certificate list:

1. **Export** the server certificate using either one of the following methods:
   - **Use `openssl` Utility:**

     Use **`openssl`** utility to connect to the destination server with the following syntax:

```
$ openssl s_client -connect <server>:<port> -showcerts
```

> **Important:** If there is no port in destination, default HTTPS port 443 should be used.
>
> For example: `$ openssl s_client -connect host.example.com:443 -showcerts`

Copy the certificate content from `BEGIN CERTIFICATE` to `END CERTIFICATE` (including BEGIN CERTIFICATE and END CERTIFICATE lines as shown in the sample certificate) into a file and save the file (such as `my_cert.cer`).

A sample output of above **`openssl`** command can be like:

```
$ openssl s_client -connect host.example.com:443 -showcerts

...
Server certificate
-----BEGIN CERTIFICATE-----
MIIFVjCCBD6gAwIBAgIQBVWzfUyIcCa5LtuV+f9WvjANBgkqhkiG9w0BAQUFADCB
sDELMAkGA1UEBhMCVVMxFzAVBgNVBAoTDlZlcmlTaWduLCBJbmMuMR8wHQYDVQQL
ExZWZXJpU2lnbiBUcnVzdCBOZXR3b3JrMTswOQYDVQQLEzJUZXJtcyBvZiB1c2Ug
YXQgaHR0cHM6Ly93d3cudmVyaXNpZ24uY29tL3JwYSAoYykwNTEqMCgGA1UEAxMh
VmVyaVNpZ24gQ2xhc3MgMyBTZWN1cmUgU2VydmVyIENBMB4XDTA5MDQyMTAwMDAw
MFoXDTEwMDUwNTIzNTk1OVowgckxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxp
Zm9ybmlhMRcwFQYDVQQHFA5SZWR3b29kIFNob3JlczEbMBkGA1UEChQST3JhY2xl
IENvcnBvcmF0aW9uMR8wHQYDVQQLFBZJbmZvcm1hdGlvbiBUZWNobm9sb2d5MTMw
MQYDVQQLFCpUZXJtcyBvZiB1c2UgYXQgd3d3LnZlcmlzaWduLmNvbS9ycGEgKGMp
MDUxGTAXBgNVBAMUECoub3JhY2xlY29ycC5jb20wgZ8wDQYJKoZIhvcNAQEBBQAD
gY0AMIGJAoGBAL/EBxxt2keWTuJbo4SogWmiaJxThYDMvy8nWkpvKIp3s7OCQW0G
t17sAirfBkUirbGRlcWi5fi0RReruGXgYxFnf12fBNAimRWVo3mjeQo8BpRBm27n
3YcTZUlaIE77FvB3913jzD9c4sbjIe2fGpVmx+X9PZmDKSY9KPGjDbFNAgMBAAGj
ggHTMIIBzzAJBgNVHRMEAjAAMAsGA1UdDwQEAwIFoDBEBgNVHR8EPTA7MDmgN6A1
hjNodHRwOi8vU1ZSU2VjdXJlLWNybC52ZXJpc2lnbi5jb20vU1ZSU2VjdXJlMjAw
NS5jcmwwRAYDVR0gBD0wOzA5BgtghkgBhvhFAQcXAzAqMCgGCCsGAQUFBwIBFhxo
dHRwczovL3d3dy52ZXJpc2lnbi5jb20vcnBhMB0GA1UdJQQWMBQGCCsGAQUFBwMB
BggrBgEFBQcDAjAfBgNVHSMEGDAWgBRv7K+g3Yqk7/UqEGctP1WCvNfvJTB5Bggr
BgEFBQcBAQRtMGswJAYIKwYBBQUHMAGGGGh0dHA6Ly9vY3NwLnZlcmlzaWduLmNv
bTBDBggrBgEFBQcwAoY3aHR0cDovL1NWUlNlY3VyZS1haWEudmVyaXNpZ24uY29t
L1NWUlNlY3VyZTIwMDUtYWlhLmNlcjBuBggrBgEFBQcBDARiMGChXqBcMFowWDBW
FglpbWFnZS9naWYwITAfMAcGBSsOAwIaBBRLa7kolgYMu9BSOJsprEsHiyEFGDAm
FiRodHRwOi8vbG9nby52ZXJpc2lnbi5jb20vdnNsb2dvMS5naWYwDQYJKoZIhvcN
AQEFBQADggEBADBi9NfljQLuD2Tnol3pmQl717rc8kKmpLYEO6u5MxIK0+L2MslV
4NE1qbNx1dfIoW68HHXtpsF5KtKFLYk9EoOkBd7oMp7fFv31RANV3LpdAHZC9EaK
CA/oKB2RrSu7ZmaUvoRb+3v5FdhAmgtoY6Wljk0yxMvXVf/TOeXqKl8C/r1gSzyC
s/jVmy6N81Oeleqtozzt/aJNGu7xu/MdtP13eyu7RSEBRGJwEwTXH+rTUKK8mle0
Kz15DgJ6ByK2XZmD4Z+O8DTUhUhIHR1OhuLR7zjGp9W7wQuCizUcTvuKEGzVf5D/
y7orhV0U+AoXnl/5wntVMZc/Tmqr/Fkb8+g=
-----END CERTIFICATE-----

...
```

- **Use Web Browser:**

  Access the WSDL file available through HTTPS URL (such as `https://<hostname>:<port>/webservices/SOAProvider/xmlgateway/ont__poi/?wsdl`) or a REST service endpoint (such as `https://<hostname>:<port>/webservices/rest/fnduserpkg/testusername/`) through a

web browser.

1. After the WSDL file or the REST service endpoint has been successfully loaded in a browser, double click the **Lock** icon in the bottom right corner of the browser and export the certificate.

   For example, in Internet Explorer, double click the **Lock** icon >**Details** > **Copy to File**. In Mozilla Firefox, double click the **Lock** icon >**Security** > **View Certificate**>**Details** >**Export**.

2. You can also use the browser menu to access the certificate. For example, in Internet Explorer, select **Internet Options** from the **Tools** drop-down menu to open the Internet Options pop-up window. Select the Content tab, click **Certificates**. Select the Personal (or Other People) tab to select your certificate and click **Export**.

3. You can export or save the certificate either in DER encoded binary X.509 (. CER) or in Base 64 encoded.

   **Note:** Different browser versions may have different steps to export TLS certificates.

2. **Import** the server's TLS certificate into an appropriate SIF JVM's certificate store to add it to the list of trusted certificates.

   **Important:** Information about where web services are invoked through the service invocation framework is described in the Setup Tasks, page 9-10.

   There are many utilities available to import certificates. For example, you can use **`keytool`**, a key and certificate management utility that stores the keys and certificates in a *keystore*. This management utility is available by default with JDK to manage a *keystore* (database) of cryptographic keys, X.509 certificate chains, and trusted certificates.

   The **`keytool`** commands have the following syntax:

   ```
   keytool -import -trustcacerts -keystore <key store location> -
   storepass <certificate store password> -alias <alias name> -
   file <exported certificate file>
   ```

   For example:

   ```
   keytool -import -trustcacerts -keystore
   "$AF_JRE_TOP/jre/lib/security/cacerts" -storepass password -
   alias xabbott_bugdbcert -file my_cert.cer
   ```

   **Note:** This must be typed as a single line. The file (`-file`) is the

exported certificate file, such as `my_cert.cer`.

### Setting Up TLS Proxy Host and Port

If a TLS-based web service resides outside the firewall, the JVM that invokes the web service has to communicate through TLS proxy. Following setup tasks are required in all appropriate tiers to use TLS proxy.

#### Setting Up Proxy Host and Port at WebLogic Server

For a web service invoked from OA Framework, the JBES seeded Java Rule Function would run within the OACORE WebLogic Server.

WebLogic Server start script (`<EBSDomain>/bin/startWebLogic.sh`) should have the following system properties setup in `JAVA_OPTIONS` in order for it to work:

```
-Dhttps.proxyHost=myproxy.host.name
-Dhttps.proxyPort=80
-Dhttps.nonProxyHosts=*.example.com|localhost
```

AutoConfig does not currently support properties `https.proxyHost` and `https.proxyPort`. To ensure the above properties are retained during the process of AutoConfig, the context file could be customized to add these two properties.

For information on how to customize AutoConfig-managed configurations, see *Using AutoConfig to Manage System Configurations in Oracle E-Business Suite Release 12*, My Oracle Support Knowledge Document 387859.1.

#### Setting Up Proxy Host and Port at Concurrent Manger Tier JVM

For a web service invoked from PL/SQL and Java using an asynchronous subscription, the event is raised by the application code wherever it runs and then the event is enqueued to the WF_JAVA_DEFERRED queue by the Event Manager. The event subscription is processed from the CM tier by the Java Deferred Agent Listener.

If a web service is invoked by the Java Deferred Agent Listener, then the code would run within the CM tier Java service's JVM. Workflow Agent Listener Service does not currently support Service Parameters to set the TLS proxy. The TLS proxy could be set up directly to the Concurrent Manager's JVM system properties in `$APPL_TOP/admin/adovars.env` using AutoConfig.

```
<oa_environment type="adovars">
 <oa_env_file type="adovars" oa_var="s_adovars_file" osd="unix">
  $APPL_TOP/admin/adovars.env</oa_env_file>
...
 <APPSJREOPTS oa_var="s_appsjreopts">="-Dhttps.proxyHost=[proxyhost]
  -Dhttps.proxyPort=[sslproxyport]</APPSJREOPTS>
...
</oa_environment>
```

#### Setting Up Proxy Host and Port When Using Standalone Java Class

You must set the following entries:

```
java -Dhttps.proxyHost=[proxyhost] -Dhttps.proxyPort=[sslproxyport]
<classname>
```

#### Performing Additional Setup Tasks

Additionally, performing the following tasks to invoke services with TLS 1.2 only and TLS 1.2 with backward compatibility:

1. Apply Patch 22612527 with the prerequisite Patch 13866584 to the FMW home ( `FMW_HOME`).

2. Update the 32-bit JDK 7 under `$OA_JRE_TOP` with the Java Cryptography Extension (JCE) updates from the following page: https://www.oracle.com/java/technologies/javase-jce7-downloads.html

3. Update the 64-bit JDK 7 under the directory referenced by the `s_fmw_jdktop` context variable with the Java Cryptography Extension (JCE) updates.

