Oracle® Retail Service Backbone Cloud Service

Implementation Guide

Release 19.0.000

F25617-01

January 2020

ORACLE®

Oracle® Retail Service Backbone Cloud Service Implementation Guide, Release 19.0.000

F25617-01

Primary Author: Sanal Parameswaran

Contributing Author: Nathan Young

Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

Contents

Send Us Your Comments

Oracle® Retail Service Backbone Cloud Service Implementation Guide, Release Release 19.0.000.

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 Online Documentation available on the Oracle Technology Network Web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_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 **http://www.oracle.com**.

Preface

This Implementation Guide provides a high-level overview of how the different components of Retail Service Backbone product integrate with each other.

Audience

This guide is for:

- Systems administration and operations personnel
- Systems analysts
- Integrators and implementers
- Business analysts who need information about Product processes and interfaces

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.

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 19.0) or a later patch release (for example, 19.0.1). If you are installing the base release and additional patch releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch releases can contain critical information related to the base release, as well as information about code changes since the base release.

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times not be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

This process will prevent delays in making critical corrections available to customers. For the customer, it means that before you begin installation, you must verify that you have the most recent version of the Oracle Retail documentation set. Oracle Retail documentation is available on the Oracle Technology Network at the following URL:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of a document with part number E123456-01.

If a more recent version of a document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Technology Network

Oracle Retail product documentation is available on the following Web site:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

(Data Model documents are not available through Oracle Technology Network. You can obtain them through My Oracle Support.)

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

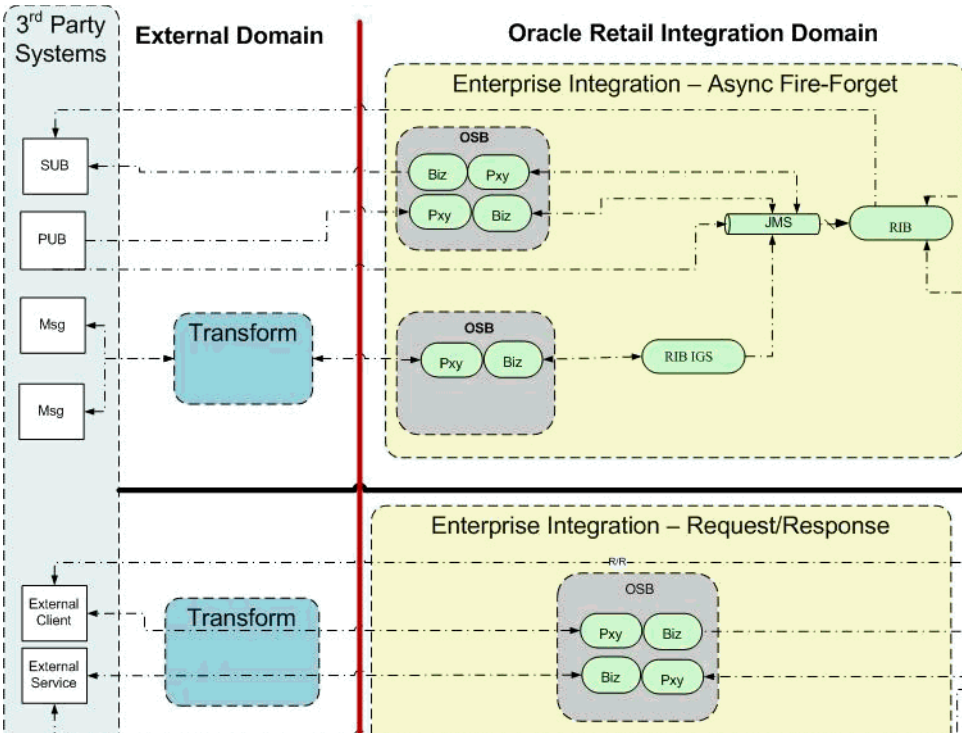
Introduction

This chapter introduces the various implementation strategies.

Oracle Retail Enterprise Integration Styles

There is no one integration approach that addresses all criteria equally well. Therefore, multiple approaches for integrating applications have evolved over time. Oracle Retail has focused on three main integration styles:

- Asynchronous JMS Pub/Sub Fire-and-Forget
- Request/Response
- Bulk Data



Request/Response Integration Style

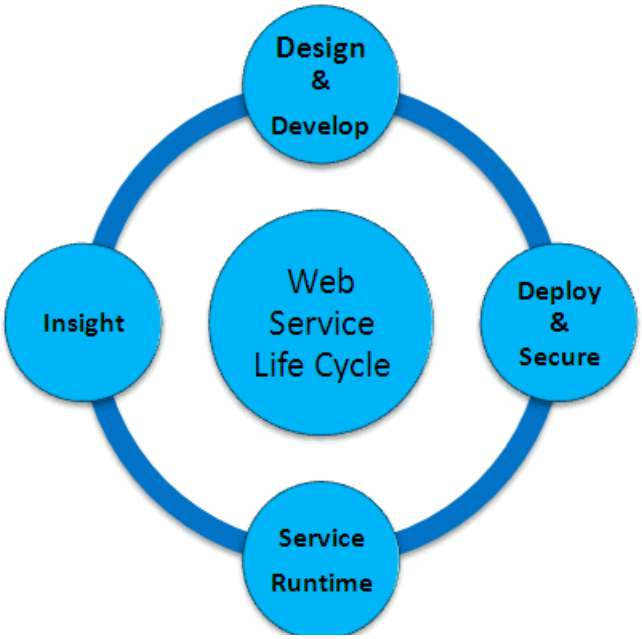
The approach taken by Oracle Retail is not to choose one style to use every time but to choose the best style for a particular integration opportunity. Each style has its

advantages and disadvantages. Applications or solutions may integrate using multiple styles so that each point of integration takes advantage of the style that suits it best.

With Retail Service Backbone (RSB), Oracle Retail has added the Request/Response Style of Enterprise Integration to the Retail Integration Suite of products. The Retail Service Backbone (RSB) is the product that defines the Oracle Retail Enterprise SOA Architecture and provides the infrastructure for the Services domain.

The Oracle Service Bus (OSB) platform is the integration infrastructure product at the core of the Oracle Retail Service Backbone (RSB) set of products.

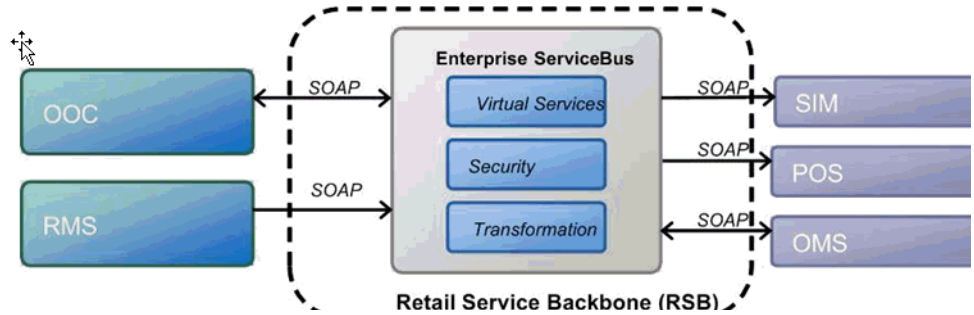
RSB products include pre-build Services and Flows and the integration API end points for all of the Oracle Retail Application's web services and the Enterprise Solution web service integration points and contracts for external application to connect to as part of the Solution business processing.



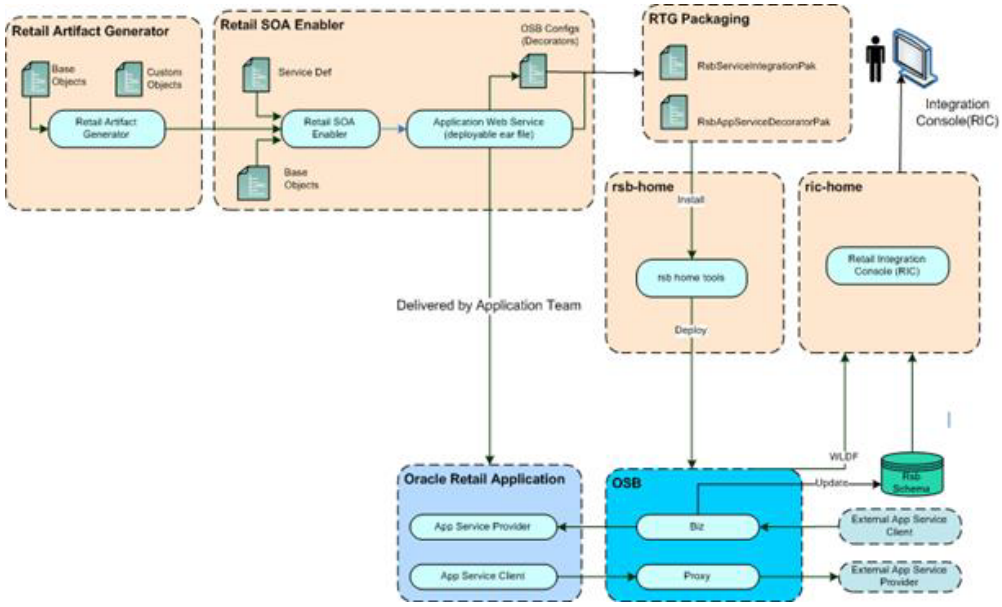
To productize Oracle Retail web services for delivery to customers, the Oracle Retail Service Backbone (RSB) also delivers as product a set of security enablement tools and the tooling required to automate the creation and packaging of the configurations for OSB and to manage the full life cycle of web services (Compile, Deploy, Patch, Monitor).

RSB enables the loose coupling between Oracle Retail and external applications and applications within Oracle Retail Suite.

- RSB provides automated OSB configurations for web service deployment and security configurations
- RSB packages all of the Oracle Retail Web Services
- RSB provides tooling for the full life cycle management of OSB hosted Web Services (Development, Compilation, Deployment and Upgrades) and automatically adds instrumentation for runtime operations monitoring (using Retail Integration Console application)



RSB features centralized software product lifecycle management. All configuration and management is from a single centralized location using specific tools to support all phases of the Software Product Lifecycle.



The framework and the tools for the common deployment model and the software life cycle management are located in rsb-home.

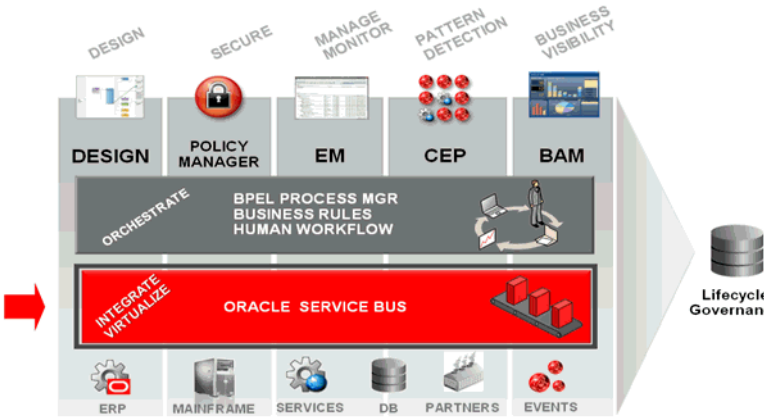
This provides an architecture and infrastructure for enterprise services, integration flows, transformations and BPEL-based components for RGBU enterprise integration solutions. Any enterprise level integration logic (complex integration flows, adapting/connector applications) can be installed, patched and upgraded from this centralized configuration location.

Note: See the Life Cycle of this document.

OSB and RSB

There are many reasons for choosing OSB as the infrastructure component for RSB.

- Oracle Service Bus is a core Oracle SOA Suite product within the Fusion Technology stack.



- OSB connects, mediates, and manages interactions between heterogeneous services, legacy applications, packaged applications, and multiple enterprise service bus instances.
- It delivers standards-based service integration for high-volume, mission-critical environments spanning the enterprise and the cloud.
- Customers are using Enterprise Service Buses (ESBs) as the integration platform of choice for integrations between Oracle Retail and their legacy and third party systems.
- Services are truly loosely coupled. Applications need not worry about integration logic, since it is completely separate and delegated to OSB.
- OSB provides built-in management and monitoring capabilities and supports out-of-the-box integration with SOA governance products.
- The OSB is specifically designed for the task of provisioning, integrating, and managing services in an SOA.
- SOA architectural style is the primary architecture pattern/strategy for Oracle going forward. Oracle Service Bus is central to Oracle's Fusion Architecture. OSB is the fundamental service infrastructure for Enterprise level SOA solution.

2

OSB Overview

Oracle Service Bus is a proven market-leading Enterprise Service Bus (ESB) built from the ground up for SOA life cycle management that provides foundation capabilities for service discovery and intermediation, rapid service provisioning and deployment, and governance. This service-infrastructure software adheres to the SOA principles of building coarse-grained, loosely coupled, and standards-based services. Additionally, OSB acts as a message brokering, service monitoring, administration, dynamic routing, and message transformation layer to infrastructure.

Oracle Service Bus relies on Oracle WebLogic Server run-time facilities. Oracle WebLogic Server provides the core services that ensure reliability, high availability, scalability, and a high-performing execution environment for your application.

Based on Oracle Platform Security Services and Oracle WebLogic security framework, Oracle Service Bus ensures service security at all levels. A comprehensive set of components for built-in security gives customers significant flexibility and choice. Users can also plug in custom or third-party security components. Built-in capabilities allow flexibility in implementation by enabling security at all levels. For authoring and managing security policies Enterprise Manager for Fusion Middleware Control (EM) is an optional product from Oracle.

Oracle Service Bus is a configuration-based, policy-driven ESB. It provides reliable service-oriented integration, service management, and traditional message brokering across heterogeneous IT environments.

3

OSB Core Concepts

OSB is the layer sandwiched between RSB and WebLogic. OSB provides its capabilities using many components including the important ones described in this chapter.

Proxy Service

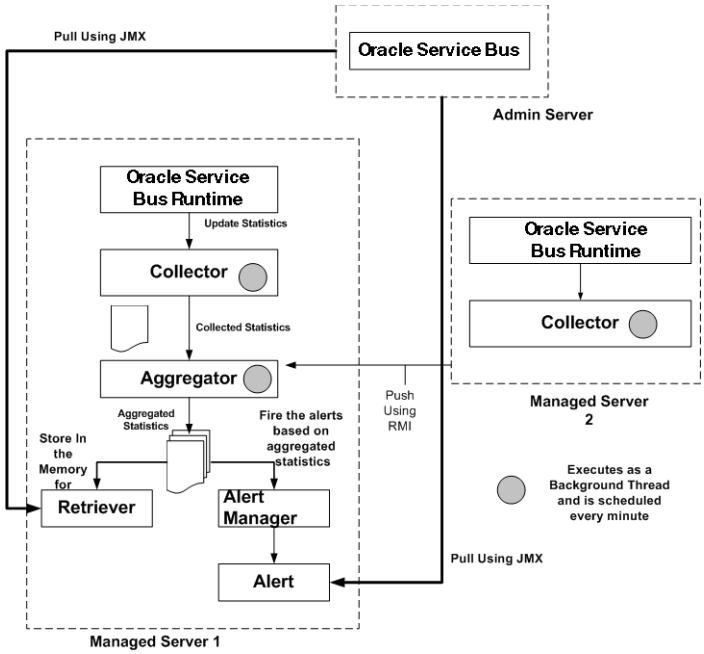
Proxy services are Oracle Service Bus definitions of intermediary Web services that Oracle Service Bus implements locally on Oracle WebLogic Server. The service consumers connect to the proxy service. For the service consumer, the proxy service provides the same interface as the application service. The only difference from a service consumer perspective is the endpoint URL. If an existing service is OSB enabled, the only programming change in the service consumer side is the change of existing URL to the proxy service URL. As the name suggests, the proxy service is a "proxy" of the actual web service.

Business Service

Business services are Oracle Service Bus definitions of the enterprise information services with which you want to exchange messages. Business services are coupled with the actual web service providers. Typically service consumers invoke proxy services. Proxy services invoke business services, which in turn invoke the service providers. This level of indirection helps to decouple the service providers and service consumers. It also helps to introduce any mediation, routing, transformation and instrumentation of the service calls.

Alerts

Alerts are generated by OSB monitoring framework. Alerts help to isolate and diagnose problems when they occur. These are captured and configured to be reported.