4. Update the Oracle E-Business Suite context variables using Oracle Applications Manager.

   1. Log in to Oracle E-Business Suite as a user who has the **Workflow Administrator Web Applications** responsibility.

   2. Select the **Oracle Applications Manager** link from the Navigator, and then select **AutoConfig**.

   3. Select the application tier context file, and choose **Edit Parameters**.

   4. Update the following context variables:

      - `s_afjsmarg = -Dhttps.protocols=TLSv1,TLSv1.1,TLSv1.2 or -Dhttps.protocols=TLSv1.2`

        - To enable TLS 1.2 with backward compatibility, add the following:

          ```
          s_afjsmarg = -Dhttps.protocols=TLSv1,TLSv1.1,TLSv1.2
          ```

        - To enable TLS 1.2 only, add the following:

          ```
          s_afjsmarg = -Dhttps.protocols=TLSv1.2
          ```

      - `s_proxyhost = fully qualified host.domain name`

      - `s_proxyport = port value`

      - `s_proxybypassdomain = domain name (For example, example.com)`

      - `s_nonproxyhosts = wildcard domain name (For example, *.example.com)`

5. Run AutoConfig using the `adautocfg.sh` script in the application tier `$ADMIN_SCRIPTS_HOME` directory.

6. Run the `adstpall.sh` script and the `adstrtal.sh` script in the same `$ADMIN_SCRIPTS_HOME` directory to stop and restart all services.

For more information about enabling TLS in Oracle E-Business Suite Release 12.2, see My Oracle Support Knowledge Document 1367293.1.

## Implementing Service Invocation Framework for SOAP Services

Service invocation framework allows you to invoke or consume external SOAP services from Oracle E-Business Suite. To successfully achieve this, both integration administrators and integration developers need to collaboratively perform and implement certain tasks. Specifically, an integration administrator needs to perform the following tasks:

- Understanding and Configuring WS-Security for the SOAP Service Invocation Framework, page 9-17
- Managing SOAP Service Errors, page 9-24
- Understanding Implementation Limitation and Consideration for the SOAP Service Invocation Framework, page 9-25

An integration developer needs to perform the following tasks:

- Defining metadata for the SOAP service to be invoked
- Calling back to Oracle E-Business Suite with SOAP service responses (optional)
- Testing SOAP service invocation
- Troubleshooting SOAP service invocation failure
- Extending seeded Java rule function for SOAP services

For more information about these tasks, see Using Service Invocation Framework to Invoke SOAP Services, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

### Understanding and Configuring WS-Security for the SOAP Service Invocation Framework

Web service security (WS-Security) is a communication protocol providing a means for applying security to web services. It describes enhancements to SOAP messaging to provide quality of protection through message integrity and single message authentication. It also describes how to attach security tokens to SOAP messages to enhance security features.

The SOAP service invocation framework supports WS-Security in a general-purpose mechanism for associating security tokens with messages to authenticate web service requests and service invocations from Oracle E-Business Suite.

To accomplish this goal, the SOAP service invocation framework supports WS-Security

through UsernameToken based security. The following sections explain the UsernameToken based security and the security configuration through the event subscription user interface:


- UsernameToken Based Security, page 9-18
- Configuring Web Service Security Through Event Subscription User Interface, page 9-19
  - Configuring Security Password with Customization Level, page 9-19
  - Specifying Expiration Time Parameter for the Security Header, page 9-23


### UsernameToken Based Security

This security mechanism authenticates the user invoking a SOAP service by passing a *user name* and an optional *password* in the SOAP Header of a SOAP request sent to the web service provider.

The user name and password information discussed here is the concept of Oracle E-Business Suite user name and password.

If the SOAP service that is invoked enforces Username/Password based authentication, then the SOAP service invocation framework also supports the UsernameToken based WS-Security header during the SOAP service invocation.

> **Note:** A SOAP request invoking a web service should include a security header consisting of a user name and plain text password. The password received as part of the SOAP request at runtime will be validated against the encrypted password stored in Oracle E-Business Suite. After validation, the plain text password from the SOAP request will be discarded.

***User name*** is a clear text. ***Password*** is the most sensitive part of the UsernameToken profile. The SOAP service invocation framework supports the UsernameToken based WS-Security during the service invocation with a user name and an optional password with Type `PasswordText`.

For example, a WS-Security header with UsernameToken can be like:

```
<wsse:Security>
...
  <wsse:UsernameToken wsu:Id="UsernameToken-1"
 xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
  <wsse:Username>myUser</wsse:Username>
  <wsse:Password
 Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-
token-profile-1.0#PasswordText">password</wsse:Password>
  <wsse:Nonce
 EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
soap-message-security-1.0#Base64Binary">RDyVo/jbXJdSKuVEPrQW6Q==</wsse:
Nonce>
  <wsu:Created>2013-09-02T04:56:48.597Z</wsu:Created>
 </wsse:UsernameToken>
</wsse:Security>
```

- The `PasswordText` password type is the password written in clear text. There is another password type called 'PasswordDigest' which is a base64-encoded SHA-1 hash value of the UTF8-encoded password and this type of password is not supported in this release.

- `<Nonce>`: This element is a unique, random string that identifies the password. This helps protect the UsernameToken security from being reused during a replay attack.

- `<Created>`: This element indicates the creation time of the security.

### Configuring Web Service Security Through Event Subscription User Interface

To easily maintain UsernameToken based security, the SOAP service invocation framework allows you to configure the security password through the design-time user interface.

After an integration developer enters web service details in the Create Event Subscription - Invoke Web Service wizard, the Web Service Security region is shown letting the developer specify or update the user name and corresponding password if appropriate. The information will then be stored in Vault securely.

#### Configuring Security Password with Customization Level

Oracle Workflow allows various levels of updates on business event. Each event and subscription is assigned a customization level that determines whether the event data can be updated or not. The customization level is used to protect Oracle E-Business Suite seed data and to preserve your customizations in an upgrade.

Event and subscription can have one of the following customization levels:

- Core - No changes can be made to the event and subscription definition. This level is used only for events seeded by Oracle E-Business Suite.

- Limit - The event status can be updated to Enabled or Disabled, but no other changes can be made to the event definition. This level is used only for the events seeded by Oracle E-Business Suite.

- User - Any property in the event and subscription definition can be updated. This level is automatically set for events that you define.

**Configuring Security Information Between Instances**

When configuring web service security with the consideration of moving event subscription definitions between instances, whether you can enter or update the security information is based on the customization level as explained in the following:

- **Customization Level – User**

  If an invoker subscription with a Customization Level of User is created in the target environment, the complete definition is editable in that environment and also in the environments to which the definition is uploaded.

  While WS-Security Username can be created in one environment and moved to another, password has to be configured in each target environment before it can be used to invoke that service.

- **Customization Level – Limit or Core**

  If an invoker subscription with a Customization Level of Limit or Core is uploaded to the target environment, following are the options.

  - *User Name Configured*

    If a user name is configured for the web service, a Workflow Administrator can have access to Oracle Workflow Business Event Manager and the invoker subscription, and can update the password for that user in the target environment. This can be achieved by logging in to Oracle E-Business Suite with the Workflow Administrator Web responsibility. Select **Business Events** from the Navigator and choose **Subscriptions** in the horizontal navigation. Search and locate the invoker event subscription and then update the password.

    User name cannot be updated.

  - *User Name Not Defined*

    A Workflow Administrator can configure both user name and password by accessing the Workflow Business Event Manager.

    **Note:** On the system side, Module and Key values to store the password in Vault are derived in the target environment.

    - Module – Module name for Vault will be derived from the

business event and restricted to 30 characters.

- Key – Key value for Vault will be derived from the business event and user name.

The following information will be stored internally as part of the subscription definition if not already available for the invoker subscription:

- WFBES_SOAP_USERNAME=<entered username>
- WFBES_SOAP_PASSWORD_MOD=<derived module name>
- WFBES_SOAP_PASSWORD_KEY=<derived key name>

#### Examples of Configuring WS-Security with Different Subscription Customization Levels

*Scenario 1:*

*Define a new business event and subscription with a Customization Level of User in the source environment and have both user name and password manually entered to invoke a web service that requires WS-Security. Move the event and subscription defined earlier to a target environment and configure the WS-Security if required.*

*Solution:*

In this scenario, both the Username and Password fields are editable in the target environment. The Username field value is automatically populated and the Password field value is not available. You can update the new user name (optional) and a corresponding password if needed.

Use the following steps to configure WS-Security in the target instance:

1. Perform all the steps described in the following topics to define a new event and a subscription with security user name and password:
   - Creating a Web Service Invoker Business Event, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*
   - Creating a Local Subscription to the SOAP Service Invoker Event, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*
2. Download the event and subscription using the Workflow XML Loader and upload them to the target environment.

   **Note:** The Workflow XML Loader is a command line utility that lets you upload and download XML definitions for the Business Event

System objects between a database and a flat file. For information on downloading and uploading events using the Workflow XML Loader, see:

- Locate and Download Business Events, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*
- Upload Annotated File to the Database, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*

3. Search and locate the invoker business event you defined earlier, such as `oracle.apps.xxx.user.webservice.invoke`, in the source instance and click the **Subscription** icon from the result table.

4. Click the **Update** icon for the subscription. All fields are updatable because of the customization level of User. Click **Next** to the last stop of the Update Subscription - Invoke Web Service page.

   In the Web Service Security region, both the Username and Password fields are editable.

   - The Username field, such as `weblogic`, is automatically populated based on the user name defined earlier in the source environment in Step 1.
   - The Password field value is not available.

   You can update the user name if desired and enter a corresponding password for the web service. Click **Apply**.

*Scenario 2:*

*All Oracle E-Business Suite products provide seeded events and subscriptions with a Customization Level of Limit or Core for service invocation. The user name may or may not be configured during the subscription creation for the product-specific seeded events to invoke web services that require WS-Security.*

*When using the seeded events and subscriptions in the target instance of Oracle E-Business Suite Release 12.2, configure the WS-Security by entering a user name, if not already provided by the subscription owner, and the corresponding password for that user to be used for service invocation.*

*Solution:*

In the Oracle E-Business Suite Release 12.2 target instance, log in as a user who has the 'Workflow System Administrator' role, such as `sysadmin`. The Username field is not updatable if the user name is already provided by the subscription owner. You can always enter an associated password for the user to be used for the service invocation.

Use the following steps to configure WS-Security in the target instance:

1. Log in to Oracle E-Business Suite 12.2 target instance. Search and locate the product-specific seeded event and click the **Subscription** icon from the result table.

2. Click the **Update** icon for the subscription to load the Subscription details. All fields are disabled except the Status field because the Customization Level is set to Limit.

   Click **Next** to the last stop of the Update Subscription - Invoke Web Service page.

   In the Web Service Security region, enter the following security information:

   - Username: Enter a user name if it is not part of the seeded subscription definition.

     If the user name is entered by the Subscription owner as part of the seeded definition, this field would show the user name value with no option to edit it. The user needs to only enter the password.