The Collector collects the updated statistics from Oracle Service Bus runtime and sends it to Aggregator. The Aggregator aggregates the statistics over the aggregation interval. The aggregated statistics are pushed to the Alert Manager. The Alert Manager triggers alerts based on these statistics. The aggregated statistics are also stored and can be retrieved by the Retriever.

Alerts can be viewed in OSB Console as well as (RIC).

OSB generates two types of alerts - SLA Alerts and Pipeline Alerts. SLA alerts are raised in Oracle Service Bus to indicate potential violation of the Service Level Agreements (SLAs). Pipeline alerts can be raised in the message flow of the proxy service. In RSB, the SLA alerts are called System Alerts and Pipeline Alerts are called Business Alert. The intention of this renaming is to keep the names non-technical.

As the name suggests, the system alerts are non-business alerts and business alerts are caused by the business data. An example of a System Alert is an alert triggered when a request takes more than a specified time to complete. An example of a business alert is an alert triggered when the order total is more than a million dollars. The users can create alert conditions catered to their business requirements.

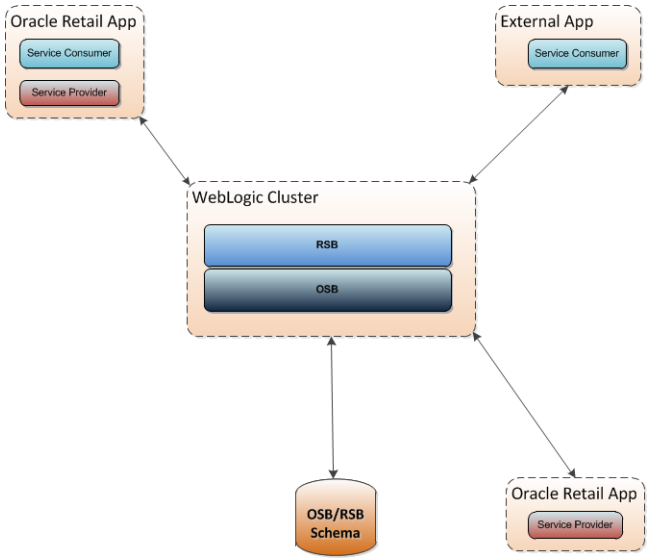
RSB comes with a default alert for each of the decorators. The alert rule name is ErrorCountAlertRule and the alert condition is Aggregation Interval 0 Hour(s) and 10 Minutes and Error Count > 0. You can view this alert in the SLA Alert Rules tab for Proxy Service in the OSB Console.

RSB does not have any default business alerts. Business alerts (Pipeline alerts) are added in the message flow of the proxy service.

See the developers guide for more information on how to add a business alert to the decorator.

Note: See also Oracle® Fusion Middleware Concepts and Architecture for Oracle Service Bus 12c Release 2 (12.2.1.3.0).

OSB and RSB High Level Architecture



RSB is built on the top of OSB components. In an architectural perspective RSB runtime can be viewed as a layer between OSB and edge applications. RSB adds instrumentation and security among other features to service invocations. Service consumers and providers communicate using RSB-OSB layer. OSB is an infrastructure product while RSB is a retail-specific integration implementation.

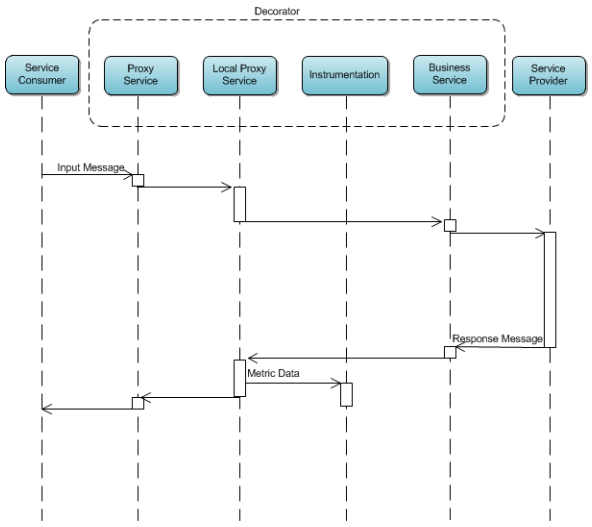
4

RSB Concepts

This chapter describes RSB concepts.

Decorators

For each service, RSB packages a proxy service, the corresponding business service and the instrumentation code as a single deployment unit. This combination of services and instrumentation is called a Decorator. It is based on the Decorator design pattern where value is added ("decorated', in other words) to the base functionality. RSB adds instrumentation code, security policies and binds the proxy service and business service together in the decorator. When a decorator is deployed both proxy service and business service are deployed along with instrumentation code. Following diagram is a sequence diagram showing the invocation call from service consumer to service provider.



Payload

A payload is the XML message that is send to and returned from web service operations. These XML messages conform to schema standards agreed upon by Oracle Retail and governed by RIB-RSB. These standards are published as XSDs and referred as Retail Business Objects (RBO)

Examples of payload types include ItemDesc, ASNInDesc, ASNOutDesc etc.

Edge App

This is a term use to refer the actual service provider like RMS, SIM, external applications etc. RSB and OSB are intermediate layers between the service consumer and service provider.

Message Family

The RSB service payloads are designed around the concept of a message family. Each message belongs to a specific message family. Each message family contains information specific to a related set of operations on a business entity or related business entities. Examples are ASNIn, ASNOut, Items and so on.

A message family may contain multiple message types. Each message type encapsulates the information specific to a business entity within one or more business events. For example, the order message family is published for events such as Create PO Header, Create PO Detail, Update PO Header, or Delete PO Detail.

This classification helps to group related messages for reporting and analysis. Examples of message family include ASNIn, Customer, Items, ItemLoc etc.

Instrumentation

Instrumentation is a method of intercepting the main flow and collecting important information for reporting or analysis. In RSB, the instrumentation code collects application name, proxy and business service operation names, timestamp, payload etc during instrumentation. The collected information is stored into a database configured at installation time. The information is collected in the most efficient way so that the primary invocation flow is least affected in terms of performance.

The RSB instrumentation captures the timestamps of the request and response, the payload of the service call and response, the service URI, operation name, application name and the result of execution. The instrumentation call is made at the response pipeline, in other words after the service call returns from the service execution in the edge app server.

The information collected by instrumentation is reported through RIC.

Note: Review the *Oracle Retail Service Backbone Integration Console Guide* for more details.

Security Policy Configurations

Between service consumer, RSB and service provider there are many options to secure the communication using web service policies. Since there are multiple layers involved in the communication, the policies chosen for each layer must be compatible. RSB supports two such sets of policy configurations; policy A and policy B. RSB comes with supports for these two configurations with scripts to automate the configurations. We believe these configurations satisfy most of the common security requirements of the customers. Any requirements outside these configurations will have to be configured manually.

Policy A is SSL with UsernameToken. The transport layer of the service invocation uses https for this policy. The consumers will have to provide the username and password for invoking the service. Policy B is message protection with

UsernameToken. Policy B should be called with http. The encryption is at the message level, instead of transport layer.

Note: Review the *Oracle Retail Service Backbone Security Guide* for more details.

5

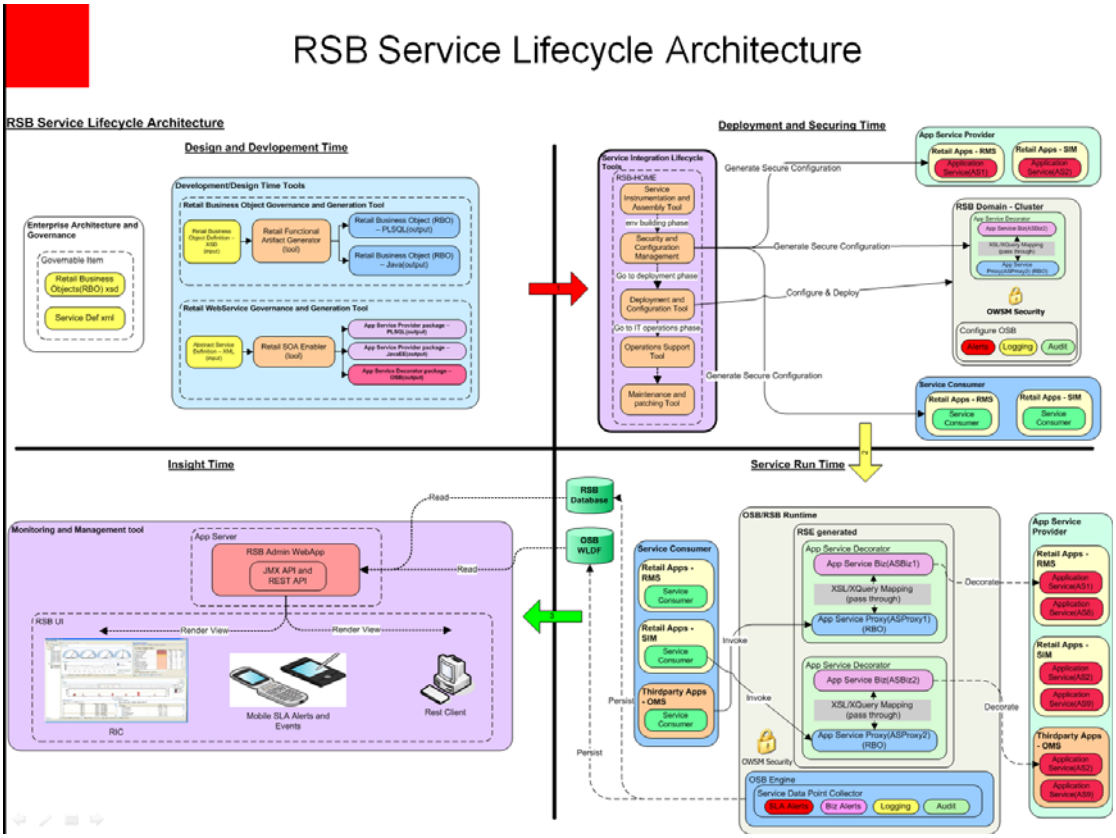
RSB Service Lifecycle

Web Services in RSB have the following lifecycle stages.

- Design and Development
- Deployment and Configuration
- Service Runtime
- Insight Time

The web services and corresponding decorator services are developed using the development time tools like Artifact Generator (AG) and Retail SOA Enabler (RSE). The web services are hosted on the edge app servers. The decorators are deployed on the RSB Server using rsb-home tools. The decorators run on the OSB runtime. The health and other service related activities are monitored using the web tools like Retail Integration Console (RIC).

The diagram below shows the architecture of the RSB service lifecycle in detail.


RSB Service Lifecycle Architecture
RSB Service Lifecycle Architecture
Design and Devlopement Time
Development/Design Time Tools
Retail Business Object Governance and Generation Tool
Retail Business Object (RBO) – PL/SQL(output)
Retail Business Object (RBO) – Java(output)
Retail Functional Artifact Generator (tool)
Retail WebService Governance and Generation Tool
Retail SOA Enabler (tool)
Enterprise Architecture and Governance
Governable Item
Retail Business Objects(RBO) xsd
Service Def xml
Deployment and Securing Time
Service Integration Lifecycle Tools
RSB-HOME
Service Instrumentation and Assembly Tool
Security and Configuration Management
Deployment and Configuration Tool
Operations Support Tool
Maintenance and patching Tool
Generate Secure Configuration
Configure & Deploy
App Service Provider
Retail Apps - RMS
Retail Apps - SIM
RSB Domain - Cluster
OWSM Security
Configure OSB
Alerts
Logging
Audit
Service Consumer
Insight Time
Service Run Time
Monitoring and Management tool
App Server
RSB Admin WebApp
JMX API and REST API
RSB UI
Render View
RIC
Mobile SLA Alerts and Events
Rest Client
Read
RSB Database
OSB WLDF
OSB/RSB Runtime
RSE generated
App Service Decorator
App Service Biz(ASBiz1)
XSL/XQuery Mapping (pass through)
App Service Proxy(ASProxy1) (RBO)
App Service Biz(ASBiz2)
App Service Proxy(ASProxy2) (RBO)
Decorate
Invoke
Persist
OSB Engine
Service Data Point Collector
SLA Alerts
Biz Alerts
Logging
Audit
Thirdparty Apps - OMS

6

RSB Components

RSB is an integration product that consists of a kernel and other independently deployable components. These components are:

- Decorators for Retails Applications
- Service Integration Flow component. These are OSB integration services that are not decorators.

RSB Kernel - Infrastructure

RSB Kernel contains the core infrastructure of RSB. It is distributed as an archive file. When extracted it builds the directory structure required for the lifecycle management of other components of RSB. The archive contains the directory structure, scripts for various lifecycle operations and libraries required for RSB.

RSB Kernel includes tools for generation of OSB projects with built-in instrumentation, tools for automating lifecycle management of services and tools for automatic enablement of certified web service security policies.

The kernel must be downloaded to a directory where you want to place the rsb-home directory structure. rsb-home serves as the central configuration management location for retail services. It also enforces standardization and verification of retail enterprise services.

On extraction of the kernel, the entire directory structure and other required scripts and jars are created as required. Since all of the lifecycle operations are based on a well defined directory structure, the location and permissions of the kernel structure is very important.

After the kernel is extracted, other components of RSB are manually downloaded and copied to the location reserved for them. Again, it is important that these components are copied to the correct location.

RSB Functional Components

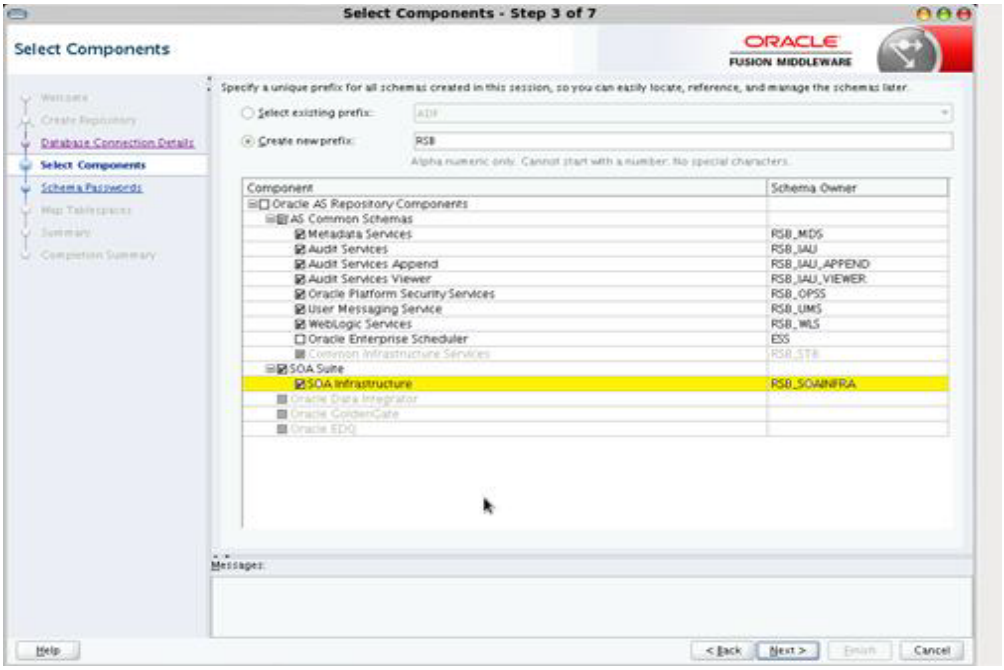
RSB Functional paks are distribution units for retail applications in the scope of integration. There are different types of functional components addressing various domains.

- App Service Decorator
- Business Process Service Decorator
- Functional Business Process
- Functional Service Integration Flow

RSB Service Infrastructure Database

OSB and OWSM (Oracle Web Service Manager) policies require runtime database schema for its functioning. These database schemas are required to be created with Repository Creation Utility (RCU). This application is available in oracle download site (http://www.oracle.com/technetwork/middleware/data-integrator/downloads/index.html). It is important that the schemas are created with this tool. Manual creation of the required database components are error prone. It is also important that the RCU version must be the one mentioned in the installation documents. If you are creating schema in a different version of RCU, the OSB system may not work properly.

While creating the schema, you should choose a prefix (The recommended prefix is RSB) and select two schemas as highlighted in the diagram below. The tool will select any other dependent schemas. Please do not deselect these.



In addition to the OSB requirement, RSB needs a table created for its own functioning. RSB Instrumentation collects meta data and payload of the services invoked and persists into this table. It is recommended to keep the RSB table in a schema separate from OSB Schema.