   - Password: Enter a corresponding password for WS-Security.

   - Repeat Password: Enter the same password that you entered in the Password field.

   Click **Apply**. With WS-Security configured, the web service is ready to be invoked.

For more information about using customization level for an event, see Reviewing the Customization Level and License Status for an Event, Managing Business Events, *Oracle Workflow Developer's Guide*.

#### Specifying Expiration Time Parameter for the Security Header

When creating the subscription to the Invoker event, you can add the following parameter in the Web Service Invoker Parameters region to set the expiration time for the security header. This helps protect the header from being reused during a replay.

- WFBES_SOAP_EXPIRY_DURATION

By default, the header is set to expire 60 seconds in the `<wsu:Timestamp>` element (with `<wsu:Created>` and `<wsu:Expires>`) after it is created. When a different time is specified in the WFBES_SOAP_EXPIRY_DURATION parameter, it overrides the default 60 seconds expiration time for the header.

```
<wsse:Security soapenv:mustUnderstand="1"
 xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-secext-1.0.xsd">
  <wsu:Timestamp wsu:Id="Timestamp-2"
 xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
   <wsu:Created>2013-09-02T04:56:59.592Z</wsu:Created>
   <wsu:Expires>2013-09-02T04:57:59.592Z</wsu:Expires>
  </wsu:Timestamp>
 <wsse:UsernameToken>
...
 </wsse:UsernameToken>
</wsse:Security>
```

Similar to other subscription parameters added in this region, if a different expiration

time is passed as the event parameter, then the event parameter overrides the subscription parameter.

## Managing SOAP Service Errors

The SOAP service invocation framework uses the same way of handling errors in the Business Event System to manage errors if occur during the implementation of business event subscriptions. If the service invocation returns a fault message, the event is enqueued to an error queue to trigger error processing. If an exception occurred during the invocation process is due to service unavailability, the service faults should be logged and error subscription should be invoked.

To effectively process runtime exceptions for the events that are enqueued to an error queue, the SOAP service invocation framework uses the following event ERROR process to specifically trigger error processing during the service invocation:

- DEFAULT_EVENT_ERROR2: Default Event Error Process (One Retry Option)

  **Note:** The DEFAULT_EVENT_ERROR2 Error workflow process is created under WFERROR itemtype.

For example, if there is a runtime exception when the Workflow Java Deferred Agent Listener processes an event subscription to invoke a web service, the event is enqueued to the WF_JAVA_ERROR queue. If the event has an Error subscription defined to launch the Error workflow process `WFERROR:DEFAULT_EVENT_ERROR2`, the Workflow Java Error Agent Listener processes the error subscription which sends a notification to SYSADMIN with the web service definition, error details, and event details. Since Oracle Workflow default event error handler provides options for SYSADMIN to retry the web service invocation process after verifying that the reported error has been corrected, SYSADMIN can invoke the web service again from the notification if necessary.

However, if there is a runtime exception when invoking the web service by raising the Invoker event with synchronous subscription (phase less than 100), the exception thrown to the calling application. It is the responsibility of the calling application to manage the exception.

**Enabling Error Processing During Service Invocation**

To enable the error processing feature during the service invocation, you must create an Error subscription with the following values:

- 'Error' source type
- 'Launch Workflow' action type
- 'WFERROR:DEFAULT_EVENT_ERROR2' workflow process

To access the Create Event Subscription page, log in to Oracle E-Business Suite as a user

who has the Workflow Administrator Web Applications responsibility. Select **Business Events** from the navigation menu and choose the **Subscriptions** subtab. In the Event Subscriptions page, click **Create Subscription**.

For detailed information on how to create an error subscription for service invocation, see Create an Error subscription with 'Launch Workflow' Action Type, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

### Understanding Implementation Limitation and Consideration for the SOAP Service Invocation Framework

While implementing the SOAP service invocation framework, consider the following limitations:

- WFBES_INPUT_`<partname>` Parameter Can Only be Passed at the Event Raise

  The SOAP service invocation framework uses the event parameter WFBES_INPUT_`<partname>` to support passing values for any header part that may be required to embed applications context into SOAP envelopes. However, unlike other parameters that can be defined while subscribing to the Invoker event, this event parameter can only be defined during the event raise.

- Support Document Style Web Services Only

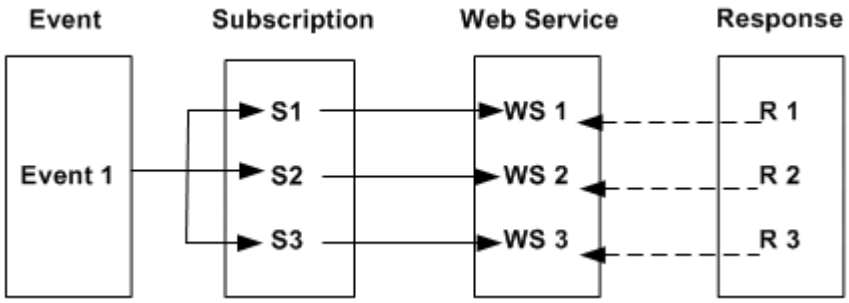
  The SOAP service invocation framework supports invoking only document-based web services. The RPC (remote procedure call) style remote web service invocation is not supported in this release.

- Support One-to-One Relationship of Event Subscriptions

  To successfully invoke SOAP services, each event should only have one subscription (with the 'Invoker Web Service' action type) associated with it. This one-to-one relationship of event subscription is especially important in regards to synchronous request - response service invocation.

  For example, if there are three event subscriptions (S1, S2, and S3) for the same event (Event 1), when a triggering event occurs at runtime, the services associated with each subscription can be invoked three times (WS1, WS2, and WS3) respectively. The scenario is illustrated in the following diagram:

*One-to-One Relationship of Event Subscriptions*




- If callback parameters are not passed, the `getResponseData()` method on the `BusinessEvent` object returns the output (response) message in the same session after the Invoker event raise. The R2 overrides the R1; the R3 overrides the R2. As a result, you will only get the R3 message back.

- If callback parameters are passed, since there are three different instances of the receive event with the same event key, it is difficult to match the response to the corresponding Invoke Web Service subscription.

## Implementing Service Invocation Framework for REST Services

Similar to invoking SOAP services through the service invocation framework, as an integration administrator, you need to perform the following tasks to implement the framework to invoke REST services from Oracle E-Business Suite:

- Supporting REST Service Security and Configuring Security with Customization Level, page 9-27

- Managing REST Service Errors, page 9-30

- Understanding Implementation Consideration for the REST Service Invocation Framework, page 9-30

An integration developer needs to perform the following tasks:

- Defining metadata for the REST service to be invoked

- Calling back to Oracle E-Business Suite with REST service responses (optional)

- Testing REST service invocation

- Troubleshooting REST service invocation failure

- Extending seeded Java rule function for REST services

For more information about these tasks performed by the developer, see Using Service Invocation Framework to Invoke REST Services, *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

## Supporting REST Service Security and Configuring Security with Customization Level

This section describes the REST service security that the REST service invocation framework supports and the customization level that affects the security settings. Specifically, it includes the following topics:

- Understanding REST Service Security, page 9-27
- Configuring REST Service Security with Customization Level, page 9-29

### Understanding REST Service Security

The REST service security this framework supports includes:

- HTTP Basic Authentication, page 9-27
- Digest HTTP Header, page 9-28
- Signature HTTP Header, page 9-28

#### HTTP Basic Authentication

When an HTTP request message is sent to invoke a REST service from Oracle E-Business Suite, user security credentials (user name/password) should be provided as input data in the HTTP header as part of the request message.

Security credentials provided in the Create Event Subscription - Invoke REST Service page during the event subscription creation are stored internally as subscription parameters.

- User name value is stored in the WFBES_REST_USERNAME subscription parameter.
- Password is securely stored in FND_VAULT. Module and Key values are auto-generated and are internally stored in the following subscription parameters:
  - WFBES_REST_PASSWORD_MOD
  - WFBES_REST_PASSWORD_KEY

At runtime, the password is retrieved from FND Vault and Authorization header is included in the HTTP header part of the request to the REST service. Authorization header as per HTTP Basic Authentication scheme is:

```
Authorization Basic base64 encoded value of <usernmae>:<password>
```

For example, `Authorization Basic xxxxxxxxxxxxxxxxxxxxxxx`

#### Digest HTTP Header

In addition to the HTTP Basic Authentication security that is required to authenticate users before invoking the REST service, the REST service invocation framework provides another layer of security allowing you to optionally extend the HTTP header request with Digest and Signature header options.

**Advanced Configuration Page for Adding the Digest Algorithm**

To define the Digest HTTP Header security, you must select a digest algorithm in the HTTP Headers region of the Create Event Subscription - Invoke REST Service Advanced Configuration page, when defining the REST service invoker's event subscription.

The Digest HTTP Header has the following format:

```
Digest: <digest-algorithm>=<digest-value>
```

For example: `Digest: sha-512=xxxxxxxxxxxxxx...`

- `<digest-algorithm>` - It represents the selected digest algorithm value (either `SHA-512` or `SHA-256`).

- `<digest-value>` - It represents the resource (such as payload) after applying the digest algorithm and encoding the result.

#### Signature HTTP Header

The Signature HTTP Header is part of the REST service security this framework supports. It lets you optionally add another layer of authentication before invoking the REST service. In this security model, the sender must authenticate itself with a digital signature produced by a private asymmetric key, such as `rsa-sha256`.

**Advanced Configuration Page for Adding the Signature HTTP Header**

When defining the REST service invoker's event subscription for the signature header security, you must select a value in the Signing Algorithm field of the Create Event Subscription - Invoke REST Service Advanced Configuration page.

Once the signing algorithm is selected in the HTTP Headers region, you must enter additional fields corresponding to the selected signing algorithm. This provides values for the following four components contained in the Signature header:

- **keyid** - This is the string that a server can use to look up the component required to validate the signature.

  This is the value you entered in the KeyID field. If no value is provided in the KeyID field, then its value will be auto-generated. It should be base64-encoded value of SHA-256 fingerprint of the associated digital certificate.

- **algorithm** - This component specifies the digital signature algorithm to use when generating the signature.

  This is the value you selected in the Signing Algorithm field. Its value can be either `rsa-sha512` or `rsa-sha256`.

- **headers** - A string value obtained from a subscription parameter that stores the value entered in the "Headers to be Signed" field.

  Note the value in the "Headers to be Signed" field is a lowercased list of HTTP Header fields, separated by a single space character. Any trailing spaces or multiple spaces between headers will be truncated.

- **signature** - A base64-encoded hash value is formed based on the name and value of each header, and the combination of keystore and certificate alias.

For example, a signature HTTP Header could be like:

```
Signature: keyId="xxxxxxxxxxxxxxxxxxxx... ", algorithm="rsa-
sha512", headers="(request-target) host date digest", signature="
xxxxxxxxxxxxxxxxx/xxxxxxx/xxxxx... "
```

The signature header or RSA-based signature is generated based on the headers to be signed and its value. The signing string is generated as:

```
header1: value1\n
header2: value2\n
..
headerX: valueX
```

Then, the base64-encoded value of the RSA signature of the signing string is added as the *signature* component in the Signature HTTP Header.

#### Configuring REST Service Security with Customization Level

Similar to the behaviors of the customization level described in the SOAP service invocation framework, the REST service invocation framework handles the REST service security for user name, password, keystore file, keystore password, and certificate alias depending on the customization level in the invoker's event subscription:

- Core - No changes can be made to the event and subscription definition. This level is used only for events seeded by Oracle E-Business Suite.

- Limit - The event status can be updated to Enabled or Disabled, but no other changes can be made to the event definition. This level is used only for the events seeded by Oracle E-Business Suite.

- User - Any property in the event and subscription definition can be updated. This level is automatically set for events that you define.

When configuring REST service security, whether you can update or configure these security fields is based on the following conditions:

- User name
  - The Invoke REST Service event subscription's customization level is User.

    You can update the user name.

- The Invoke REST Service event subscription's customization level is Core or Limit.
  - If the subscription owner has provided user name, then the Username field should be read-only and you cannot edit this field.
  - If the user name was not supplied, you can update the user name.

- Password

  You can always update the Password field in the REST Service Security region regardless of the customization level.

- Keystore file, Keystore password, and Certificate Alias

  You can always update these fields in the Create Event Subscription - Invoke REST Service Advanced Configuration page regardless of the customization level.

For information about usage examples of configuring REST service security with different customization levels, see Configuring Security Password with Customization Level, page 9-19.

### Managing REST Service Invocation Errors

The REST service invocation framework leverages the Business Event System to manage errors and exceptions, similar to the error handling mechanism used in the SOAP service invocation framework.

For details on handling errors and exceptions, see Managing SOAP Service Errors, page 9-24.

If there is an error or exception, HTTP Return Status Code should be captured and available to the consuming program.

### Understanding Implementation Consideration for the REST Service Invocation Framework

Oracle E-Business Suite Adapter with Oracle Integration has leveraged the REST service invocation framework for Business Event capabilities in integrations.

For the event subscriptions created automatically from Oracle E-Business Suite Adapter with Oracle Integration, in the Create Event Subscription page you can only update the values of the Status and Phase fields. All other fields in the page are displayed as read-only fields and are not updatable.

For more information about the Oracle E-Business Suite Adapter with Oracle Integration, see *Using the Oracle E-Business Suite Adapter with Oracle Integration*, available in the Oracle Cloud Library on the Oracle Help Center.

# 10

# Monitoring and Managing Outbound Service Invocation Messages Using Service Invocation Monitor

## Service Invocation Monitor Overview

To effectively track and manage outbound or external service invocations through Service Invocation Framework, Service Invocation Monitor provides the monitoring and auditing capabilities for these outbound invocations from Oracle E-Business Suite. Similar to Service Monitor, Service Invocation Monitor fetches data and statistics for each instance of SOAP and REST service request and response messages and provides tracking records through the UI pages where administrators can view and manage the monitored data.

> **Note:** Service Invocation Monitor and Service Monitor are both monitoring and management tools, but each is designed and used for distinct purposes. Service Invocation Monitor tracks all outbound service invocations through Service Invocation Framework whereas Service Monitor tracks all inbound invocations for Oracle E-Business Suite services. For information about Service Monitor, see Monitoring and Managing Inbound Service Invocation Messages Using Service Monitor, page 8-1.

> **Note:** In this release, Service Invocation Monitor does not track and monitor the service invocations from the PL/SQL layer.

Service Invocation Monitor provides the following features:

- **Monitoring and Auditing:** These monitoring and auditing features provide audit trails for administrators to quickly identify any errors if occurred during the

invocation activities.

By default, the monitoring and auditing features are disabled. You can enable or disable these features if desired.

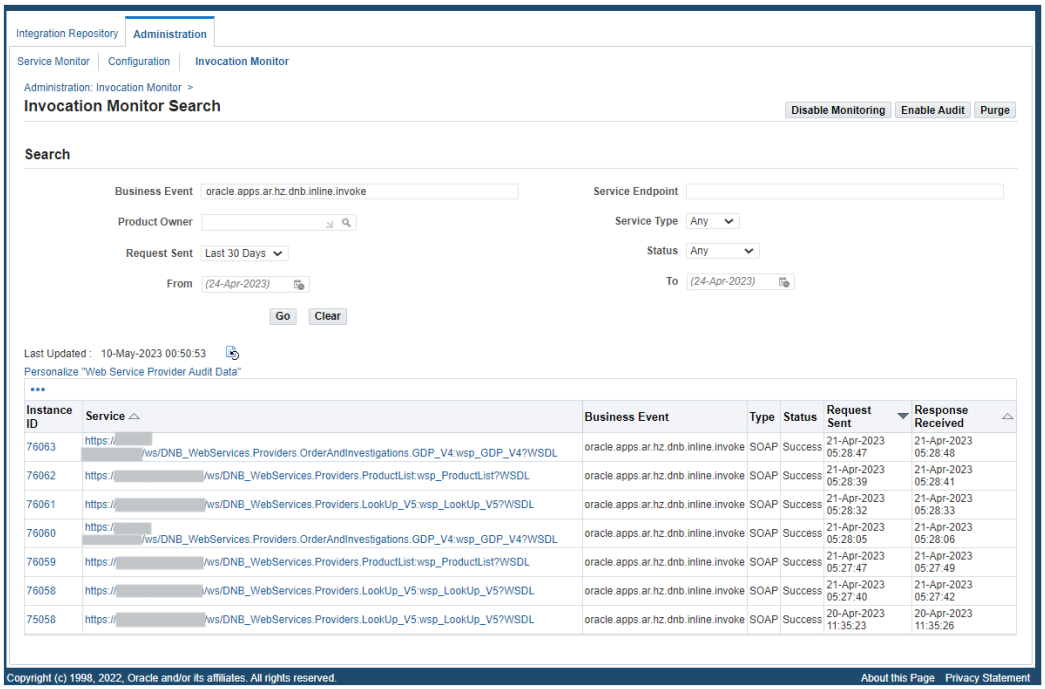
- **Purging:** It purges SOAP and REST messages stored in the Oracle E-Business Suite database, along with the audit records that have been collected through Service Invocation Monitor for a period of time.

### Accessing Service Invocation Monitor

Log in to Oracle E-Business Suite as a user who has the Integration Administrator role.

Select the **Integrated SOA Gateway** responsibility from the navigation menu and then select the **Administration > Invocation Monitor** link. The **Invocation Monitor** subtab is displayed with the Invocation Monitor Search page.

***Invocation Monitor Search Page with Search Results***



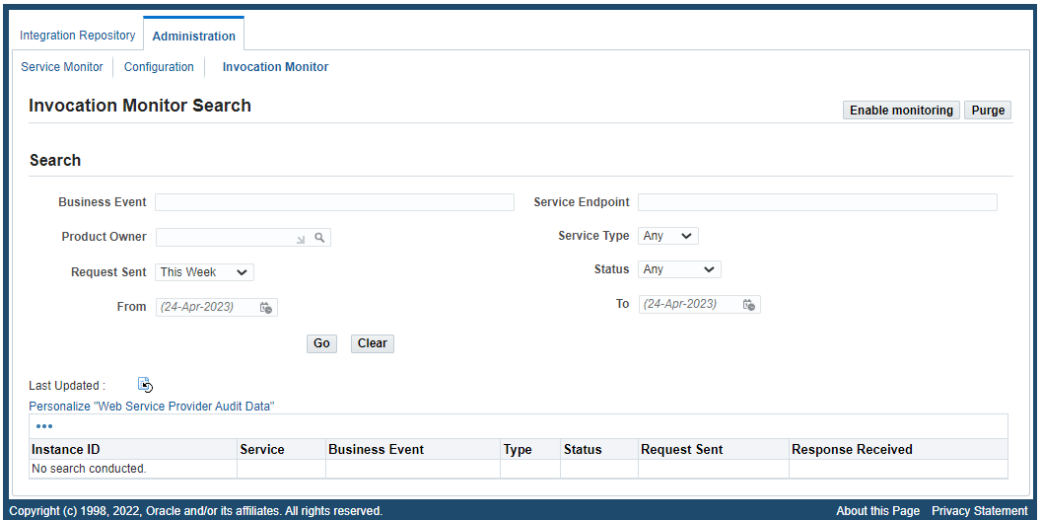

Integration administrators can perform the following activities through Service Invocation Monitor:

- Enabling Monitoring and Auditing for Outbound Invocations, page 10-3
- Searching SOAP and REST Requests, page 10-3
- Viewing SOAP and REST Request and Response Details, page 10-5
- Purging Service Invocation Monitor Data, page 10-10

## Enabling Monitoring and Auditing for Outbound Invocations

Similar to Service Monitor, Service Invocation Monitor provides monitoring and auditing features. If these features are enabled, all invocation request and response messages through Service Invocation Framework can be tracked and saved in Service Invocation Monitor.

***Service Invocation Search Page for Enabling Monitoring***




By default, the monitoring and auditing features are disabled.

To enable or disable the monitoring feature, click **Enable Monitoring** or **Disable Monitoring** in the Invocation Monitor Search page.

Clicking **Enable Monitoring** only enables the monitoring feature, the auditing feature is not enabled by default. You need to click **Enable Auditing** to enable the auditing feature. Only when the auditing feature is enabled, the payload messages can be captured. Click **Disable Auditing** to disable the auditing feature if it is enabled.

> **Note:** Clicking **Disable Monitoring** also disables the auditing feature. To enable the auditing feature, you must first enable the monitoring feature by clicking **Enable Monitoring**, and then **Enable Auditing**.

## Searching SOAP and REST Requests

In the Invocation Monitor Search page, perform a search to locate your desired instances of outbound service invocations through Service Invocation Framework.

In the Search region, enter your desired search criteria, such as business event name, service endpoint, product owner, service type, request sent time, request sent time

period, and request status.

- **Business Event**: Enter an event name that is used to invoke an external service. For example, enter `oracle.apps.po%` to locate the event names containing `oracle.apps.po` as the search result.

- **Service Endpoint**: Enter a desired service endpoint name. For example, enter `%getPO%` to locate all service endpoint containing `getPO`.