Note: See also RCU User Guide https://docs.oracle.com/middleware/1221/core/RCUUG/toc.htm

---

7

RSB Home

The RSB configuration and installation process follows the RSB lifecycle phases. Each of the lifecycle phases can be managed by a certain role. To support the separation of roles and responsibilities and to clearly define these phases, RSB has adopted a specific directory structure. The tools required for each of these roles are provided within this directory structure.

This directory structure supports access permissions to different tools that are managed according to the site-specific business requirements. For example, a systems administrator can be given access permissions to all the tools, while a RSB administrator or applications administrator can be provided access to only certain operation tools.

The RSB directory structure is fixed and is created by the RSB kernel tar file: **`RsbKernel<version>ForAll<version>Apps_eng_ga.tar`**.

The rsb-home is a controlled structure and there are very specific rules for using the tools and the key files within it. A key rule is that the tools scan and check versions of all files within rsb-home.

Directory Structure and Key Files

RSB uses a well defined directory structure for the lifecycle management of the services. rsb-home is the root of this directory structure. Each of the lifecycle operation has a subdirectory under it. These are

- download-home: All the component archives are downloaded into subdirectory under this directory. This directory contains following additional subdirectories.
  - all-app-service-decorator: Download location of all the decorator archives.
  - all-business-process-service-decorator: This is for future use.
  - all-functional-business-process: This is for future use.
  - all-functional-service-int-flow: Download location of all the service integration flow archives.
- service-assembly-home: This subdirectory hosts the service compilation related configurations and scripts
- deployment-home: This subdirectory has all the deployment configurations and deployment scripts.

Each of the lifecycle directories has one or more of the following subdirectories

- bin: contains all the executables
- conf: contains all the configuration files

- log: contains all the log files

The structure of the directory is shown below. Files are shown in *italics* and scripts are shown in ***bold-italics***.

```
  rsb-home
|
|---- deployment-home
|     |---- bin
|     |     |---- configure-rsb-app-server-for-security-policy-b.sh
|     |     `---- rsb-deployer.sh
|     |---- conf
|     |     |---- ddl
|     |     |     `---- RSB_INFRASTRUCTURE_SCHEMA_DEFINITION.SQL
|     |     |---- rsb-decorator-instrumentation.properties
|     |     |---- rsb-decorator-service-to-family-name-association.properties
|     |     |---- rsb-deployment-env-info.properties
|     |     |---- ddi-integration-flows.xml
|     |     |---- ddi-integration-flows.xsd
|     |     `---- rsb-integration-flows.xml
|     `---- log
|---- download-home
|     |---- all-app-service-decorator
|     |---- all-business-process-service-decorator
|     |---- all-functional-business-process
|     |---- all-functional-service-int-flow
|     |---- bin
|     |     `---- check-version-and-unpack.sh
|     `---- log
|---- integration-lib
`---- service-assembly-home
        |---- app-service-decorator
        |---- bin
        |     |---- download-app-service-wsdl.sh
        |     |---- generate-rsb-decorator-security-config.sh
        |     |---- rsb-compiler.sh
        |     `---- setup-message-protection-security-credentials.sh
        |---- conf
        |     |---- logging.properties
        |     |----
app-service-policy-id-to-decorator-service-policy-id-map.properties
        |     `---- rsb-builder-internal-trust-store.jks
        |---- log
        |---- service-integration-flow
        `---- service-policy-config
                |---- input
                |     |---- app-service-provider-wsdl
                |     `---- security-policy
                `---- output
                        |---- app-service-provider-security-policy
                        |---- decorator-service-biz-security-policy
                        |---- decorator-service-consumer-client-security-policy
                        |---- decorator-service-proxy-security-policy
                        |     `---- service-name-to-policy-id-map.properties
                        `---- decorator-service-proxy-wsdl
```

Deployment Property File

Deployment property file (rsb-deployment-env-info.properties) is where you make most, if not all, of the configurations you need to make to configure RSB for your

enterprise. The details of the properties in the file are described later in the document. During upgrades, it is recommended to backup this file and restore after extracting the RSB kernel. After any changes are made to this file, RSB must be compiled and the affected components must be redeployed.

check-version-and-unpack.sh

This script is run after the download of components or during upgrade. It checks for version compatibility and extracts each downloaded archive to its appropriate location in rsb-home.

download-app-service-wsdl.sh

This script is used only if one or more services are attached with security Policy A or B. The usage of the script is:

```
download-app-service-security-policy.sh [-a] [-s] [-u] [-h]
 -a,--app <app>  Download by application. <app> is the acronym of the application.
```

For example, rms, sim etc

```
 -h,--help Show usage information
 -s,--service <app service-name>   Download a single service. Specify app and
service name separated by space
```

As shown above the script can be called for an app or for a service. If no parameters are specified, the script downloads the WSDLs for all the apps in scope. If you are running the script for an app or a service, it checks if the corresponding app is in scope. If the services are secured with https, make sure the trust certificates are in the trust store used by the script (conf/rsb-builder-internal-trust-store.jks)

generate-rsb-decorator-security-config.sh

This script is used only if one or more services are attached with security Policy A or B.

This script is run with no parameters. It generates all the internal configuration files required to configure RSB for the selected policy. The policy information is extracted from the downloaded WSDL.

setup-message-protection-security-credentials.sh

This script is used only if one or more services are attached with security Policy B.

This script captures credential information required for Policy B configuration and stores in the wallet file. Following information is captured.

- keystore-csf-key: Keystore password for the RSB server policy B keystore.
- RSB Server Alias. This alias name is server -public-private-key-alias prefixed with short hostname (hostname without the domain name) of the RSB server. For example, rsbhost-public-private-key-alias

Following two aliases are setup for each application in scope that has Policy B configured.

- remote-host-public-key-alias prefixed with app name and short host name of the edge app server. For example, sim-simhost-remote-host-public-key-alias
- user-alias prefixed with app name and service name. For example, igs-ASNInPublishing-user-alias. This alias will store the username and password

used by the edge app for the services.

configure-rsb-app-server-for-security-policy-b.sh

This script is used only if one or more services are attached with security Policy B.

This script does the following configuration changes:

- Creates WebLogic users for the applications using policy B. The usernames and passwords are taken from the credential wallet file. These credentials are set at when the previous script (setup-message-protection-security-credentials.sh) was run.
- Updates domain wallet file for OWSM security configuration. For each app the encryption alias and passwords set the service assemble phase is read from the wallet file. The domain wallet file is updated with the information collected at the time of setup.
- Updates the JPS config with keystore file (<server host> + -keystore.jks) and private key alias (<server host> + -public-private-key-alias)

rsb-compiler.sh

This script compiles the configurations into the deployable components. Any change in the configuration must be compiled to take effect. When you are running the script for the very first time it should be called as

> rsb-compiler.sh -setup-security-credential

This is to setup the security credential information for databases, WebLogic console, and so on. The security information is persisted to rsb-home/deployment-home/conf/security folder in an encrypted credential wallet file (cwallet.sso). For subsequent compilations the -setup-security-credential parameter is not needed. It is recommended to backup the security folder to save data entry in case of any reinstallation.

rsb-deployer.sh

This script deploys the specified component to WebLogic server. The WebLogic server must be running for the deployment to work. The script has many options as described below. You cannot combine options in a single step.

-deploy-rsb-service <Decorator Jar File>

This option is used to deploy a single decorator. Specify the jar file name as the parameter.

-deploy-all-rsb-service

This option is used to deploy all the decorator services for all the apps in scope.

-deploy-all-rsb-service-for-app <appName>

This option is used to deploy all the decorator services for the specified app.

-prepare-wls

This option is used to create and configure WebLogic for RSB installation. This option does the following configurations:

- Creates the required data sources for RSB (jdbc/rsbAdminDs). Database user ID and password are taken from the wallet file created at compile time.
- Sets the log file size and number of log files to keep
- Creates the Log Filter (RsbLogFilter) and log broadcasting configurations.
- Creates the data retirement policies for all servers.

-undeploy-rsb-service <Decorator Jar Name>

This option is used to undeploy a single service.

-undeploy-all-rsb-service

This option is used to undeploy all the decorator services for all apps in scope.

-undeploy-all-rsb-service-for-app <appName>

This option is used to undeploy all the services of the specified app.

8

RSB Paks

This chapter describes RSB paks.

App Service Decorator

Decorator paks contain the proxy and business service definitions of the services provided by the application. There is one pak for each application. Decorators are generated using Retail SOA Enabler (RSE) tool. The RSE tool uses the service definition xml file as input for generating the decorators. The generated decorator is included in the archive generated by RSE. The decorator paks are available as zip archives. The decorator paks of the applications in scope must be downloaded and copied into the specified directory in the service-assembly-home directory.

The list of all app service decorator paks in this release is given in Appendix A.

Note: See also the *Oracle Retail SOA Enabler (RSE) Guide*.

Business Process Service Decorator

This is not currently used, it is a placeholder for future functionality.

Functional Business Process

This is not currently used, it is a placeholder for future functionality.

Functional Service Integration Flow

RSB Functional Integration Flows are OSB integration services that are not decorators. Decorators have one proxy service and one business service tied to the proxy. Any other type of integration service should be placed under this category.

This pak contains the proxy service which serves as the router service for the incoming payload. Depending on the message family and message type of the incoming payload, the proxy service routes the service to appropriate decorators.

9

RSB and RIB

RSB implements a synchronous request-reply based integration pattern while RIB implements near real time asynchronous fire and forget integration pattern. RSB is implemented using Web Services while RIB uses JMS. RSB and RIB can exist independently. However, they are different types of integration and in many cases they complement each other. They can co-exist in an enterprise. Both implementations can be bridged using IGS for RIB bound transactions and Injector service for RSB bound transactions.

In RSB integration errors have to be handled by the consumer. RSB does not offer any retry mechanism like RIB. RIB and RSB are not mutually exclusive. Both can be implemented together and both RIB and RSB can communicate with each other. RSB does not replace RIB. It complements RIB with the request reply type of integration. RSB can be clustered in active - active mode. RSB does not need sequencing of messages.

RIC is designed as a consolidated view of both RSB and RIB, if each exists. The current version of RIC shows RIB configuration and transaction data, in addition to RSB configuration and transaction data.

Note: See also the *Oracle Retail Integration Console Guide*.

Table 9–1 RIB Vs RSB

RIB	RSB
JMS Publish - Subscribe	Web Services on OSB
Guaranteed Delivery	Consumer gets error on failure
Failed Message Recovery	Consumer has to manage errors
Fire-and-Forget	Request Response
XSD Schema	XSD, XSLT, WSDL
	Remote Invocation
Only active-passive cluster supported	Active-active cluster supported
Near real-time response	Real-time response

10

RSB and JMX

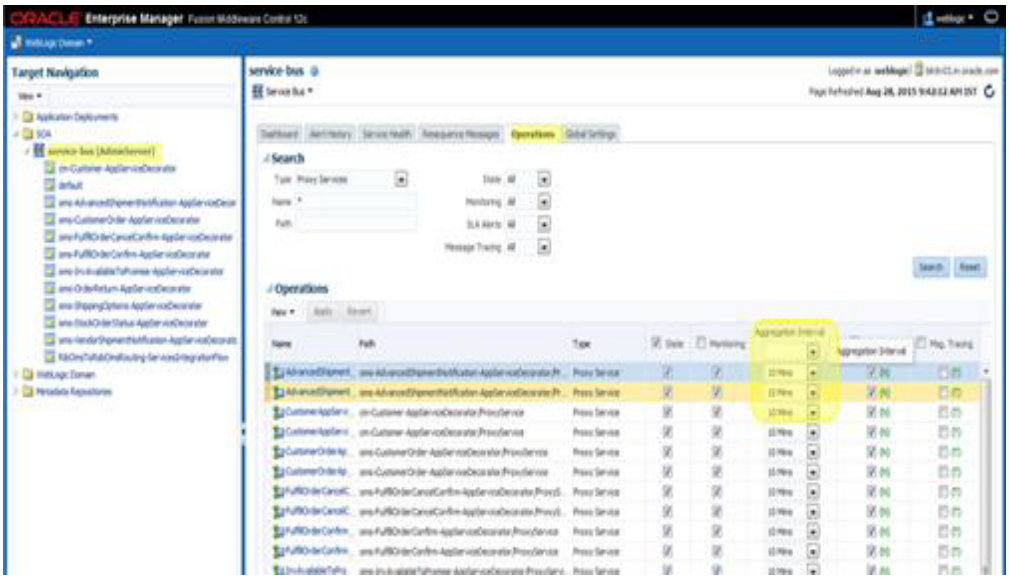
This section describes the RSB JMX infrastructure. JMX is a specification that provides capability for runtime management of Java components. WebLogic server implements JMX specification. RSB uses JMX for collecting useful information for monitoring the functioning of the integration infrastructure.

Any third party JMX console compatible with the Java EE container can be used to manage RSB components. In RSB some of information captured by OSB MBeans are collected and reported in context through the administrating application (RIC).

Note: See also JMX Specification **http://docs.oracle.com/javase/7/docs/technotes/guides/jmx/JMX_1_4_specification.pdf**.

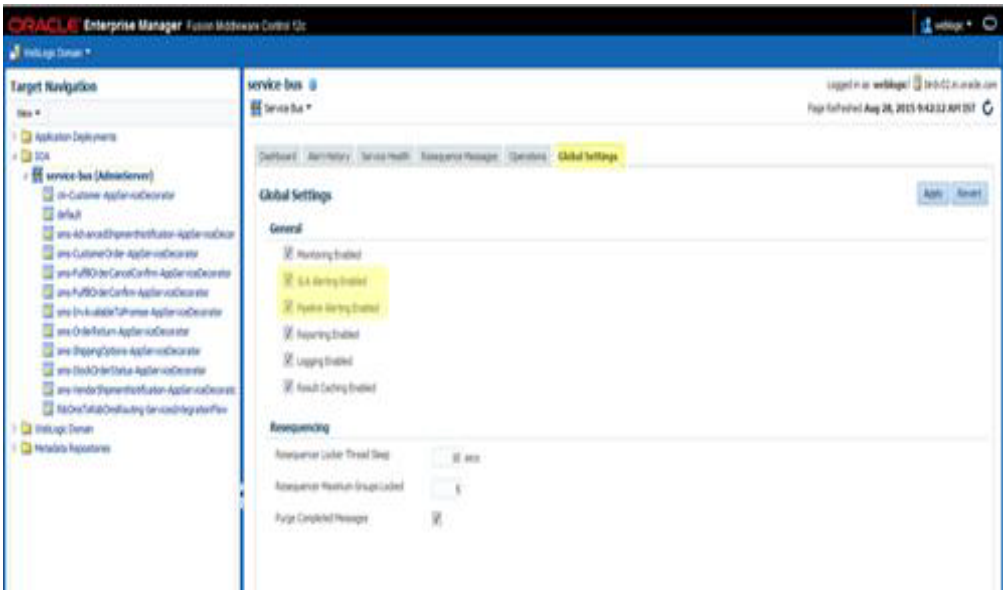
Metrics Data

In RIC, the following metrics are captured using JMX. The metrics data is collected by OSB from all the managed servers on regular interval. The default interval is 10 minutes. The interval for each service (Proxy and Business) can be changed in EM console. The following graphic shows a proxy service aggregation interval field location, in case it is required to change.



RIC retrieves the metric data from EM when the user visits the Metrics Graphs tab. Since EM metric collection is decided by the collection interval, the data shown in RIC can be out of sync by a maximum delay of the interval. So, in a default configuration the metric data can be outdated by a maximum of 10 minutes.

In order for the alerts to work, it should be enabled in the global settings. These are enabled by default. Following screen shows the location to enable alerts in the EM Console.



Success and failure count

This count shows the number of successful invocations and number of failures of decorator and app service for the past 24 hours.

Min, max and average response time

This information shows minimum, maximum and average response time of service operations for the past 24 hours.

Success and Failure count for operations

This is a count of success and failure by operations for the past 24 hours.