- **Product Owner**: Use the list of values to select a product that owns the business event subscription.

- **Service Type**: Select a desired service type including "Any", "REST", and "SOAP". By default, "Any" is selected.

- **Request Sent**: Select a desired value from the list of values. It includes "Any Time", "Today", "This Week", "Last 2 Weeks", "Last 30 Days", "Last 60 Days", and "Last 90 Days". By default, "Today" is selected.

  > **Note:** All the selection values listed here include the requests sent day of Today except "Any Time". For example, "This Week" means the last 7 days inclusive of today the requests have been sent, and "Last 30 Days" means the last 30 days inclusive of today the requests have been sent. "Any Time" means a blind search of requests sent regardless of the Request Sent date.

- **Status**: Select a desired value from the list of values for your search. It contains "Any", "In Process", "Processed Successfully, and "Processed with Error". By default, "Any" is selected.

- **From**: Use the Date Time Picker to select an appropriate request sent start date.

- **To**: Use the Date Time Picker to select an appropriate request sent end date for your search.

Click **Go** to process your search. All entries that match your search criteria will be retrieved and displayed in a table. Each entry in the result table includes the instance ID, service name, business event name, service type, date and time the request was sent and the response was received.

From the search result page, you can perform the following tasks:

- View the request and response details in the Invocation Instance page by clicking the **Instance ID** link for a given request

  See: Viewing SOAP and REST Service Invocation Instance Details, page 10-5.

- View the status of each monitored request and response

- View the business event for a given request
- Purge outbound Service Invocation Monitor data collected over a period of time by clicking **Purge**

  See: Purging Service Invocation Monitor Data, page 10-10.

For information on how to enable the monitoring and auditing features in Service Invocation Monitor, see Enabling Monitoring and Auditing for Outbound Invocations, page 10-3.

**To perform a search:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility. From the navigation menu, select the **Invocation Monitor** link from the **Administration** section to open the Invocation Monitor Search page.
2. In the Search region, enter appropriate search criteria including business event name, service endpoint, product owner, request sent time, service type, request status, and requests sent between the From and To date-time values for your search. Click **Go** to process your search.
3. All requests that match your search criteria appear.
4. Click the **Instance ID** link for a given ID to view the request and response details in the Service Instance page.
5. Click the **Business Event** link to view the selected business event details.
6. Click **Purge** to purge monitored data collected for a period of time.

## Viewing SOAP and REST Service Invocation Instance Details

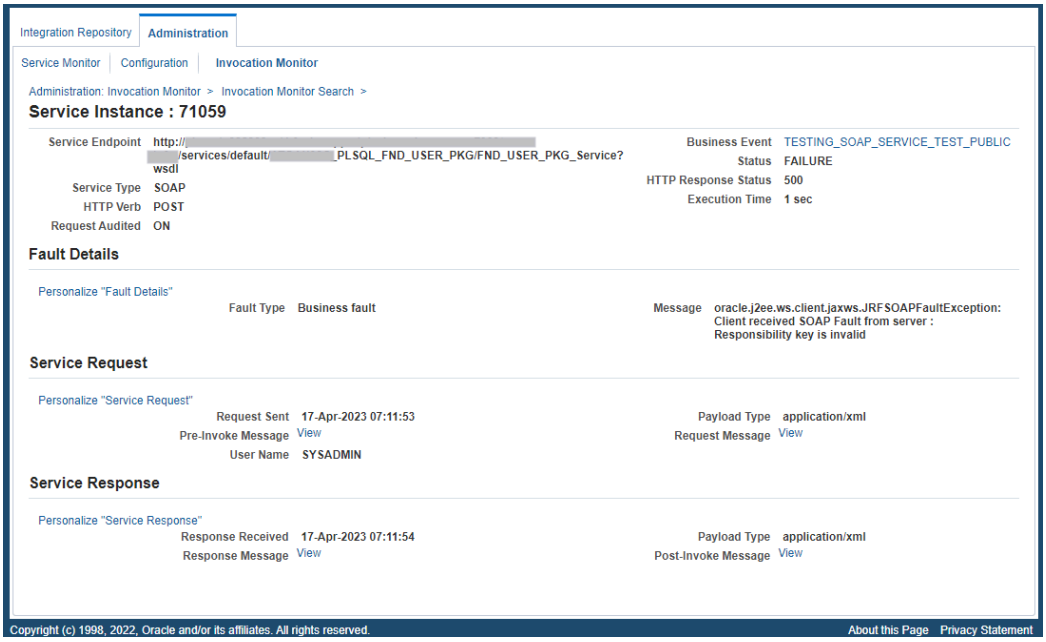
From the search result table in the Invocation Monitor Search page, you can view all SOAP and REST invocation messages that match your search criteria. To view the request and response details for a given instance, click the **Instance ID** link of a desired instance. The Invocation Instance Details page appears for your selected instance with "SOAP" or "REST" service type:

- Viewing SOAP Service Invocation Details, page 10-5
- Viewing REST Service Invocation Details, page 10-7

### Viewing SOAP Service Invocation Details

When you click an instance ID link from the search result table and that instance is for a service type of "SOAP", then the SOAP Invocation Details page appears.

***SOAP Invocation Details Page***




This page displays the invocation details, along with the Service Request region and Service Response region.

The invocation information includes the complete Service Endpoint information along with the Business Event link, Service Type "SOAP", Status, HTTP Verb, HTTP Response Code, Request Audit (On or Off to indicate whether the auditing feature is enabled or not when the request is sent), and the Execution Time between the request is sent and response is received.

- Business Event: Click the **Business Event** link to view the corresponding Business Event Details page.
- HTTP Verb: "POST" is displayed.

**Fault Details Region (Conditional)**

This region appears only if the request has a "Failure" status.

*Fault Type*: Depending upon the nature of fault, the value could be one of the following types:

- Client Fault - Service exceptions returning 4xx HTTP status code fall under this category.
- Business Fault - Service exceptions returning 5xx HTTP status code fall under this category.

*Fault Message*: Fault code or the first 150 characters of the error message are shown.

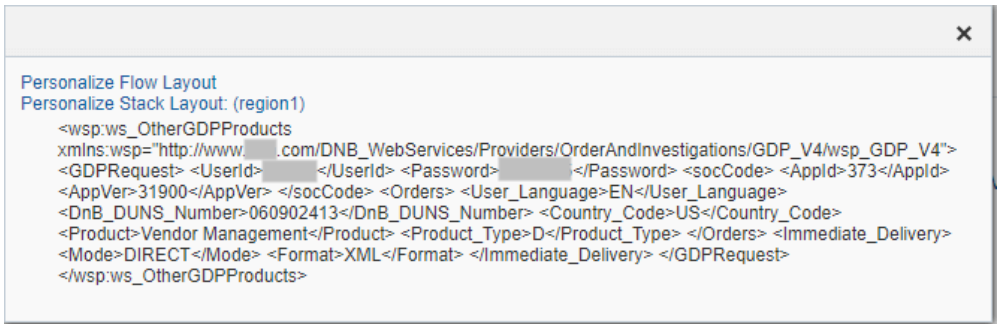
### Service Request Region

This region contains the SOAP invocation request information including request sent date and time, payload type (XML), pre-invoke message, SOAP request message, and username.

- Click the Pre-Invoke Message **View** link if it is available to view the event data payload in XML format. This is the payload data before the request message is constructed or before any processing in the `PreInvoke` method.

- Click the Request Message **View** link if it is available to view the content in HTTP body sent as the request. Note that this **View** link is not displayed if the SOAP request is for the GET method.

***A Sample SOAP Request Message***



Personalize Flow Layout
Personalize Stack Layout: (region1)


```
<wsp:ws_OtherGDPProducts
xmlns:wsp="http://www.[redacted].com/DNB_WebServices/Providers/OrderAndInvestigations/GDP_V4/wsp_GDP_V4">
<GDPRequest> <UserId>[redacted]</UserId> <Password>[redacted]</Password> <socCode> <AppId>373</AppId>
<AppVer>31900</AppVer> </socCode> <Orders> <User_Language>EN</User_Language>
<DnB_DUNS_Number>060902413</DnB_DUNS_Number> <Country_Code>US</Country_Code>
<Product>Vendor Management</Product> <Product_Type>D</Product_Type> </Orders> <Immediate_Delivery>
<Mode>DIRECT</Mode> <Format>XML</Format> </Immediate_Delivery> </GDPRequest>
</wsp:ws_OtherGDPProducts>
```


### Service Response Region

This region contains the SOAP invocation response details, such as response received date and time, payload type (XML), post-invoke message, and SOAP response message.

- Click the Response Message **View** link if it is available to view the content in HTTP body received as the response.

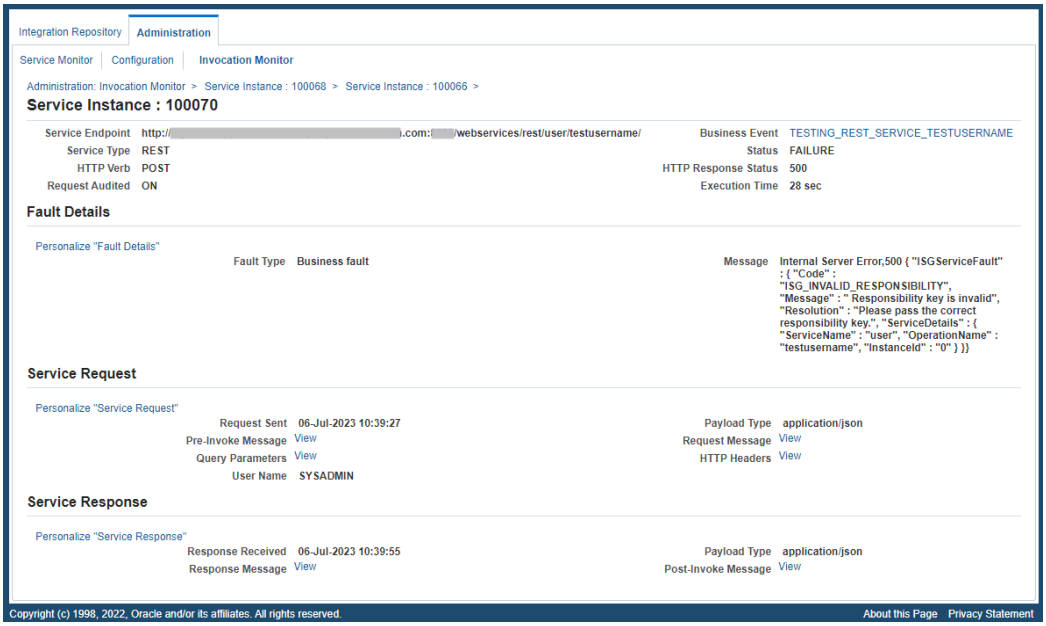
- Click the Post-Invoke Message **View** link to view the event data payload in XML format. This is the payload data after post processing on the response message received or after any processing in the `PostInvoke` method.