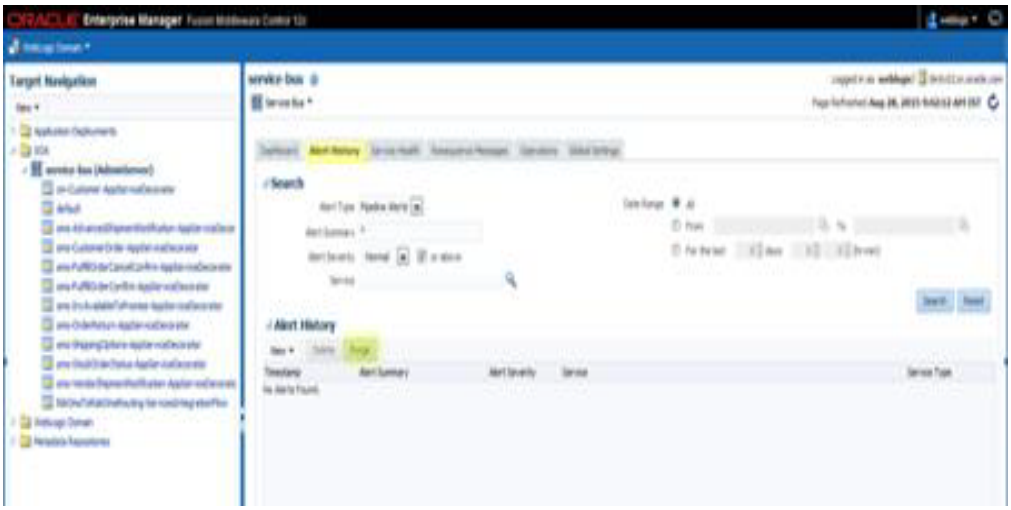
Business and SLA Alerts

Review the *Oracle Retail Service Backbone Developers Guide* for instructions on adding Business and System Alerts.

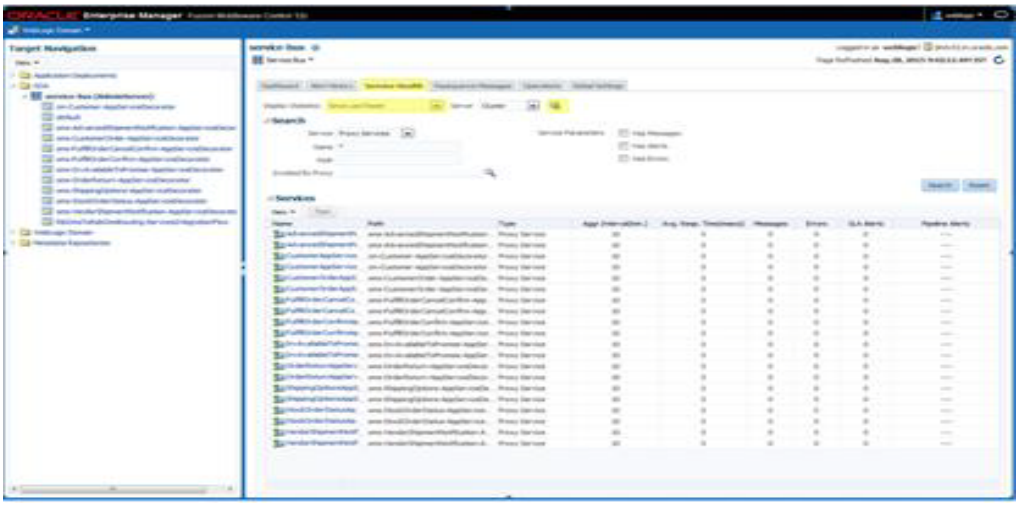
Alert Purging

The performance of alert retrieval is affected by the number of alerts in the system. As the number of alerts grows, the retrieval becomes slow. This becomes evident in the alert screen of RIC application. The screen rendering gets slower as the number of alerts increases over time. Here are a couple of ways to purge the alert.

- Purge the alert through EM console. From the EM console dashboard navigate to the Alert History for SLA and Pipeline alerts. Search for an alert based on a specific search criteria and click on Purge to purge the alert.



- The following graphic shows how to reset statistics using EM Console
  - Select **Service Health** tab.
  - Choose Since Last Reset option for Display Statistics drop-down.
  - Click **Reset Statistics**.



Note: See also Oracle Service Bus Guides.

11

RSB Tools

RSB comes with following supporting tools to help the implementation.

SIT Tools

There are two types of SIT tools: JSIT and PSIT. SIT tools are used to mimic provider edge applications. JSIT mimics Java applications and PSIT mimics PLSQL applications.

Since RSB is an integration product, it cannot be tested standalone. Ideally it needs the edge applications deployed. SIT tools comes handy in situations where one or more edge applications are not available. The role of the edge application can be taken by the SIT tool. In fact, RSB configuration can be validated without even a single edge app.

A single instance of JSIT/PSIT can mimic one or more edge apps.

The SIT Tool distribution is a separate download. In this release SIT tools are supported only on JEE 7 servers (e.g., WebLogic 12c Release 2 (12.2.1.3.0)).

JSIT

JSIT delivers a Java EE application called **`javaee-service-interface-tester-<version>.ear`**. The RSE generated Java EE provider ejb-jar for a provider edge application gets merged into the JSIT ear (**`javaee-service-interface-tester-<version>.ear`**) at application assembly time and can be deployed to Glassfish (Java EE 7) and WebLogic 12c Release 2 (12.2.1.3.0) (Java EE 7+) application server. After deployment, all the web services in the RSE ejb-jar get exposed by the JSIT ear. You can use any WSDL consumer API/Tool to call the web services immediately after deployment without any lengthy configuration process.

SOAP-UI works very well to create the service consumer (WSDL client) without having to write any code, but application developers can use any API of their choice to call the WSDL based web service. When the web service gets called the JSIT engine will generate a mock but fully defined response soap xml with dummy data. It generates a hash of the request soap xml and persist the request/response data for future/subsequent use. The user can modify and fine tune their response soap xml by editing the persisted data using the JSIT webapp which is also included in the JSIT ear **`(javaee-service-interface-tester-<version>.ear`**).

The JSIT Tool can be accessed through the URL
http://<hostname>:<port>/javaee-service-interface-tester-web19.0.000

Download and Prepare JSIT

1. Download and extract JavaEeSit19.0.000ForAll19.x.xApps_eng_ga.zip
2. Copy javaee-service-interface-tester-<version>.ear to an install stage folder, referred as SIT_JAVAEE_APP_HOME.
3. Download and save RSE generated JavaEE ejb-jar (<app>-service-ejb.jar) in SIT_JAVAEE_APP_HOME. <app> is the application name that hosts the application service. For example, sim-service-ejb.jar.

   Merge the two components:

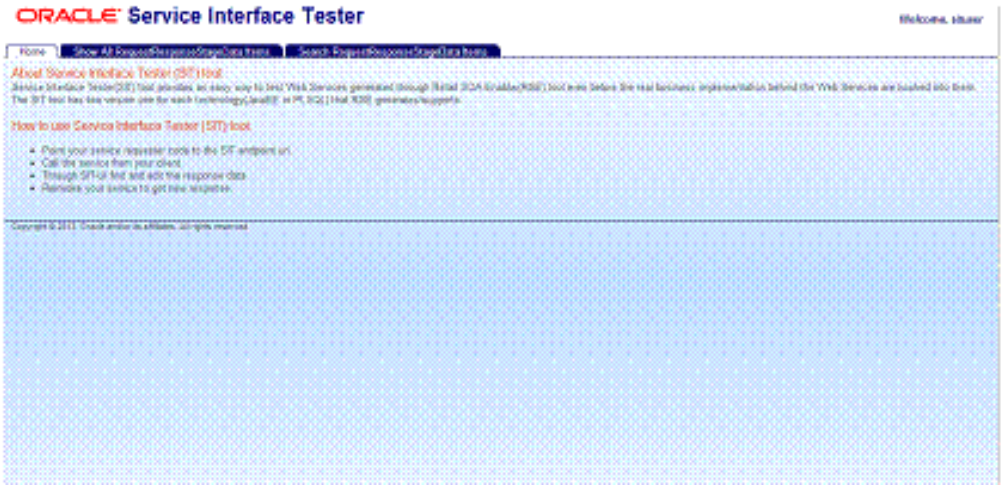
   ```
   jar -uvf javaee-service-interface-tester-<version>.ear <app>-service-ejb.jar
   ```

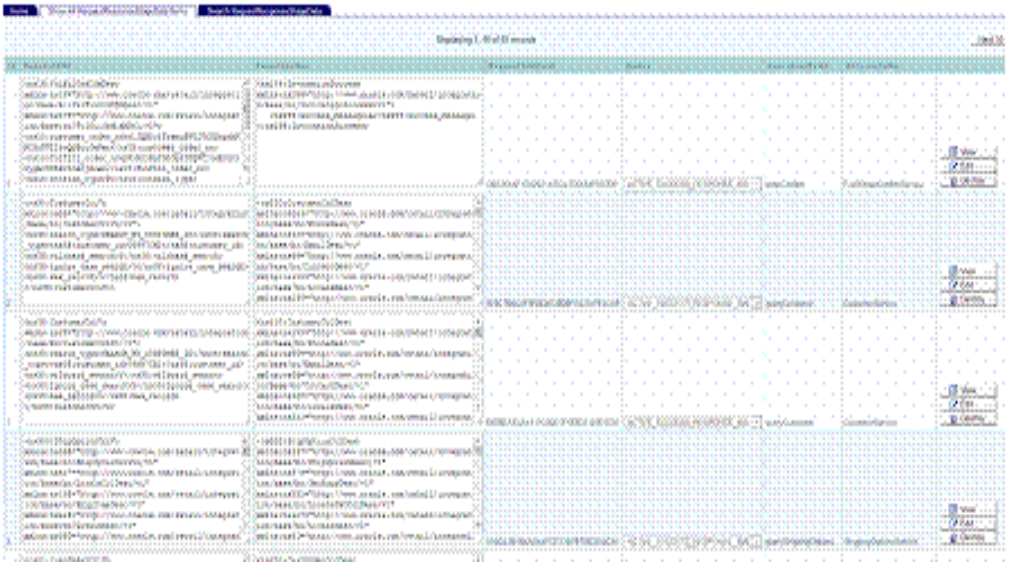
   The above command can be repeated with any number of <app>-service-ejb.jar files.

Deploy JSIT to WebLogic 12c Release 2 (12.2.1.3.0)

1. Open WebLogic 12c Release 2 (12.2.1.3.0) Console.
   a. Go to the Deployments page and click **Install**. Upload or browse to javaee-service-interface-tester-<version>.ear. Click **Next**.
   b. Select Install this deployment as an application. Click **Next**.
   c. Select the target server. Click **Next**.
   d. Click **Finish to Deploy**.
   e. Configure a group and user for JSIT. Click **Security Realms**.
   f. Click the default realm.
   g. Click the **Groups** tab. Add a group called sitadmin.
   h. Click the **Users** tab. Add a new user. Add the new user to the sitadmin group.

Note: Do not change the default application name. It must be kept as javaee-service-interface-tester-<version>. JSIT web app is accessible at **`http://<host>:<port>/javaee-service-interface-tester-web-<version>/faces/index.xhtml`**.





PSIT

This section describes the PSIT related details.

Installation and Setup

To install and set up PSIT, take the following steps:

1. Extract the PlsqlSit19.0.000ForAll19.x.xApps_eng_ga.tar

   For example, tar -xvf PlsqlSit19.0.000ForAll19.x.xApps_eng_ga.tar

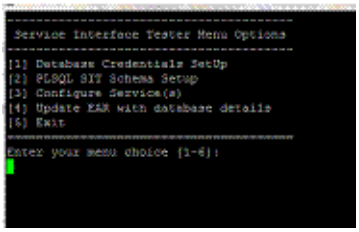
   This creates plsql-service-interface-tester.

2. Update database details in psit.properties available under plsql-service-interface-tester/conf folder.

3. Launch SIT Menu to perform plsql sit operations.

   For example, sh sitmenu.sh

SIT Menu looks as shown in following window:



Menu Options

Database Credentials Setup

Database Credential Setup is a onetime operation. It is done corresponding to server details provided in psit.properties files. Make sure that server name, port and sid are correct before performing this step.

1. Select Menu Option 1 to setup Database credentials.
2. Console provides with option to add/modify/delete credentials. Select appropriate option and once done, press D to return to the main menu.

PLSQL SIT Schema Setup

It is a one-time process which creates oracle objects in the database. Before starting the setup, make sure that all Oracle objects (that is, sqls) are copied into Scripts folder from the retail-public-payload-database-object-types-<version> located under RibFuncArtifact<Version>ForAll<version>Apps_eng_ga. Only the scripts present in the scripts folder are executed for schema setup.

1. Select Menu Option 2 to start the setup.
2. On success, console returns Schema Setup Complete message with no errors or exceptions.

Configure Service(s)

Configuring Service(s) operation creates procedures and implements procedure body in database. It takes ServiceProviderDefLibrary.xml from plsql-service-interface-tester/conf location as input and generates StoredProcedureMetaData.xml. By default, a ServiceProviderDefLibrary.xml is provided in the conf folder.

1. Select Menu Option 3 to start the configuring service(s) listed in ServiceProviderDef.xml.
2. On success, console returns Configuration Complete message with no errors or exceptions.

Update EAR with Database Details

This option will update the plsql-service-interface-tester-web-<version>.war file with database credentials provided in psit.properties

Testing PSIT

1. Get the live/deployed RSE generated web services (for example, PayTermService) WSDLs

   Your Web Browser > WebLogic Deployment page > RSE generated Web Service's ear > Testing > Expand Service (for example, PayTerm) > Under Testing point, click? WSDL > Copy the WSDL URL from browser.

2. Start SOAPUI and create a new project (if needed) to import the WSDL.
   - Right click and add/import the WSDL into the project.
   - Open Request 1 window for your test web service operation.
   - Edit the SOAP Request XML in the request editor. Find ? and replace-all with 1 for a quick test or hand edit the request xml with your test data.
   - Click the > (submit request) button on top. If everything went okay you should get a SOAP response XML (not fault) in the response window.

The SOAP response XML is a mock response, automatically generated by PSIT.

Deploying the plsql-service-interface-tester-app-19.0.000.ear

Take the following steps to deploy the ear file:

1. The plsql-service-interface-tester-app-19.0.000.ear is available in PlsqlSit19.0.000ForAll19.x.xApps_eng_ga.tar download.
2. Edit psit.properties with database credentials of PSIT schema
3. Execute the Menu Option 4 from SIT Menu, this option updates the ear file with database credentials provided in psit.properties.
4. Deploy the ear file on the application server.
5. Configure a group and user for PSIT. To do this, log in to the WebLogic console and click **Security Realms**.
6. Click the default realm.
7. Click the **Groups** tab. Add a group called sitadmin.
8. Click the **Users** tab. Add a new user. Add the new user to the sitadmin group.
9. Verify the UI is accessible from by using launch option from the WebLogic console.

RIC

Retail Integration Console (RIC) is a web based application distributed along with RSB. This application is deployed as a part of the RSB installation. The application can be access by the URL **`http://<host>:<port>/rsb-admin`**.

Note: Review the *Oracle Retail Service Backbone Integration Console Guide* for more details.

12

Instrumentation in RSB

In RSB the service consumers and providers communicate through RSB layer. The consumers call a proxy service in RSB and RSB calls the service providers using business service. This decoupling provides many benefits, once of which is the opportunity for instrumentation. The service calls are intercepted and useful information extracted and persisted without affecting the functionality. RSB uses instrumentation to get the meta data of services for reporting, analyzing and troubleshooting.

Decorator Service Instrumentation in RSB

In RSB the instrumentation is done in the decorator module. The service invocation is intercepted and the data fields are captured and persisted to the database configured at the deployment. This process is done asynchronously so that performance of the main flow is not directly affected. The data fields are persisted to rsb_service_activity table.

RSB_SERVICE_ACTIVITY Table: Table Structure Definition

Table 12–1

Column Name	Data Type	Nullable?	Description
ACTIVITY_NUM	NUMBER(19,0)	No	Generated unique number.
ACTIVITY_STATE	VARCHAR2(8 BYTE)	Yes	Can have RESPONSE or ERROR as value.
APPLICATION_ NAME	VARCHAR2(12 BYTE)	Yes	Name of the service provider application.
BUSINESS_ OPERATION_ NAME	VARCHAR2(256 BYTE)	Yes	The service operation name.
BUSINESS_ SERVICE_NAME	VARCHAR2(256 BYTE)	Yes	The name of OSB business service.
BUSINESS_ SERVICE_URI	VARCHAR2(256 BYTE)	Yes	Not populated by OSB.
ERROR_CODE	VARCHAR2(64 BYTE)	Yes	Error code returned by service provider.
ERROR_DETAIL	VARCHAR2(2048 BYTE)	Yes	Not populated by OSB.

Table 12–1 (Cont.)

Column Name	Data Type	Nullable?	Description
ERROR_REASON	VARCHAR2(1024 BYTE)	Yes	Error message returned by service provider.
MESSAGE_ECID	VARCHAR2(64 BYTE)	No	Generated ID.
MESSAGE_ FAMILY	VARCHAR2(25 BYTE)	Yes	Retail Message Family.
PROXY_ OPERATION_ NAME	VARCHAR2(64 BYTE)	Yes	Name of the OSB proxy service operation.
PROXY_SERVICE_ NAME	VARCHAR2(256 BYTE)	Yes	Name of the OSB proxy service.
PROXY_SERVICE_ URI	VARCHAR2(256 BYTE)	Yes	URI of the proxy service.
REQUEST_ TIMESTAMP	TIMESTAMP(6)	Yes	The timestamp of the service request.
REQUEST_XML	CLOB	Yes	XML payload of the service request.
RESPONSE_ TIMESTAMP	TIMESTAMP(6)	Yes	The timestamp of the service response.
RESPONSE_XML	CLOB	Yes	XML payload of the service response.

Note: REQUEST_XML and RESPONSE_XML are populated only when the audit is enabled. See Appendix D for more details on the RSB Service Activity Table.

Broadcasting Logs

In a clustered environment the RSB components may run in any of the managed servers depending on the load of the system. Thus the log messages are distributed across the managed server logs. This posts a huge hurdle in debugging issues, since there is no way to predict which component runs in which managed server. RIC solves this problem by broadcasting the logs from all the managed servers to the admin server and providing a single place to view the consolidated log. The consolidated log file is retrieved from the server using WebLogic Diagnostic Framework (WLDF). The consolidated log can be viewed using RIC or by directly viewing the <domain name>.log in the servers/AdminServer/logs folder in the domain home. The log file name and related attributes are set in the WebLogic server during the -prepare-wls step of the deployment.

13

Pre-Implementation Considerations

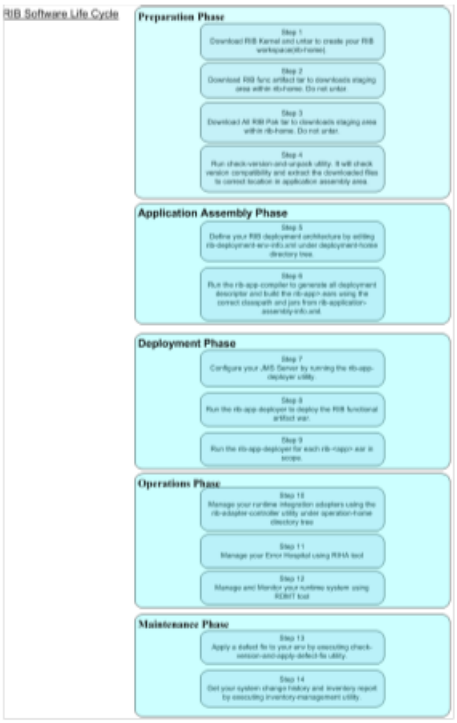
Before the RSB is installed in an enterprise, there are many factors that need to be considered. Planning and addressing each of the factors will avoid having to reinstall or re-architect because of performance or operational problems.

The process of RSB implementation requires the creation or modification of a retailer's Enterprise Integration Architecture. Typically, retailers will already have an integration strategy, plan or architecture and products in place to integrate their current systems.

Life-cycle Management

Software applications, after being made generally available (GA), have a well defined lifecycle process. The implementer must manage and perform tasks in these phases.

- Acquire the software components
- Prepare the environment
- Assemble the application
- Deploy and start the application
- Perform day-to-day monitoring to make sure the application is running properly
- Apply code fixes to the application



RSB supports and follows the RSB Software Lifecycle Management, a well-defined process life cycle that has implemented specific tools and functionality for each of these phases.

- **Preparation Phase**: In this phase all relevant components are downloaded, extracted configured, and version compatibility checks done.
- **Application Assembly Phase**: In this phase, site specific configuration changes are completed. All the relevant RSB decorators and service integration flows are generated.
- **Deployment Phase**: In this phase, using the decorators and administration application created in the previous step and the site specific information present in a global configuration file, the decorators and applications are deployed to the application server instances.
- **Operations Phase**: In this phase, day-to-day operations of the decorator applications are performed.
- **Maintenance Phase**: In this phase, code fixes, patching and configuration changes and maintenance of the RSB are performed.

High Availability Considerations

As businesses are maturing and having to do everything quicker, better, faster, and with less resource and money they are pushing similar expectation onto their IT infrastructure. Business users are expecting more out of their IT investments, with zero down time, consistent predictable responding systems which are highly available has become basic requirements of today's business applications.

Modern business application requirements are classified by the abilities that the system must provide. This list of abilities such as availability, scalability, reliability,

auditability, recoverability, portability, manageability, and maintainability determine the success or failure of a business.

With a clustered system many of these business requirement abilities gets addressed without having to do lots of development work within the business application. Clustering directly address availability, scalability, recoverability requirements which is very attractive to a business. In reality though it is a trade-off, clustered system increases complexity, is normally more difficult to manage and secure and so one should evaluate the pros and cons before deciding to use clustering.

Oracle provides many clustering solutions and options; those relevant to RSB are WebLogic server clusters and Oracle database cluster (RAC).

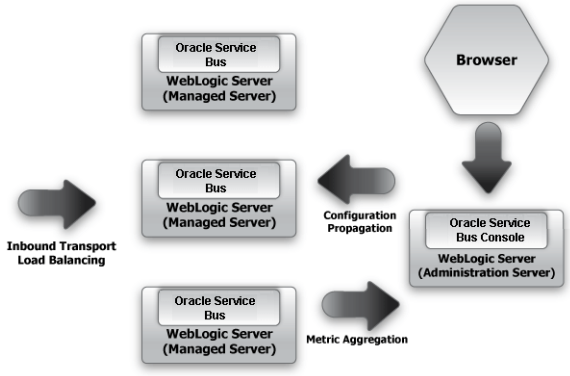
WebLogic Server Cluster Concepts

A WebLogic server cluster consists of multiple WebLogic server instances running simultaneously and working together to provide increased scalability and reliability. A cluster appears to clients to be a single WebLogic server instance. The server instances that constitute a cluster can run on the same machine, or be located on different machines. You can increase a cluster's capacity by adding additional server instances to the cluster on an existing machine, or you can add machines to the cluster to host the incremental server instances. Each server instance in a cluster must run the same version of WebLogic Server.

In an active-passive configuration the passive components are only used when the active component fails. Active-passive solutions deploy an active instance that handles requests and a passive instance that is on standby. In addition, a heartbeat mechanism is usually set up between these two instances together with a hardware cluster (such as Sun Cluster, Veritas, RedHat Cluster Manager, and Oracle CRS) agent so that when the active instance fails, the agent shuts down the active instance completely, brings up the passive instance, and resumes application services.

In an active-active model all equivalent members are active and none are on standby. All instances handle requests concurrently.

An active-active system generally provides higher transparency to consumers and has a greater scalability than an active-passive system. On the other hand, the operational and licensing costs of an active-passive model are lower than that of an active-active deployment.



Note: See also Oracle® Fusion Middleware Using Clusters for Oracle WebLogic Server.

Oracle Database Cluster (RAC) Concepts

A cluster comprises multiple interconnected computers or servers that appear as if they are one server to end users and applications. Oracle Database Real Application Clusters (Oracle RAC) enables the clustering of the Oracle database. Oracle RAC uses Oracle Clusterware for the infrastructure to bind multiple servers so that they operate as a single system.

Single-instance Oracle databases have a one-to-one relationship between the Oracle database and the instance. Oracle RAC environments, however, have a one-to-many relationship between the database and instances. In Oracle RAC environments, the cluster database instances access one database. The combined processing power of the multiple servers can provide greater throughput and scalability than is available from a single server. Oracle RAC is the Oracle database option that provides a single system image for multiple servers to access one Oracle database. In Oracle RAC, each Oracle instance usually runs on a separate server.

Oracle RAC technology provides high availability and scalability for all database applications. Having multiple instances access a single database prevents the server from being a single point of failure. Oracle RAC enables capability to combine smaller commodity servers into a cluster to create scalable environments that support mission critical business applications.

Recommended Deployment Options

There are no physical location constraints on where RSB and edge applications are deployed as long as they are visible from the same network. But the decision on where to physically and logically locate your decorators and applications has impact on the high availability, performance and maintainability of your integration solution, so this decision must be given careful consideration.

Cluster Deployment

The supported deployment configuration for RSB is cluster. However, for development purposes non-cluster deployment can be used.

A cluster configuration will have following server instances:

- AdminServer
- HTTP Proxy server
- One or more managed servers

AdminServer is used to manage the cluster. Optionally, you can also use a node manager for startup and shutdown of managed servers.

The HTTP Proxy server acts as the load balancer proxy server for the cluster. The service consumers will connect to the proxy server port. The proxy server will redirect the call to one of the managed servers, depending on the load balancing algorithm chosen at the configuration time. The default load balancing algorithm is round-robin.

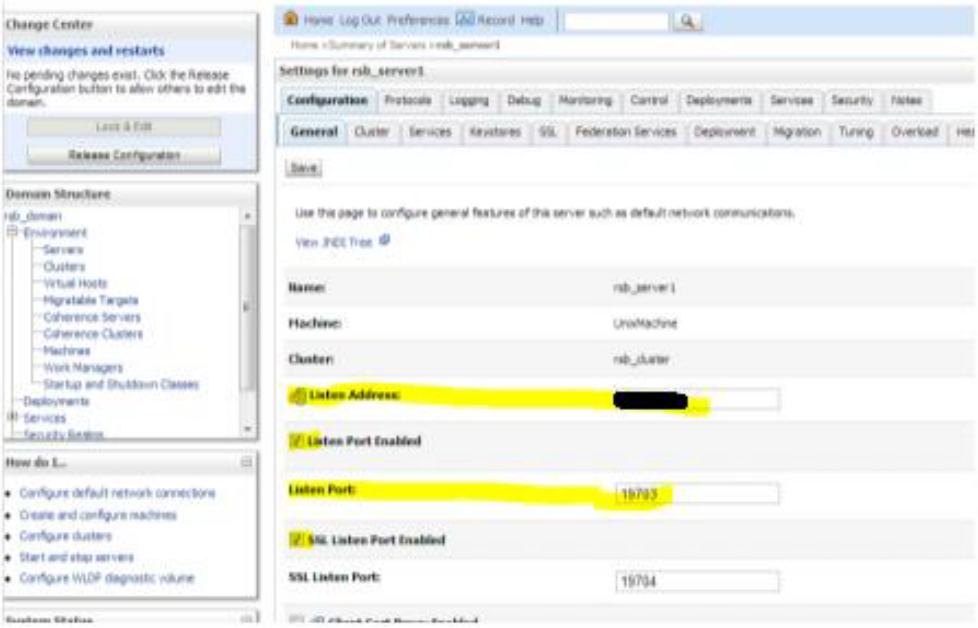
Managed servers do the actual work of the service invocation. The decorators are deployed to the managed servers. A cluster can have one or more managed servers. Even though cluster configuration supports one managed server, it really doesn't make sense to have a cluster with single managed server. So for all practical purposes, the minimum number of managed servers must be two. The configuration of all the managed servers in the cluster is same except for one of the managed server. One managed server has a special role of hosting singleton components in addition to

being a member of the managed server pool. Typically, this is the first managed server configured in the cluster. There is no user input needed to identify or configure this managed server.

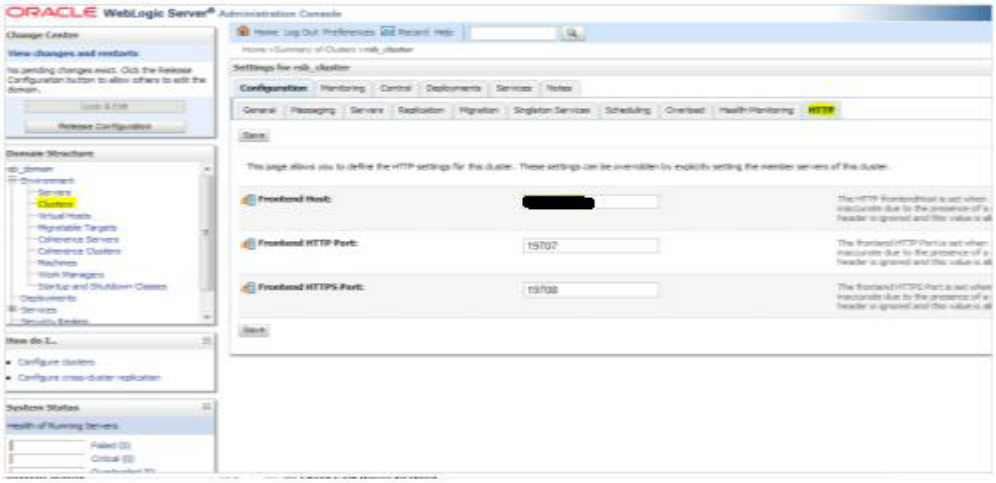
HTTPS Cluster Configurations

If you want the cluster configured for https, you have to do a few additional configurations to make it work in RSB.

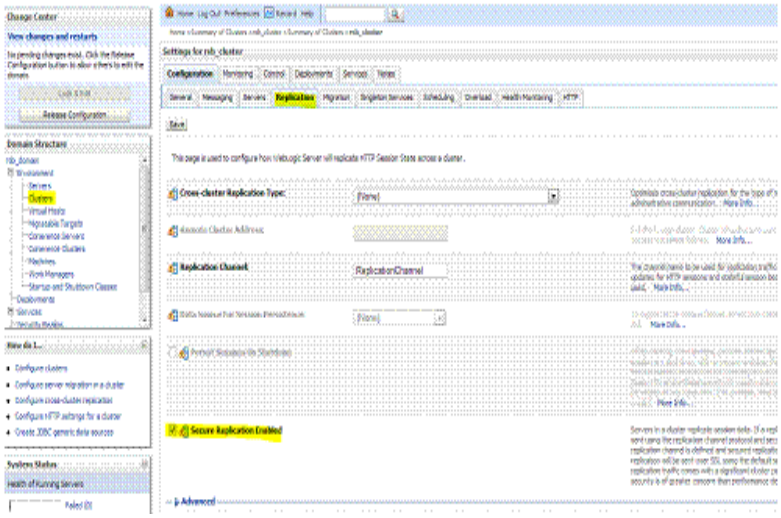
1. Enable https port for AdminServer, Http Proxy Server and all managed servers. Specify the Listen Address. The Listen Address must match the CN entry of the server certificate. Sometimes the CN entry of the server certificate is the fully qualified name (for example, rsbhost.example.com) instead of the short hostname (for example, rsbhost). If the entries do not match, the security configurations will not work.



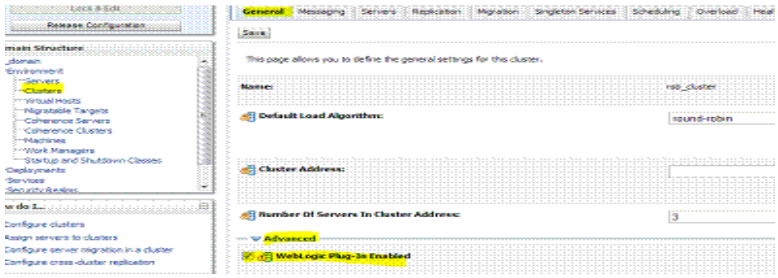
2. Set the Front end Host (as wells as HTTP and HTTPS ports, if not configured already) for the cluster. This should match the CN entry of the certificate. Environment > Clusters > [cluster name] > HTTP.



3. Enable Secure Replication Enabled option available in Environment > Clusters > [cluster name] > Replication.



4. Enable WebLogic plug-in. Select the WebLogic Plug-In Enabled option in Environment > Clusters > [cluster name] > Advanced.



Who Should Use This Configuration?

All the production deployments must use the cluster configuration. This configuration is appropriate when the machine hosting WLS is adequately sized for its job. Development environment may use a non-cluster deployment configuration.