## Viewing REST Service Invocation Details

When you click an instance ID link from the search result table and that instance is for a service type of "REST", then the REST Invocation Details page appears.

This page contains invocation details for the selected instance, Service Request region, and Service Response region.

***REST Invocation Details Page***




The invocation information shown on the top of the page includes the complete Service Endpoint, Business Event link, Service Type "REST", Status, HTTP Verb, HTTP Response Code, Request Audit (On or Off to indicate whether the auditing feature is enabled or not when the request is sent), and the Execution Time between the request is sent and response is received.

- Business Event: Click the **Business Event** link to view the corresponding Business Event Details page.

- HTTP Verb: "POST" or "GET" is displayed.

### Fault Details Region (Conditional)

Similar to the Fault Details region for the SOAP Invocation Details page, this region appears only if the REST request has a "Failure" status.

You can find the fault details through the Fault Type and Message fields, as described in the Viewing SOAP Service Invocation Details, page 10-5.

### Service Request Region

This region contains the REST invocation request information including request sent date and time, payload type (XML or JSON), pre-invoke message, request message, query parameter, HTTP Headers, and username.

- Click the Pre-Invoke Message **View** link if available to view the event data payload. This is the payload data before the request message is constructed or before any processing in the `PreInvoke` method.

- Click the Request Message **View** link if it is available to view the content in HTTP body sent as the request. Note that this **View** link is not displayed if the REST request is for the GET method.

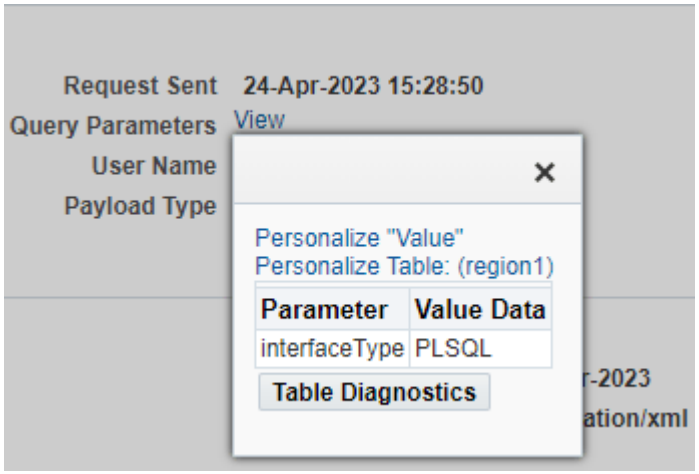
- Click the Query Parameters **View** link to display a list of query parameters and their corresponding values in a table. For example, the following table includes one query parameter called "Interface Type" and the associated value "PLSQL":

  ***Query Parameters***

  | Parameter | Value |
  |---|---|
  | Interface Type | PLSQL |

  Click the **View** link to display the query parameters in a dialog.

  ***REST Request Message Query Parameters***

- Click the HTTP Headers **View** link to view the HTTP Headers sent in the request. For example, it can be like:

  ```
  GET /home.html HTTP/1.1
   Host: www.example.com
   Accept: application/json
   Referrer: https://www.example.com
   Connection: keep-alive
  ```

### Service Response Region

This region contains the REST invocation response details, such as response received date and time, payload type (XML or JSON), post-invoke message, and response message.

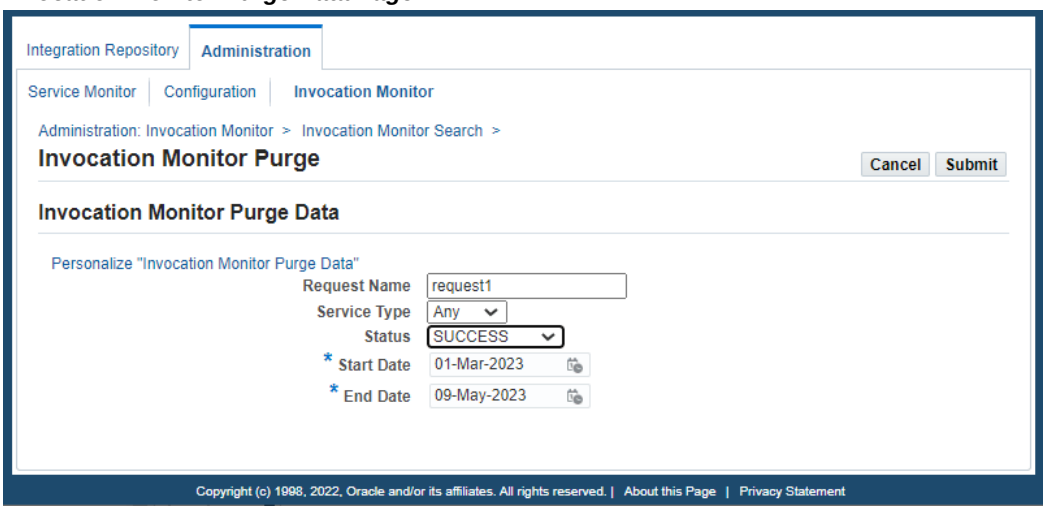
- Click the Response Message **View** link if it is available to view the content in HTTP body received as the response.

- Click the Post-Invoke Message **View** link to view the event data payload after post processing on the response message received or after any processing in the `PostInvoke` method.

## Purging Service Invocation Monitor Data

All SOAP and REST invocation messages stored in the Oracle E-Business Suite database, and audit records that have been collected through Service Invocation Monitor for a period of time can be purged. Click **Purge** in the Invocation Monitor Search page to launch the Invocation Monitor Purge Data page.

***Invocation Monitor Purge Data Page***




Enter the following purge parameters in the Invocation Monitor Purge Data page:

- **Request Name**: Specify the Request Name for your request.

- **Service Type**: Select a desired service type from the drop-down list. It can be 'SOAP', 'REST', or 'Any'. By default, 'Any' is selected, meaning that both SOAP and REST service types are included.

- **Status**: Specify a desired status from the drop-down menu for your request.

- **Start Date**: Identify the start date of the date range for your purge.

- **End Date**: Identify the end date of the date range for your purge.

Click **Submit**. A request number will be automatically assigned to you for your purge request indicating that your request has been submitted for processing. Service

Invocation Monitor will purge all monitored invocation data that match your purge criteria within your specified date range.

**To purge service invocation requests and responses:**

1. Log in to Oracle E-Business Suite as a user who has the Integration Administrator role. Select the Integrated SOA Gateway responsibility.

   From the navigation menu, select the **Invocation Monitor** link from the Administration section to open the Invocation Monitor Search page.

2. Click **Purge**.

3. Enter the following information in the Invocation Monitor Purge Data page:

   1. Enter the request name for your purge request.

   2. Select a desired Service Type value (SOAP, REST, or Any) for your purge.

   3. Select a desired Status value for your purge request.

   4. Enter the Start Date and End Date fields to specify the time range for your purge.

4. Click **Submit** to submit your purge request.

# A

# Oracle E-Business Suite Integrated SOA Gateway Diagnostic Tests

## Overview

The installation of Oracle E-Business Suite Integrated SOA Gateway requires number of manual setup tasks on both Oracle E-Business Suite and Oracle SOA Suite. To effectively identify any issues, Oracle E-Business Suite Integrated SOA Gateway provides a suite of diagnostic tests that are run through a backend script to help determine specific causes or issues with installation steps.

This diagnostic test suite includes multiple tests with various test functions to check on both Oracle E-Business Suite and Oracle SOA Suite instances. For example, certain tests validate if correct versions of required software and libraries are installed, some tests check if needed patches are applied, or if issues are functional related, such as Generate, Deploy, or other design-time activities.

At the end of test run, a report will be generated which may contain corrective actions mostly regarding the installation of Oracle E-Business Suite Integrated SOA Gateway.

To have better understanding on the diagnostic tests and how the test suite is run, the following topics are included in this chapter:



### Understanding the Usage of Backend Script $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml

Oracle E-Business Suite Integrated SOA Gateway uses an Ant script `$JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml` to run the diagnostic tests through backend processing.

**Note:** `$JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml` is a multipurpose script. It can also be used to implement the design-time activities such as generate, regenerate, deploy, undeploy, activate, retire, and reset services as well as to upgrade or postclone services from command line.

For more information on how to use the script to perform the design-time activities, see Managing SOAP Service Lifecycle Activities Using An Ant Script, page 3-56 and Managing REST Service Lifecycle Activities Using An Ant Script, page 3-64.

## Running Diagnostic Tests

Use the backend script `isgDesigner.xml` to run complete diagnostic tests on both Oracle E-Business Suite and Oracle SOA Suite with the following syntax:

```
ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml
DiagnoseISGSetup
```

Alternatively, use the same script supplying with different targets to run the configuration checks for various purposes. For example, use the test to check only on the Oracle E-Business Suite side or the Oracle SOA Suite side, or to test the design-time operations of Oracle E-Business Suite Integrated SOA Gateway for all types of interfaces.

Use the following commands to run diagnostic tests depending on your needs:

- `ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml DiagnoseAGENTSetup`

  This command runs configuration checks on the Oracle E-Business Suite side.

- `ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml DiagnoseAPPSetup`

  This command runs configuration checks on the Oracle SOA Suite side.

- `ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml DiagnoseISGSetup`

  This command runs complete diagnostic tests on both Oracle E-Business Suite and Oracle SOA Suite.

- `ant -f $JAVA_TOP/oracle/apps/fnd/isg/ant/isgDesigner.xml DiagnoseISGFunctionality`

  This command runs all design-time operations for all types of interfaces in Oracle E-Business Suite Integrated SOA Gateway.

After each test run, a report DiagnosticsReport.xml will be generated as a result. The generated report will have test name, status, and message if the test is failed. Message will convey the information that what type of error occurred and what is the error and

corresponding actions if available.