14

Implementation Process

This release of RSB defines the full life cycle of the RSB software product. The RSB life cycle and phases are described in detail in the software lifecycle management section of this document. For every life cycle phases and task that RSB defines it provides corresponding tools and utilities to manage and operate on those phases.

There are several prerequisite steps that should be followed to have a successful RSB installation and deployment.

- Understand the RSB Core Concepts.
- Understand the deployment options.
- Understand the RSB life cycle.
- Understand the physical and logical requirements and limitations of the RSB Components.
- Understand the RSB Operational considerations.

The process of implementation should follow these general steps:

- Work with the teams at your organization dedicated to Oracle Retail to coordinate plans for the number and type of environments needed (for example, Dev, Integration, Production).
- Each type of environment needs to be sized, deployed, and managed in conjunction with the implementation of the Oracle Retail applications.
  - It is critical to understand the volume requirements of the production system so that the appropriate decisions can be made about the deployment option and the physical location and sizing.
- All deployments have integration to existing retailer systems. It is critical to understand the position of the RSB as it fits into the overall integration architecture and that the current operations and architecture team understand the RSB and its capabilities.
- Select a deployment option (cluster or non-cluster).
  - This may be mixed depending on the phases of deployment. Development and test may be non-clustered and production clustered.
  - Understand the operational complexities of each and plan for the staffing.
  - Work with the application server administration teams to determine the physical and logical placement of the RSB components.
  - Work with the system administrators to select the central RSB management location, rsb-home.

- The installation of the RSB has many prerequisites and dependencies that require the understanding, support and effort of database administrators, system administrators, application server administrators, and your organization's Oracle Retail application teams. It is a critical role of the RSB system administrator to work with each team, regardless of the site organization structure. See the Oracle Retail Service Backbone Installation Guide.
- Create operational plans for the RSB life cycle.
- Create plans for environment monitoring and maintenance.
- Plan for performance test. The RSB supplies tools to aid in the testing, but it is a difficult task that involves the database administrators, system administrators, application server administrators, and the Oracle Retail application teams.

Verification and Validation

Verification is the process of reviewing, inspecting, testing, and documenting that the product behaves in a manner as defined by the product requirement specification. Validation on the other hand is the process of making sure that the product's runtime behavior meets the retailer's needs and requirements. RSB provides tools and utilities to verify that a RIB installation is configured correctly and works properly when service invocations occur in your enterprise. RSB also provides tools to test integration infrastructure standalone, independent of any Oracle Retail applications.

Implementation Environment Verification

After the deployment, the implementation can be verified using RIC. RIC shows a variety of information including the diagrammatic representation of integration flows, deployment topology, server details, WSDL URLs. In addition to this, it supports verification of services by calling the ping operation. It can also test the database connectivity.

Testing the Web service calls using ping in RIC is an almost complete validation of your implementation. The ping call goes through the decorator to the edge app service and returns. Hence, if ping works, generally the implementation is correctly configured. However, ping test will not detect issues with payload version differences, certificates and keystores in the consumer side and so on.

15

RSB in Operation

This section contains common issues that need to be thought about and addressed by a retailer as they progress towards a production environment involving RSB. It is not a comprehensive list, nor does it seek to answer the questions, since they are very dependent on the retailer implementation. The intent of this section is to provide a starting point for a site-specific RSB operations planning effort.

Operational Considerations

RSB has built-in alerts and notification through JMX. An external system can subscribe to all of the built-ins. The JMX alerts are raised by the underlying OSB. Any JMX client adhering to JMX standard should be able to subscribe to the JMX alerts.

Log File Archive and Purge

RSB uses log4j for all of its logging control. It manages the logs size through its control file. In various phases of deployment and in triaging a problem it is often desirable or necessary to archive the logs so the logs are smaller and scanning by tools or people is easier.

The number of log files retained and size of the log files are fixed during RSB deployment. However, these configurations can be changed through the WebLogic Admin Console.

Message Flow in a Decorator Proxy Service

A message flow is associated with a proxy service. It shows various nodes in the flow of control from the proxy to business and back. Message flow provides a means to tap into the flow of control and perform any custom processing like transformation, routing etc. The flow can be modified using XSLT or XQuery as the primary programming language. The decorator already has a flow configured with calls to RSB instrumentation code. Do not modify this code, since there are a lot of RSB features dependent on the instrumentation.

Note: For more information, see the *Oracle Retail Service Backbone Developers Guide*.

Error Handling in Decorator Services

In RSB the decorator proxy services comes with a default OSB error handler at the service level. This error handler replaces the soap body with the error reason from the

edge app service and the operation name. Then the error handler calls the instrumentation code and returns a failure to the service consumer. OSB allows users to configure error handling at the message flow, pipeline, route node, and stage level.

Service and Business Alerts

Oracle Service Bus evaluates rules against its aggregated metrics each time it updates that data. When a rule evaluates to True, it raises an alert.

All the SLA alerts are shown in RIC as System Alerts and all pipeline alerts are shown as Business Alerts. You can also view the alerts through OSB console. RIC is retrieving the alerts from OSB using JMX. If the alerts are purged in OSB, they will also disappear in RIC.

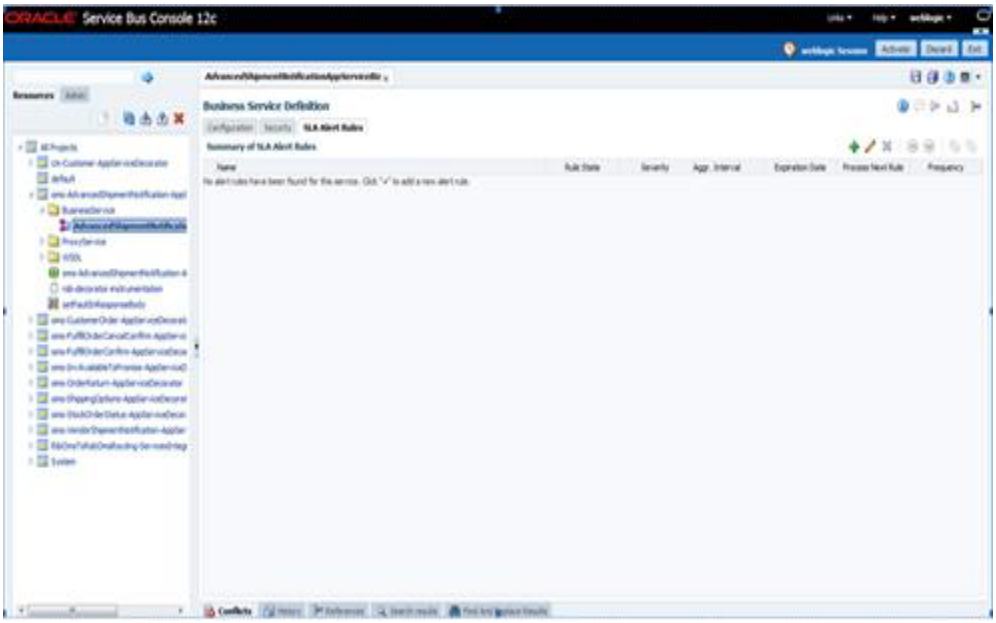
Users or applications can subscribe to the alerts using JMX clients or by configuring e-mail server in OSB. See RSB Developers Guide for instructions to configure OSB server for subscription of alerts through e-mail.

RSB comes with a default SLA alert configured. Users can change the rule or add new rule depending on the business requirements.

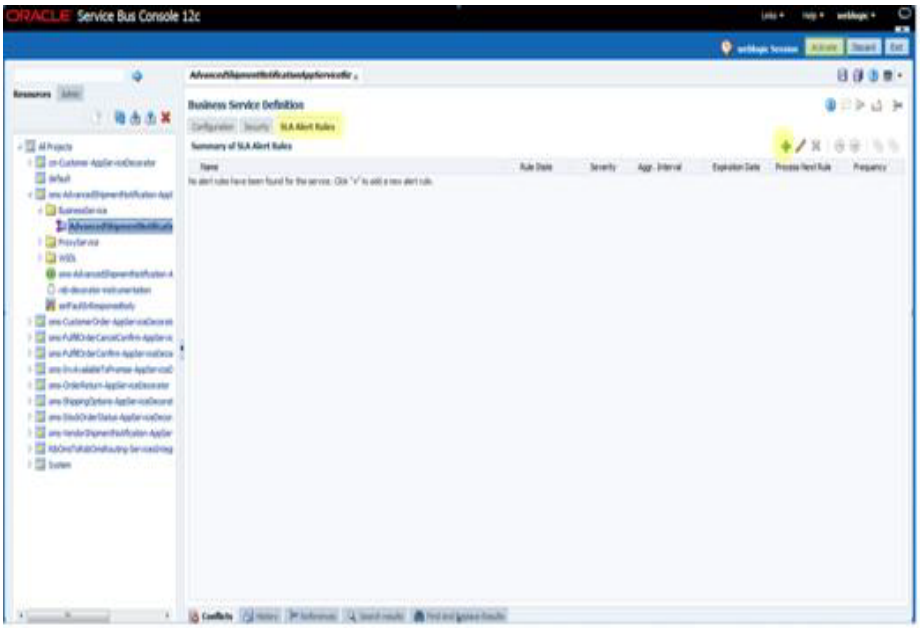
Adding New Alerts

Both Business (pipeline) and Service (SLA) alerts can be added either through OSB console or programmatically. If the alert is added through OSB console, the alert configuration will get overwritten when the OSB project is redeployed. So it is recommended to add only temporary alerts through OSB console. Any alerts that are permanently needed must be added programmatically. SLA alerts are added at the proxy or business services. Pipeline alerts are added to the message flow. Take the following steps to add an SLA alert to a business service:

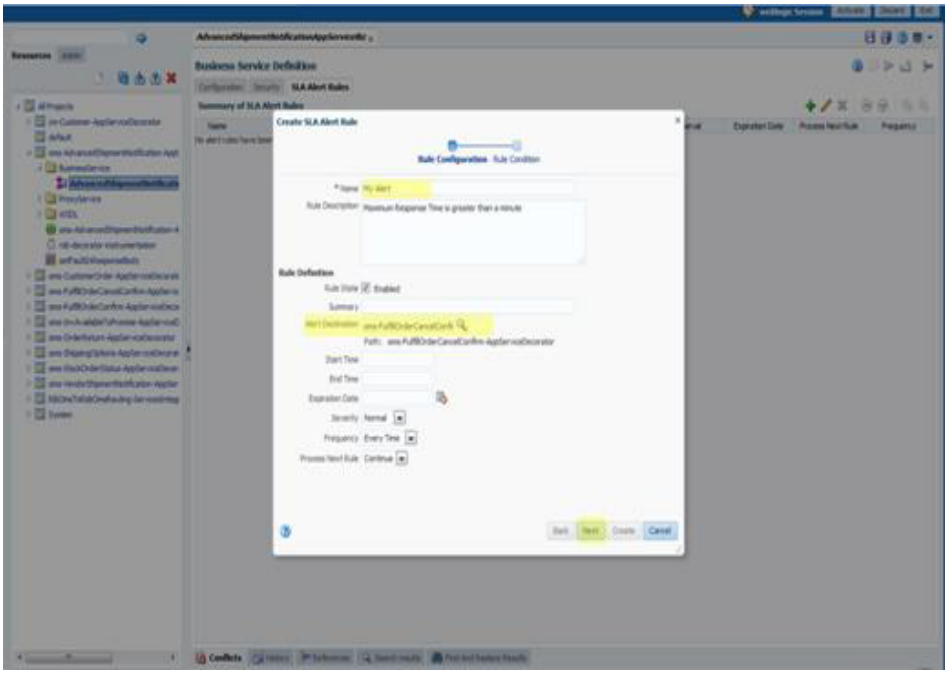
1. Click on the **Business Services** link in Resource Browser of OSB console.
2. Click the service for which you want to add the alert.



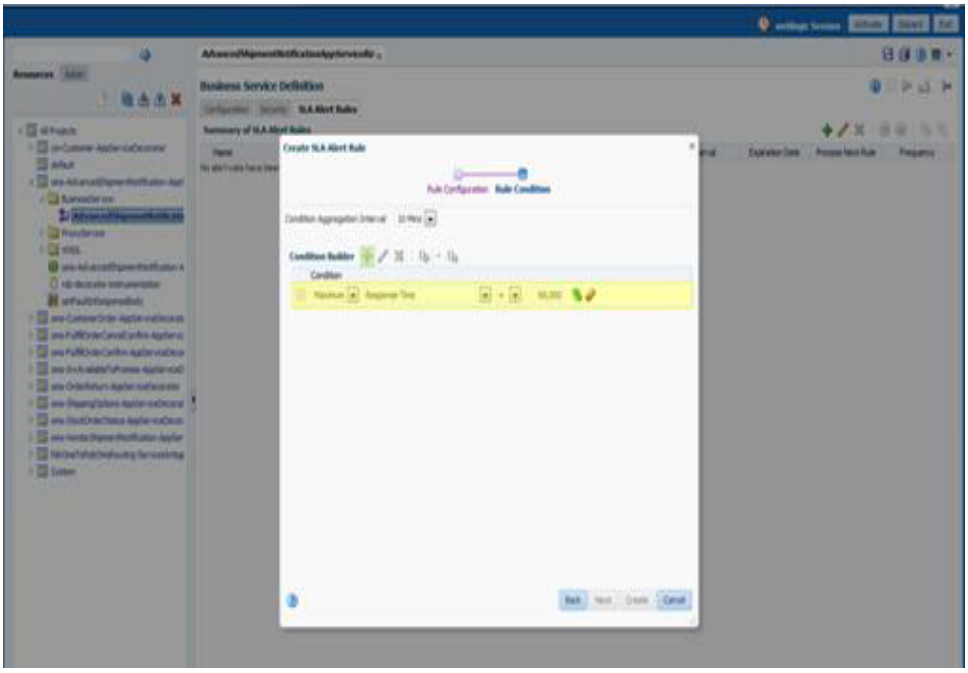
3. Select the **SLA Alert Rules** tab. Click **Add**.



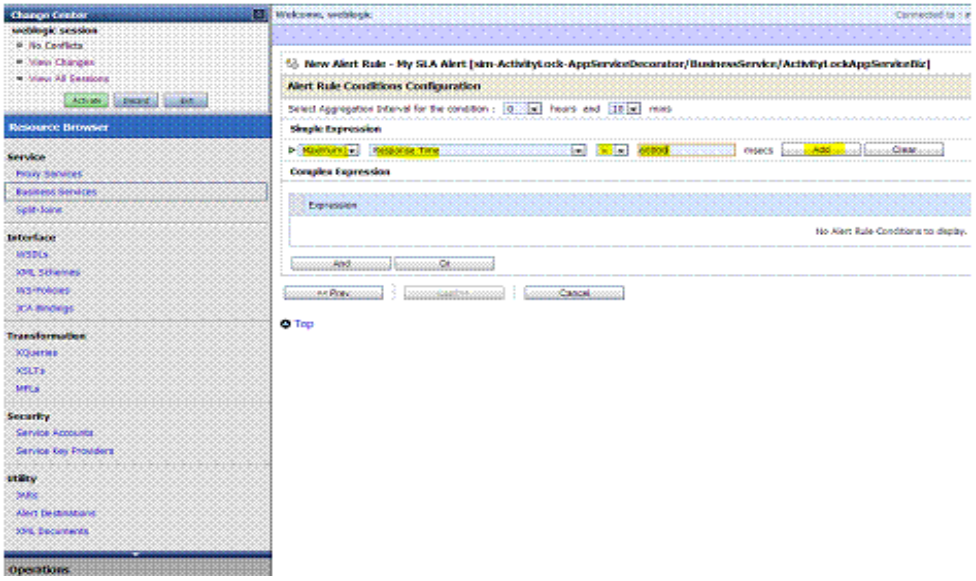
4. Enter **Rule Name** and **Alert Destination** corresponding to the decorator.



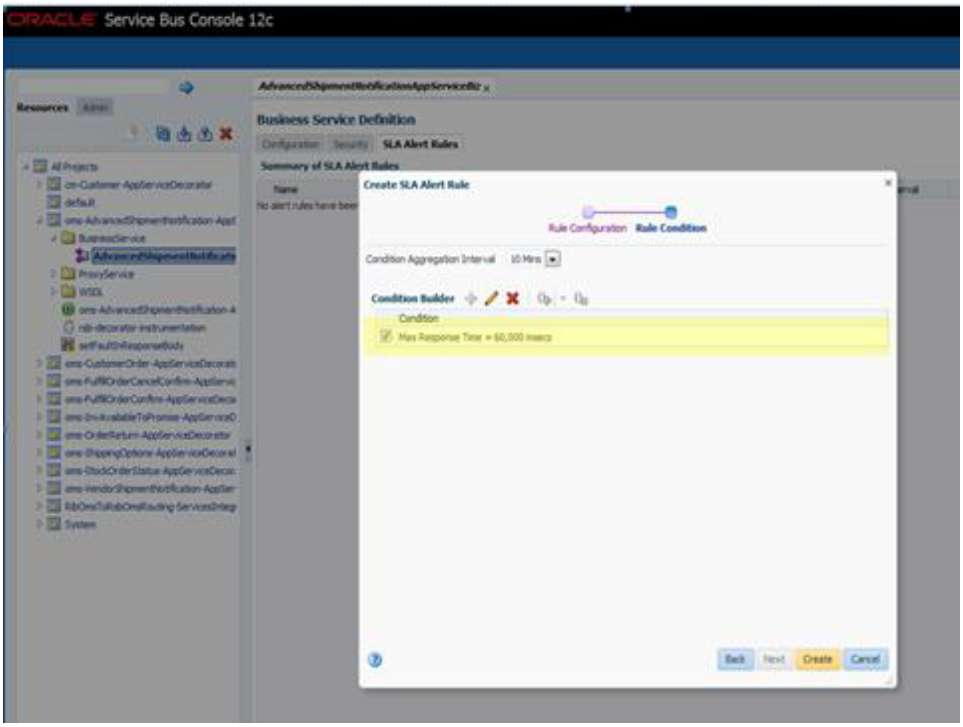
5. Construct the Alert Rule Condition as shown in the following screen:



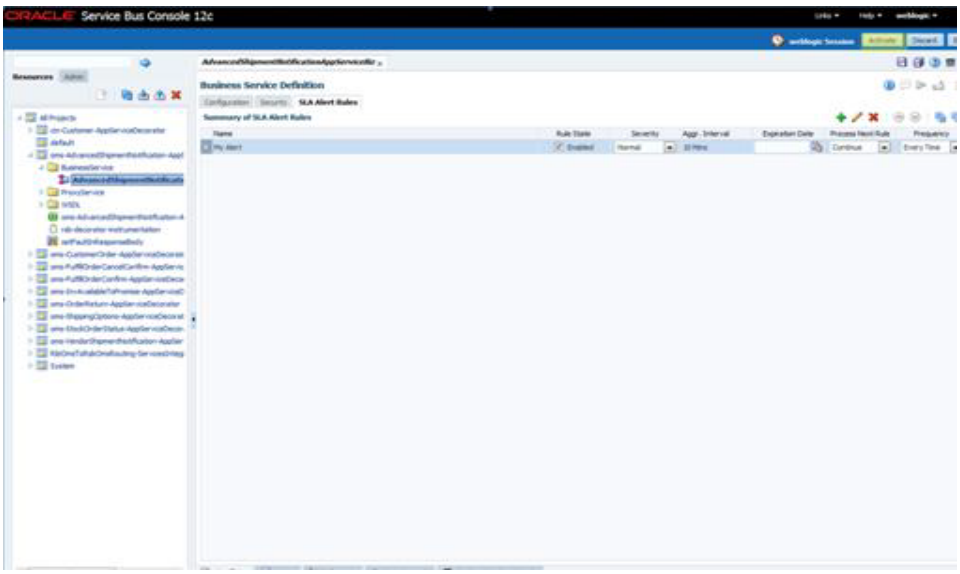
6. Save the alert and activate the session.



7. Click **Update** and then **Create**.



8. Activate the session.



Note: See also *Oracle Service Backbone Developers Guide*.

Performance Monitoring

The decorator services are designed for minimal impact on the service invocation. However, there can be many reasons for a slow-performing system. RSB provides helpful information to identify potential reasons for performance drop. RIC is very useful in narrowing down the components that may be causing the performance issues. RIC provides different ways to monitor the performance of service invocation.

In the RSB Integration Summary tab of RIC, you can view the CPU and Memory usage of RSB servers. So, if the RSB server is the bottleneck in the performance flow, it would be indicated by a high CPU usage and/or memory usage.

In the RSB Integration Summary tab, the applications are shown in the graph against the average response time. This graph helps in identifying applications that are slow. This potentially indicates slow application servers or network performance issue to slow performing application servers.

Performance tab under Performance and Diagnostics, you can query for slow performing services or high volume service. If the performance issue is at a service level, it can be identified here.

System Logs screen is another place to look for any log entry that would point towards the causes of performance degradation. RIC provides a consolidated view of logs from all the domain servers. Users can also filter to log entries by severity, timestamp, server and so on.

OSB Console

OSB console is the operational console for Oracle Service Bus. OSB console helps to perform many operational tasks of the OSB hosted applications.

- Manage Transport, SOAP binding and Message handling configurations
- Manage Monitoring, Throttling and Tracing configurations
- Manage SLA alert rules
- Manage Security Policies
- Export WSDLs
- Export Projects

Note: See also Oracle® Service Bus Guides.

Service Activity Table Growth and Purge

The service activity table stores the data collected by instrumentation. The services for which trace is enabled are intercepted and meta data is persisted in this table. If audit is enabled, the payload is also saved to the database. There will be one row per each service invocation.

RSB system will not remove any records from this table. The table will grow with each call of the instrumented service.

Each implementation should design their own purge strategy for the service activity table. This may depend on various factors like

- How many days of historical data have to be maintained.
- The average volume of transactions per day.
- How many services are audit enabled.

If the table is not purged, the data will grow indefinitely.

Support Staff Requirements

RSB requires OSB as the infrastructure component. This is one of the main differences between RSB and other Oracle Retail applications. Thus, familiarity of OSB is an additional skill requirement for support staff of RSB.

16

Administration and Logging

This chapter describes the RSB's administration and logging options.

RSB Deployment Properties

RSB has following configuration properties to customize the RSB installation to your environment. After any configuration changes, the rsb-compiler must be run and then decorators should be deployed to the server.

JAVA_HOME

This variable must point to the JDK home folder. This is a mandatory property.

For example

```
JAVA_HOME=/u00/java/jdk1.8.0_65.64bit
```

rsb-deployment-env-info.service-provider-app-in-scope-for-integration

This property holds the list of applications that act as service providers. The list is a comma separated list of the short app names.

For example,

```
rsb-deployment-env-info.service-provider-app-in-scope-for-integration=rms,igs,sim,
cm, ooc, mms, rwms, fin, rpm, rm, co, rob, rce
```

rsb-deployment-env-info.service-requester-app-in-scope-for-integration

This property holds the list of applications that act as service consumers. The list is a comma separated list of the short app names.

For example,

```
rsb-deployment-env-info.service-requester-app-in-scope-for-integration=rms, rwms,
sim, rpm, mms, reim, pos, ooc, fin
```

rsb-osb-container.domain-name

This property holds the name of the WebLogic domain where the RSB is deployed.

For example,

```
rsb-osb-container.domain-name=rsb_domain
```

rsb-osb-container.<Domain Name>.home

This property holds the WebLogic domain home directory of the RSB installation. Replace the <Domain Name> with actual domain name.

For example,

```
rsb-osb-container.rsb_domain.home=/u00/rsb/Oracle/Middleware/user_
projects/domains/rsb_domain
```

rsb-osb-container.<Domain Name>.cluster-name

This property holds the name of the cluster used when configuring the domain.

For example,

```
rsb-osb-container.rsb_domain.cluster-name=rsb_cluster
```

rsb-osb-container.<Domain Name>.<Cluster Name>.http-url

This property holds the URL for the HTTP Proxy server of the cluster.

For example,

```
rsb-osb-container.rsb_domain.rsb_cluster.http-url=http://rsbhost.example.com:19707
```

rsb-osb-container.<Domain Name>.<Cluster Name>.https-url

This property holds the https URL for the HTTP Proxy server of the cluster. This property is needed only if the services are secured with https policies (like PolicyA).

For example,

```
rsb-osb-container.rsb_domain.rsb_cluster.https-url=https://rsbhost.example:19708
```

rsb-osb-container<Domain Name>.admin-server-http-url

This property holds the URL for the Admin server of the cluster

For example,

```
rsb-osb-container.rsb_domain.admin-server-http-url=http://rsbhost.example:19701
```

rsb-osb-container.<Domain Name>.admin-server-connection-url

This property holds the connection URL for the Admin server of the cluster.

For example,

```
rsb-osb-container.rsb_domain.admin-server-connection-url=t3://rsbhost:19701
```

rsb-osb-container.<Domain Name>.<Cluster Name>.managed-servers

This property holds the names of managed servers in the cluster. The names are comma separated.

For example,

```
rsb-osb-container.rsb_domain.rsb_cluster.managed-servers=rsb_server1,rsb_server2
```

rsb-osb-container.<Domain Name>.<Cluster Name>.<Managed Server>.managed-server-connection-url

This property holds the connection URL for each of the managed servers in the cluster. There should be one line item for each of the managed servers.

For example,

```
rsb-osb-container.rsb_domain.rsb_cluster.rsb_
server1.managed-server-connection-url=http://rsbhost.example.com:19703

rsb-osb-container.rsb_domain.rsb_cluster.rsb_
server2.managed-server-connection-url=http://rsbhost.example.com:19705
```

service-infrastructure-db.jdbc-url

This property holds the JDBC URL for the schema where the RSB_SERVICE_ ACTIVITY table resides.

For example,

```
service-infrastructure-db.jdbc-url=jdbc:oracle:thin:@rsbhost.example.com:1521:orcl
```

edge-app-container.<App>.connection-url

This property holds the connection URL for the edge application servers where the services are hosted. There must be one line item for each of the applications in scope.

For example,

```
edge-app-container.rms.connection-url=t3://rsbhost.example.com:19720

edge-app-container.sim.connection-url=t3://rsbhost.example.com:19721
```

global.app-service-end-point-url-pattern=http://<HTTP_HOSTNAME>:<HTTP_ PORT>/<SERVICE_NAME>Bean/<SERVICE_NAME>Service

This property holds the URL pattern of the edge application web services. This pattern is used to generate the actual URLs of the services. Do not replace the entries in upper case with any values. This hard coded text, which gets replaced by the values taken from other property by the compiler. You should not change that text. You can modify the pattern (except the uppercase entries) to match the service URL. The pattern given above works for WebLogic hosted service. In that case, you do not have to change anything.

If the service is hosted in Glassfish, the pattern would be http://<HTTP_ HOSTNAME>:<HTTP_PORT>/<SERVICE_NAME>Service/<SERVICE_ NAME>Bean.

<App>.app-service-end-point-url-pattern=http://<HTTP_HOSTNAME>:<HTTP_ PORT>/<SERVICE_NAME>Bean/<SERVICE_NAME>Service

This property holds the application specific URL pattern for the web services. This is an optional property and you need to specify only when the application service URLs do not match the global pattern. If application pattern is present, the global pattern is ignored for that application.

For example,

```
oms.app-service-end-point-url-pattern=https://<HTTP_HOSTNAME>:<HTTP_
```

```
PORT>/<SERVICE_NAME>Bean/<SERVICE_NAME>Service
```

<Decorator>.app-service-end-point-url=<Service URL>

This property holds the decorator level URL pattern. This is an optional property and need to be specified only if the service URL do not match the application pattern or global pattern. This is the lowest granular level of control for the URL pattern. This entry need not be a pattern; you can provide the actual URL instead.

If a decorator level property is present, the app level and global level properties are ignored for that decorator. You can give any number of decorator level properties.

The global, app and decorator level properties give the complete control to specify any type of web service URLs.

For example,

```
rms-Supplier-AppServiceDecorator.app-service-end-point-url=https://rsbhost.example
.com:19721/SupplierBean/SupplierService
```

Note: Use service level pattern, if there are only few services with a different pattern. If most of the services in an application follow a pattern, use application level pattern and service level pattern for the exceptions.

Enabling and Disabling Instrumentation

There are a few properties in rsb-decorator-instrumentation.properties that determines the instrumentation. The services can be instrumented at the global level or service level. Two factors that affect this configuration are business requirements and performance. The tracing and audit are designed to persist the information asynchronously. It should not affect the main flow of execution other than just an asynchronous call to the data persistence code. However, if all the services are instrumented and there is very high transaction volume, performance of the server is affected. In those cases, the instrumentation is enabled based on service.

There are two levels of instrumentation: Trace and Audit. Tracing is the flag to enable instrumentation. Audit flag decides whether payload is also persisted along with other instrumentation data. Persisting the payload will have an impact on the size of the database since size of the payload bigger than the combined sizes of all other instrumented data. So the decision to audit should consider the overhead it has.

Tracing for all the service level is decided by the following property:

```
global.instrumentation.tracing.enabled
```

Audit for all the service level is decided by the following property:

```
global.instrumentation.auditing.enabled
```

Auditing and Tracing for a specific service is enabled by the properties <App>-<ServiceName>-AppServiceDecorator.instrumentation.auditing.enabled and <App>-<ServiceName>-AppServiceDecorator.instrumentation.tracing.enabled where <ServiceName> is the name of the service and <App> is the name of the edge app.

Examples:

```
rms-PayTerm-AppServiceDecorator.instrumentation.auditing.enabled=true
rms-PayTerm-AppServiceDecorator.instrumentation.tracing.enabled=true
```

Tracing and Auditing can be enabled while RSB is up. The server polls the property files at the specified interval and the new changes are in effect from that moment. The

default interval is 10 minutes. If this interval does not suit the requirement, it can be changed by specifying new interval (in minutes) for the property:

instrumentation.property.reload.interval
Following steps show how to change the trace and audit properties:

- Make changes to desired trace and audit properties in rsb-decorator-instrumentation.properties file.
- Compile RSB using rsb-compiler.sh.
- Deploy any decorator (just one decorator is enough). This will deploy the jar file containing the property to the server.
- The changes will take effect in utmost 10 minutes (The default interval for the background task that runs to pick up the changes).

Changing Passwords

RSB uses many passwords during the installation. These usernames and passwords are stored in an encrypted credential wallet file under predetermined aliases.

- Service Activity Schema (sidb-jdbc-user-alias)
- WebLogic Server Admin (admin-server-user-alias)

If you are using Policy B for securing the services, there are additional credentials captured at compile time. If any of the passwords need to be changed, run the script setup-message-protection-security-credentials.sh to update the password in the credential wallet file. After updating the wallet file, run the script configure-rsb-app-server-for-security-policy-b.sh to update WebLogic server with new password(s) for policy B.

If any of the passwords need to be changed for any reason, follow the steps listed below.

- Change the password of the target system (Database, WebLogic).
- Run the rsb-compiler with -setup-security-credential option and update with new password.
- Run rsb-deployer.sh with -prepare-wls option.
- Deploy decorators and service integration flows.

Trace and Audit Logs

The trace and audit information is maintained in the RSB_SERVICE_ACTIVITY table. The contents of the table can be viewed through RIC. The Trace and Audit Log page in RIC allows users to filter the content by server by app name, message family, service name, time and time interval.

Broadcasting Logs

In RSB, the decorators are deployed to OSB cluster. When a consumer invokes a service, the invocation can happen in any of the managed servers of the cluster, depending on the load balancing algorithm. Each of the managed servers has its own log file. So the log information for the service execution will go to the log file of the managed server where the service got executed. This poses a hurdle when troubleshooting. RSB solves this issue by having the managed servers broadcast the

log to the admin server. Thus the logs from all managed servers are made available in a single location.

Each of the managed server is enabled for log broadcasting. A global log filter (RSBLogFilter) is selected as the filter for selecting the log entries to be sent to the Admin Server. The default setting for this log filter is to send all the log entries. This is done using a log filter expression: "(SERVER LIKE '%')". The log filter expression may be modified, if only specific types of logs need to be broadcasted. For example, if the system requirement is to broadcast only message of certain log level, then the log filter expression needs to be modified to filter the entries accordingly.