# B

# Synchronous and Asynchronous Web Services

## Synchronous and Asynchronous Web Services

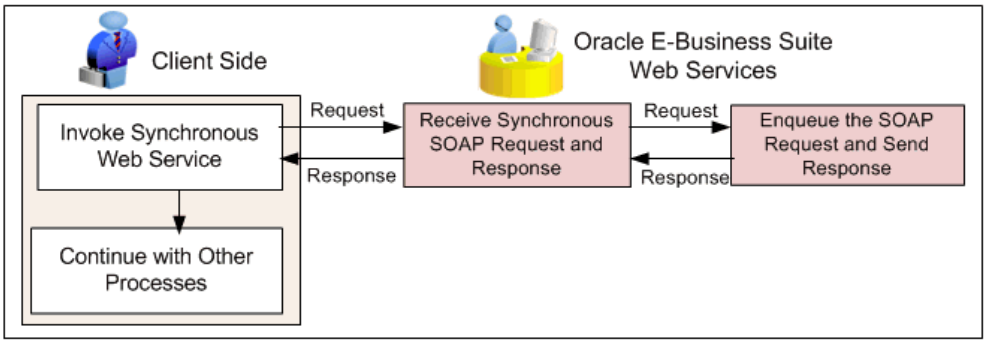
Oracle E-Business Suite Integrated SOA Gateway supports both synchronous and asynchronous service processing and invocation for SOAP-based services.

Interfaces exposed as REST services can be generated with the support for synchronous interaction pattern only. Asynchronous pattern for REST services is currently not supported in this release.

- **Synchronous Web Services**

  This type of service invocation provides an immediate response to a query. In this situation, the client will wait until the server sends back the response message. The advantage of using the synchronous service is that the client application knows the status of the web service operation in a very short time.

**Synchronous Web Service Transaction**




When a web service client sends a synchronous SOAP request to an Oracle E-Business Suite service, the SOAP response will be sent back to the client as soon as the process completes.

- **Asynchronous Web Services (SOAP Web Services Only)**

  This type of service invocation may require a significant amount of time to process a request. However, the client that invokes the Oracle E-Business Suite service can continue with other processing in the meantime rather than wait for the response.

  Asynchronous operation is extremely useful for environments in which a service, such as a loan processor, can take a long time to process a client request.

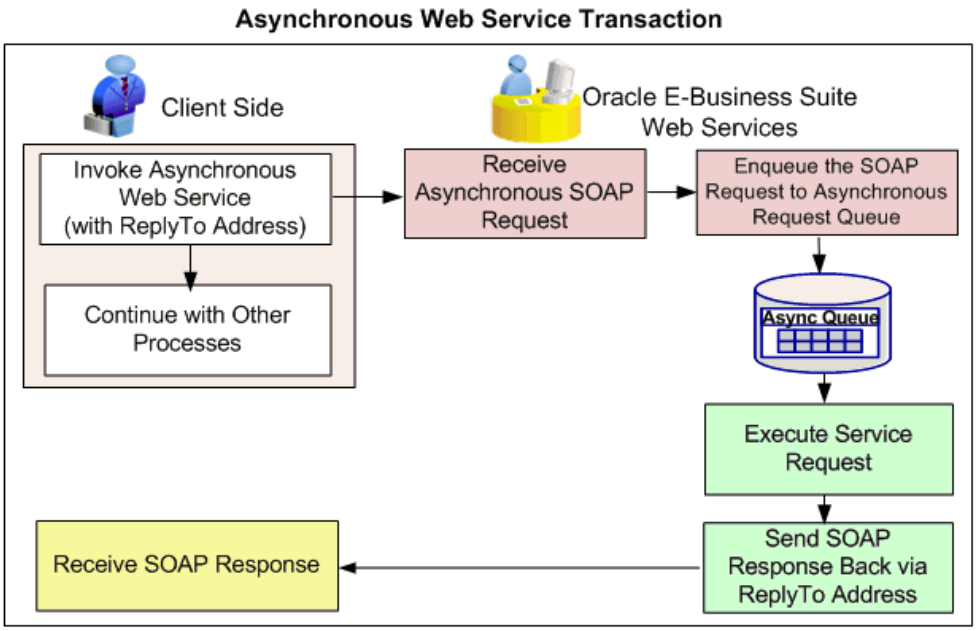
  In this release, asynchronous operation pattern is supported for SOAP-based services only.

  ***Callback without Acknowledgement***

  Callback pattern is a very important communication method in asynchronous services - a request is made to the service provider and a response is sent back to the requester when it is ready. This pattern can be used in conjunction with acknowledgement to recognize the receipt of a request sent by a requester. Only *callback without acknowledgement* pattern is supported in this release.

  In callback without acknowledgment pattern, a SOAP Callback header becomes necessary when the web service is asynchronous and the client contact information is unknown at the deployment time. Callback header allows the client to specify how to contact the client (`ReplyTo` address) in the request for service. Therefore, the client must publish a listener or a receive service. In other words, the structure of the WSDL dictates how the client will receive the response.

  A web service client must provide `MessageID`, and an appropriate callback endpoint address (`ReplyTo` and `FaultTo`) using WS-Addressing in the SOA Headers for the asynchronous request callback pattern.




When a web service client sends a SOAP request to an Oracle E-Business Suite service, on completion of the service invocation, the SOAP response (service response payload) is sent to the ReplyTo address of the client. This pattern does not expect acknowledgment from the client as it is a fire-and-forget message exchange pattern for callback.

SOAP services, depending on specified interaction patterns, can be generated synchronously, asynchronously, or both synchronously and asynchronously to meet your business needs. REST services can be generated with synchronous operation only.

Once a SOAP service has been generated and deployed to an Oracle SOA Suite WebLogic managed server, service consumers or web service clients can send request messages through Oracle SOA Suite. After security checks on the inbound requests, Oracle E-Business Suite web services can be invoked synchronously or asynchronously.

For information on how to specify interaction patterns for a given interface, see Generating SOAP Web Services, page 3-4.

# C

# Error Messages

## Error Messages and Solutions

The following table describes error message if occurs during the design-time activities through the Integration Repository user interface, as well as during the service runtime invocation from a service provider.

The error codes and corresponding solutions are also listed in the table for possible solutions.

| Error Code | Error Message | Resolution | HTTP Status Code |
|---|---|---|---|
| FND_SOA_AUDIT_REQUEST_ERROR | Error in audit request | N/A | N/A |
| FND_SOA_AUTHENTICATION_FAILURE | Invalid user name or password | Please use proper username password combination. | 401 |
| ISG_SERVICE_AUTH_FAILURE | User not authorized to execute service | Please grant the function to the user and responsibility combination. | 403 |
| ISG_DB_CONNECTION_ERROR | Error in creating database connection | N/A | 500 |

| Error Code | Error Message | Resolution | HTTP Status Code |
|---|---|---|---|
| ISG_DUPLICATE_RESPONSIBILITY | Multiple occurrences of the Responsibility element are found in the request. | RESTHeader can have only one occurrence of the Responsibility element. Please send a request with only one Responsibility element in RESTHeader. | 500 |
| ISG_DUPLICATE_RESPAPPLICATION | Multiple occurrences of the Responsibility Application element are found in the request. | RESTHeader can have only one occurrence of the Responsibility Application element. Please send a request with only one Responsibility Application element in RESTHeader. | 500 |
| ISG_DUPLICATE_NLSLANGUAGE | Multiple occurrences of the NLS language element are found in the request. | RESTHeader can have only one occurrence of the NLS language element. Please send a request with only one NLS language element in RESTHeader. | 500 |
| ISG_DUPLICATE_LANGUAGE | Multiple occurrences of the Language element are found in the request. | RESTHeader can have only one occurrence of the Language element. Please send a request with only one Language element in RESTHeader. | 500 |
| ISG_DUPLICATE_ORG_ID | Multiple occurrences of the Org Id element are found in the request. | RESTHeader can have only one occurrence of the Org Id element. Please send a request with only one Org Id element in RESTHeader. | 500 |

| Error Code | Error Message | Resolution | HTTP Status Code |
|---|---|---|---|
| ISG_DUPLICATE_SEC_GROUP_KEY | Multiple occurrences of the Security Group element are found in the request. | RESTHeader can have only one occurrence of the Security Group element. Please send a request with only one Security Group element in RESTHeader. | 500 |
| FND_SOA_DUPLICATE_MSGID | Server has already received this request | N/A | N/A |
| ISG_FILE_ACCESS_ERROR | Error in accessing server file system | N/A | N/A |
| HTTP_COMMUNICATION_ERROR | Error in HTTP communication | N/A | N/A |
| ISG_INVALID_STORAGE_LOC | System property TEMP_DIRECTORY_LOCATION is not set correctly | Please set proper value for property TEMP_DIRECTORY_LOCATION | 500 |
| ISG_SOAP_INVALID_BODY | Body did not match schema | N/A | N/A |
| ISG_SOAP_INVALID_ENVELOPE | SOAP Envelope/Request could not be parsed | N/A | N/A |
| ISG_SOAP_INVALID_HEADER | Invalid or missing header in request | N/A | N/A |
| ISG_INVALID_IREP_CLASS_ID | Irep class Id does not exist in database | N/A | N/A |

| Error Code | Error Message | Resolution | HTTP Status Code |
|---|---|---|---|
| ISG_INVALID_LANGUAGE_CODE | Language code is not valid | N/A | 403 |
| ISG_INVALID_ORG_ID | Org Id is not valid | N/A | 403 |
| ISG_INVALID_RESPONSIBILITY | Responsibility key is not valid | N/A | 403 |
| ISG_INVALID_RESP_SHORT_CODE | Responsibility application short code id not valid | N/A | 403 |
| ISG_INVALID_SEC_GROUP | Security group key is not valid | N/A | 403 |
| FND_SOA_INVALID_USERNAME | Username is not valid | N/A | 401 |
| ISG_IREP_ACCESS_ERROR | Error in accessing Integration Repository | N/A | 400 |
| ISG_DATA_TYPE_CONVERSION_ERROR | Error encountered while converting data into different format | Error encountered while converting data into different format. Contact administrator for further diagnosis. | 400 |
| ISG_INVALID_FUNCTION | The requested operation could not be found | The operation specified in the request URL may not be mapped to any valid function for this service. Please check the operation for this service. | 501 |

| Error Code | Error Message | Resolution | HTTP Status Code |
|---|---|---|---|
| ISG_INVALID_URL_PATTERN | The requested URL pattern is invalid | The requested URL pattern or parameters cannot be mapped to a specific resource. Please check the validity of URL pattern or URL query parameters. | 400 |
| ISG_LANGUAGE_NOT_INSTALLED | Language is not installed | Language is not installed. | 500 |
| ISG_SERVICE_ACTIVATE_ERROR | Error occurred while activating the web service | Check the server logs for details. | 500 |
| ISG_SERVICE_DEPLOY_ERROR | Error occurred while deploying the web service | Check the server logs for details. | 500 |
| ISG_SERVICE_EXECUTION_ERROR | Error occurred while executing the web service request | Check the server logs for details. | 500 |
| ISG_SERVICE_GENERATE_ERROR | Error occurred while generating the web service | Check the server logs for details. | 500 |
| ISG_SERVICE_NOT_DEPLOYED | Web service is not deployed | Please deploy the web service. | 500 |
| ISG_SERVICE_PROCESSING_ERROR | Error occurred while processing the web service request | Check the server logs for details. | 500 |
| ISG_SERVICE_RESET_ERROR | Error occurred while resetting the web service | Check the server logs for details. | 500 |

| Error Code | Error Message | Resolution | HTTP Status Code |
|---|---|---|---|
| ISG_SERVICE_RETIRE_ERROR | Error occurred while retiring the web service | Check the server logs for details. | 500 |
| ISG_SERVICE_UNDEPLOY_ERROR | Error occurred while undeploying the web service | Check the server logs for details. | 500 |
| ISG_STREAM_CONVERSION_ERROR | Error encountered while converting data to stream | Error encountered while converting data to stream. Contact administrator for further diagnosis. | 400 |
| ISG_SYSTEM_ERROR | System error while processing the request | Check the server logs for details. | 500 |
| ISG_UNKNOWN_SERVICE | Requested web service or web resource does not exist | Requested web service or web resource does not exist or it is undeployed. Please deploy it. | 404 |
| ISG_UNSUPPORTED_IFACE_TYPE | Interface type is either invalid or it is not currently supported | Please check the interface type. | 404 |
| ISG_UNSUPPORTED_MEDIA_TYPE | The content type is not supported | Please use a supported content type. Refer to product documentation for more information on supported content type. | 415 |
| ISG_UNSUPPORTED_REST_VERB | System does not support the HTTP verb | Follow product documentation to ensure the HTTP verbs supported. The service description also lists the supported verbs. | 400 |

| Error Code | Error Message | Resolution | HTTP Status Code |
|---|---|---|---|
| ISG_USER_RESP_MISMATCH | Responsibility is not assigned to user | Assign the Responsibility to user. | N/A |
| ISG_WSDL_ACCESS_ERROR | Error occurred while accessing the WSDL file | Check the server logs for details. | N/A |
| ISG_WSDL_PARSE_ERROR | Error occurred while parsing the WSDL file | Check the server logs for details. | N/A |
| FND_SOA_SERVICE_IN_PROGRESS | One more generate is already in progress | N/A | 500 |
| SQL_EXEC_ERROR | Error while executing SQL | N/A | N/A |
| FND_XML_PARSE_ERROR | Error in XML parsing | N/A | N/A |
| ISG_MULTIPLE_RESP_RECS_FOUND | Multiple records are found associated with the responsibility key or responsibility name | Pass an appropriate responsibility application short name also. | N/A |
| ISG_INVALID_USER_SECGRP | The user is not associated with the security group | Please pass a security group associated with the user. | N/A |
| ISG_REQUEST_PARSE_ERROR | The request could not be parsed correctly | This may be due to malformed construction of the payload or incorrect Content-Type header. Please check the wellformedness of payload, matching Content-Type header of the HTTP request and retry. | 400 |

**Note: Additional Information for PL/SQL APIs**

Most of the Oracle seeded PL/SQL APIs use a standard OUT parameter X_RETURN_STATUS to indicate the return status of the APIs. When such an API is executed as a REST service, the HTTP Response Code is determined based on the runtime value of the API execution status in the output parameter X_RETURN_STATUS. The X_RETURN_STATUS parameter can have the following values:

- FND_API.G_RET_STS_SUCCESS - The HTTP Response Status Code is set to 200.
- FND_API.G_RET_STS_ERROR - The HTTP Response Status Code is set to 422.
- FND_API.G_RET_STS_UNEXP_ERROR - The HTTP Response Status Code is set to 422.
- Any other value - The HTTP Response Status Code is set to 200.

**Additional Information for Application Module Services**

Under exceptional or error cases, an Application Module Service may return ISG_SERVICE_CUSTOM_ERROR followed by ERROR_SUB_CODE. It may have business function specific error message and resolution. For such cases, HTTP 500 status code would be returned.

Additionally, for an Application Module Service, if the interface does not return business function specific resolution and if the FND: Diagnostics profile option is enabled, then an error stack trace would be returned as the resolution.

# Glossary

### Agent

A named point of communication within a system.

### Agent Listener

A type of service component that processes event messages on inbound agents.

### Asynchronous Operation

Unlike the synchronous service processing to obtain the result immediately, an asynchronous operation may require a significant amount of time to process a request.

However, the client that invokes the Oracle E-Business Suite service can continue with other processing in the meantime rather than wait for the response.

### Business Event

See Event.

### Callback Pattern

Callback pattern is an important communication method in asynchronous services. An asynchronous callback means that a request is made to the service provider and a response (callback) is sent back to the requester when it is ready. This pattern can be used in conjunction with acknowledgement to recognize the receipt of a request sent by a requester.

### Concurrent Manager

An Oracle E-Business Suite component that manages the queuing of requests and the operation of concurrent programs.

### Event

An occurrence in an internet or intranet application or program that might be significant to other objects in a system or to external agents.

### Event Activity

A business event modelled as an activity so that it can be included in a workflow

process.

### Event Data

A set of additional details describing an event. The event data can be structured as an XML document. Together, the event name, event key, and event data fully communicate what occurred in the event.

### Event Key

A string that uniquely identifies an instance of an event. Together, the event name, event key, and event data fully communicate what occurred in the event.

### Event Message

A standard Workflow structure for communicating business events, defined by the datatype `WF_EVENT_T`. The event message contains the event data as well as several header properties, including the event name, event key, addressing attributes, and error information.

### Event Subscription

A registration indicating that a particular event is significant to a system and specifying the processing to perform when the triggering event occurs. Subscription processing can include calling custom code, sending the event message to a workflow process, or sending the event message to an agent.

### Function

A PL/SQL stored procedure that can define business rules, perform automated tasks within an application, or retrieve application information. The stored procedure accepts standard arguments and returns a completion result.

### Integration Repository

Oracle Integration Repository is the key component or user interface for Oracle E-Business Suite Integrated SOA Gateway. This centralized repository stores native packaged integration interface definitions and composite services.

### Integration Repository Parser

It is a standalone design-time tool used by the integration administrator to validate annotated custom interface definitions against the annotation standards and generate an Integration Repository loader file (iLDT). This generated iLDT file can be uploaded to Integration Repository where custom interfaces can be exposed to all users.

### Interface Type

Integration interfaces are grouped into different interface types.

### JSON

JSON (JavaScript Object Notation) is a text-based open standard designed for human-readable data interchange. The JSON format is often used with REST services to transmit structured data between a server and web application, serving as an alternative to XML.

### Loose Coupling

Loose coupling describes a resilient relationship between two or more systems or organizations with some kind of exchange relationship. Each end of the transaction makes its requirements explicit and makes few assumptions about the other end.

### Lookup Code

An internal name of a value defined in a lookup type.

### Lookup Type

A predefined list of values. Each value in a lookup type has an internal and a display name.

### Message

The information that is sent by a notification activity. A message must be defined before it can be associated with a notification activity. A message contains a subject, a priority, a body, and possibly one or more message attributes.

### Message Attribute

A variable that you define for a particular message to either provide information or prompt for a response when the message is sent in a notification. You can use a predefine item type attribute as a message attribute. Defined as a 'Send' source, a message attribute gets replaced with a runtime value when the message is sent. Defined as a 'Respond' source, a message attribute prompts a user for a response when the message is sent.

### Notification

An instance of a message delivered to a user.

### Notification Worklist

A web page that you can access to query and respond to workflow notifications.

### Operation

An abstract description of an action supported by a service.

### Port

A port defines an individual endpoint by specifying a single address for a binding.

### Port Type

A port type is a named set of abstract operations and abstract messages involved.

### Process

A set of activities that need to be performed to accomplish a business goal.

### REST

Representational State Transfer (REST) is an architecture principle in which the web services are viewed as resources and can be uniquely identified by their URLs. The key characteristic of a REST service is the explicit use of HTTP methods (GET, POST, PUT, and DELETE) to denote the invocation of different operations.

### SAML Token (Sender-Vouches)

This type of security model authenticates web services relying on sending a username only through Security Assertion Markup Language (SAML) assertion.

SAML is an XML-based standard for exchanging authentication and authorization data between security domains, that is, between an identity provider and a service provider. SAML Token uses a sender-vouches method to establish the correspondence between a SOAP message and the SAML assertions added to the SOAP message.

See Username Token.

### Service

A service is a collection of related endpoints.

### Service Component

An instance of a Java program which has been defined according to the Generic Service Component Framework standards so that it can be managed through this framework.

### Service Monitor

It is the monitoring and auditing tool in Oracle E-Business Suite allowing administrators to monitor inbound SOAP and REST service invocation messages.

It is known as SOA Monitor in earlier releases.

### Service Invocation Monitor

Service Invocation Monitor is a monitoring and auditing tool allowing administrators to monitor outbound SOAP and REST service invocations from Oracle E-Business Suite through Service Invocation Framework.

### SOA

Service-oriented Architecture (SOA) is an architecture to achieve loose coupling among interacting software components and enable seamless and standards-based integration

in a heterogeneous IT ecosystem.

### SOAP

Simple Object Access Protocol (SOAP) is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses XML technologies to define an extensible messaging framework providing a message construct that can be exchanged over a variety of underlying protocols.

### Subscription

See Event Subscription.

### Synchronous Operation

Synchronous operation provides an immediate response to a query. In this situation, the client connection remains open from the time the request is submitted to the server. The client will wait until the server sends back the response message.

### Username Token

A type of security model based on username and password to authenticate SOAP requests at runtime.

See SAML Token (Sender-Vouches).

### WADL

Web Application Description Language (WADL) is designed to provide a machine-processable description of HTTP-based web applications. It models the resources provided by a service and the relationships between them.

### Web Services

A web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in WSDL. Other systems interact with the web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards.

### Workflow Engine

The Oracle Workflow component that implements a workflow process definition. The Workflow Engine manages the state of all activities for an item, automatically processes functions and sends notifications, maintains a history of completed activities, and detects error conditions and starts error processes. The Workflow Engine is implemented in server PL/SQL and activated when a call to an engine API is made.

### WSDL

Web Services Description Language (WSDL) is an XML format for describing network services as a set of endpoints operating on messages containing either document-

oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint.

### WS-Addressing

WS-Addressing is a way of describing the address of the recipient (and sender) of a message, inside the SOAP message itself.

### WS-Security

WS-Security defines how to use XML Signature in SOAP to secure message exchanges, as an alternative or extension to using HTTPS to secure the channel.

### XML

XML (Extensible Markup Language) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.

# Index

## L



## M



## O



## R



## S



## U

## W