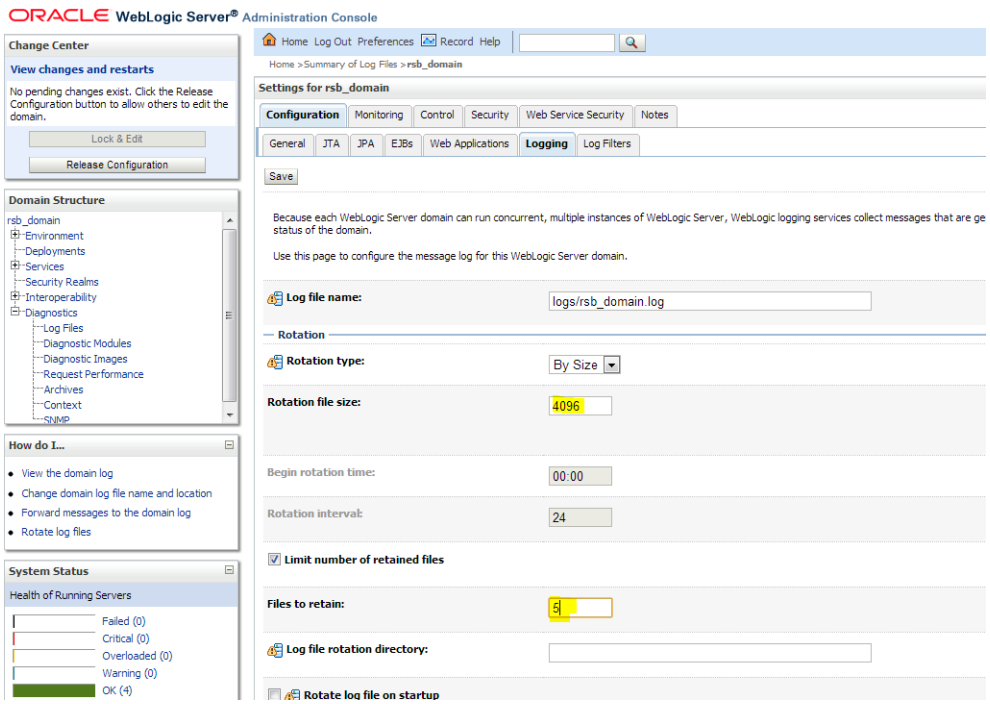
The users can view the consolidated log on its location or via the RIC application.

Log File Archive and Purge

The default log file size or number of logs may not be adequate in a production environment. Every customer must choose the file size and number of files suitable for their environment. However, larger log file sizes may impact the performance of Retail Integration Console (RIC) pages that show log information.

To change the default size of the log file and the number of log files through WebLogic Admin console.

1. Click the domain name (rsb_domain).
2. Click the **Logging** tab to reach the screen.
3. Click **Lock & Edit**, if necessary.
4. Click **Save** and then **Release Configuration**, if needed.



17

Security

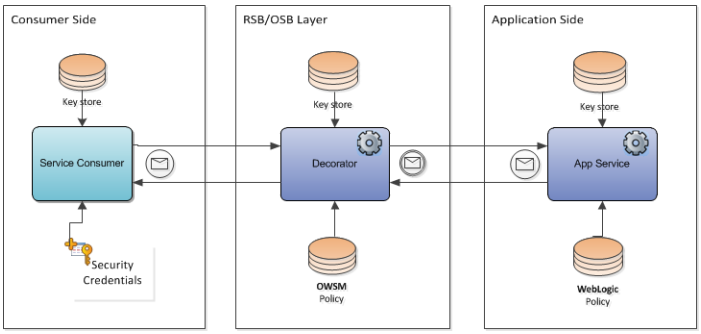
Security in the integration layer is a big concern for every retail enterprise. The security system should be open enough to allow trusted remote applications to integrate easily and, at the same time, lock down unauthorized remote access. To address security concerns, RSB utilizes a standard policy based security model supported by WebLogic and OSB. The security policy and the implementation details are described in the RSB Security Guide.

Web Service Security Policies

The RSB decorator services, edge app Web services and consumers can be secured to in many ways. The preferred method of security is by using web service security policies. There are many security policies available in OSB and edge app servers. Additionally, you can write custom security policies using security policy authoring tools.

The choice of security policy for a layer depends on the security policies available in other layers. The policies have to be compatible in order for the system to work. RSB supports two sets of such compatible policies out of the box. These are called Policy A and Policy B. While there is no restriction on RSB side to use any other policy, there is no built-in support for any other policies other than the two sets mentioned before.

In both Policy A and Policy B, OWSM is the security provider in RSB/OSB layer and WebLogic is the security provider in the edge app layer. Policy A is basically SSL plus Username Token and Policy B is Message Protection plus Username Token.



Note: Review the *Oracle Retail Service Backbone Security Guide*.

18

Testing the Retail Service Backbone

The Oracle Retail Service Backbone is difficult to test as a stand-alone subsystem. It is part infrastructure and part application, and needs to have the integrating application end-points for even a simple installation.

RSB provides a testing framework to test and validate the RSB installation. This is called Service Interface Testing (SIT) tool.

SIT

There are two types of test framework tools: PSIT and JSIT. PSIT is for PLSQL applications and JSIT is for Java applications. This application can be used to represent any of the Oracle Retail applications. SIT Tools are able to provide response to web service consumers. The responses can be customized for the request to enable different testing scenarios.

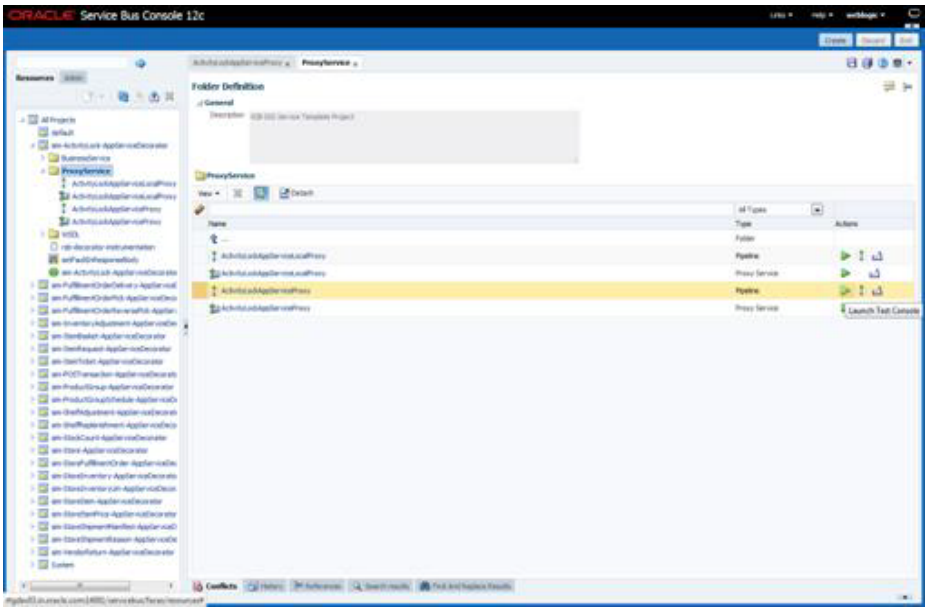
SIT tools enable customers to conduct RSB testing without waiting for the participating Retail application implemented. When the Retail application is implemented, the change may be as simple as changing the URL for the application.

SIT tools are not tools for testing, but application that helps testing by providing the missing pieces of integration.

To test a decorator service there are different options like SB Console, third-party applications like SoapUI, RIC, standalone programs and so on.

SB Console

SB Console has a test option for proxy services. In order to test a decorator, select the proxy service of the decorator service. Click the **Test Launch Console** icon (play icon) Select the operation to test and provide the input. The results of the test are displayed as XML in the response window.



SB Console can test unsecured services and services using Policy B. It cannot test https based service (thus eliminating Policy A services).

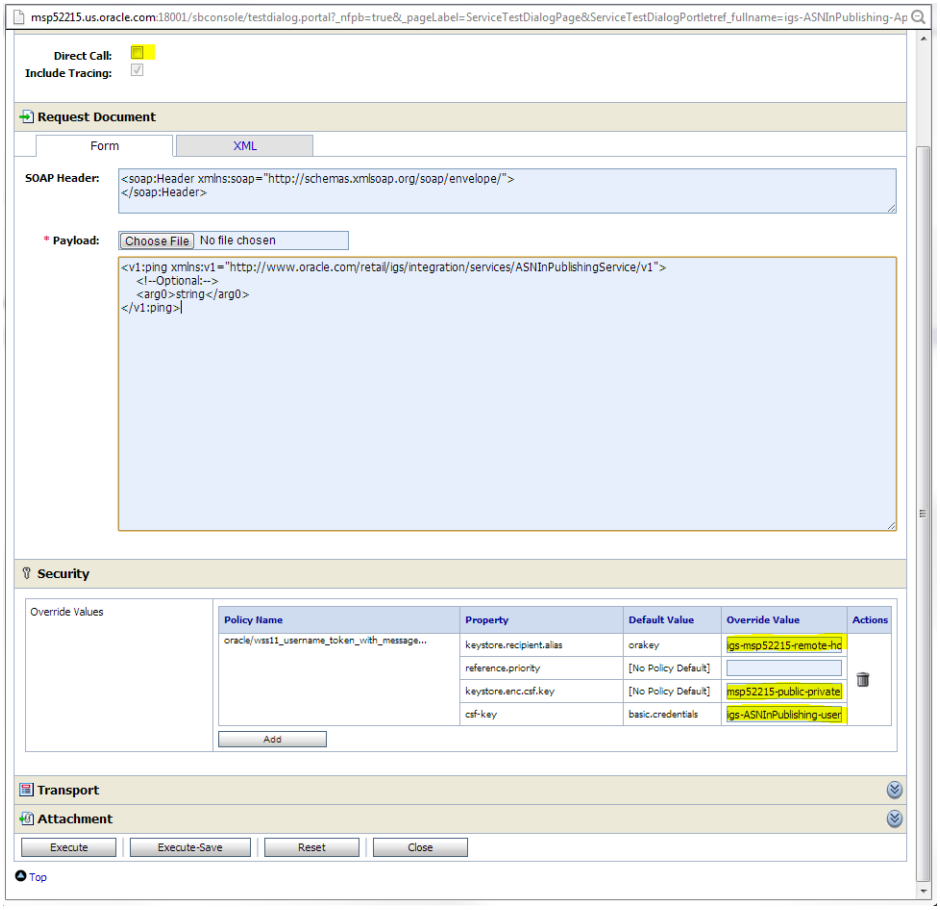
Policy B Testing

Click the proxy service and then **Test Launch Console** icon to test the service. You can invoke any method. Ping method is the easiest to test. If the invocation is successful, it shows both encrypted and decrypted versions of request and response messages.

In order to test proxy service from test page, you need to provide values for following fields which are at the bottom of the test page:

- keystore.recipient.alias
- keystore.enc.csf.key
- csf-key

You can find the values for these fields from the Business Services > [Business Service] > Security tab of the same decorator project. You will see these values pre-populated in that page. Copy these values and enter on the proxy service test page for successful invocation of a service.

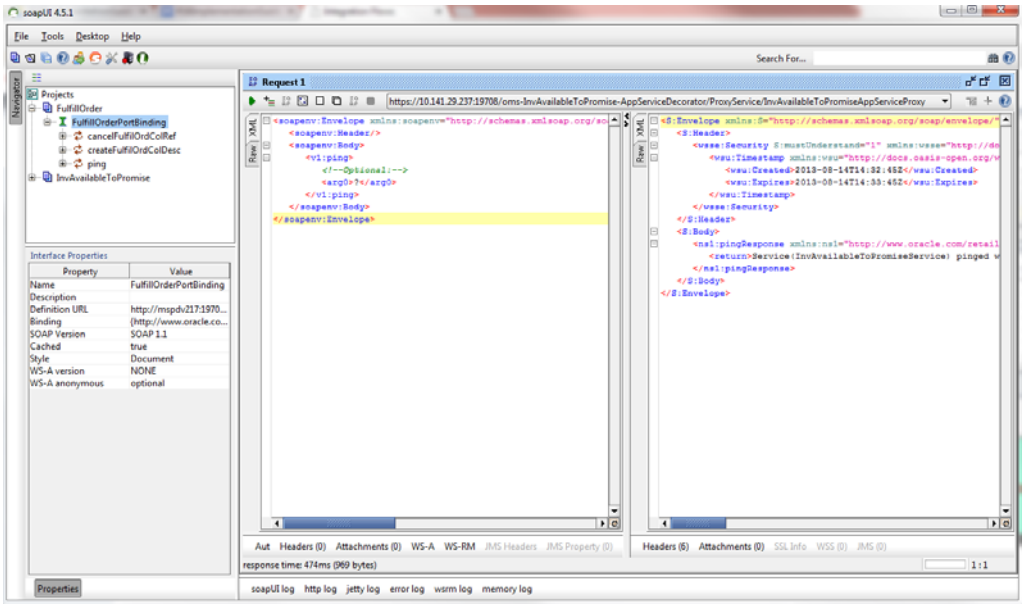


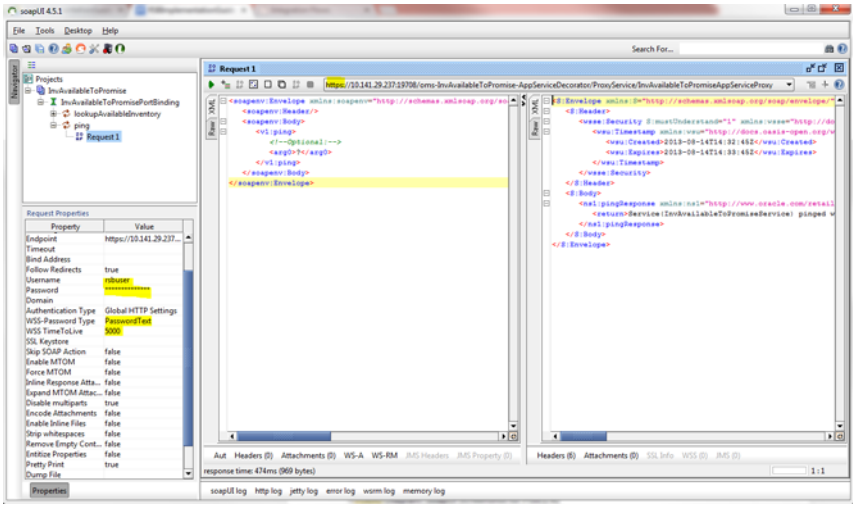
Third-party tools (SoapUI)

SoapUI is an Open Source Functional Testing tool for Web Services.

See http://www.soapui.org for more details.

SoapUI can be used to test decorator services. Unsecured services and services secured with Policy A can be tested using SoapUI. Currently there is no easy way to test services secured with Policy B.





RIC

RIC is another application that can validate the Retail services. In the Health Check tab under Performance And Diagnostics tab the services are listed and the ping operation can be invoked. RIC can test ping operation of unsecured services and services that are secured with Policy A and Policy B. RIC cannot test any other operations other than ping. RIC is an easy way to validate the service connectivity between the Retail applications. However, even if the ping operation is successful there are still scenarios like the ones listed below that can prevent a successful service invocation.

- The payload mismatch between service consumer and service provider (because of different versions of artifacts)
- Incorrect certificate (for secured services) in service consumer side
- Network connectivity between service consumer and service provider machines
- Un-synchronized machine time between service consumer and service provider
- Invalid credentials passed from the service consumer.

19

Performance

This chapter discusses the performance characteristics of RSB, the factors that affect it, and a process to test it.

Performance of RSB within a customer site is critical to the performance of the business, and is determined by factors specific to a given deployment. Because of this is, a Performance Test is recommended as part of every deployment plan. Even if formal testing is not planned, the use of the tools and processes discussed can measure the relative performance of the RSB sub-system and can be used to diagnose bottlenecks.

It is beyond the scope of this document to discuss all of the tools and techniques available at the host, network, database, and application server level.

Performance Factors

The performance of each of these components is influential in the overall performance of the system:

- The application server(s) topology and configuration.
- The RSB deployment approach.
- The hardware sizing and configuration of the RSB hosts.
- The hardware sizing and configuration of the applications that are connected to the RSB.

There are other factors that determine the performance of the overall system. Some of these factors in a RSB environment are:

- Number of managed servers in the cluster
- Number of service calls
- Size of the payload
- Database clustering

Cluster Deployment

RSB needs to be installed in a cluster. A cluster configuration allows for horizontal scaling. Managed servers can be added or removed depending on the volume of transactions and performance requirement. Also, customers can configure different load balancing algorithm to suit their needs.

Enabling/Disabling Instrumentation

If the performance drop is due to heavy instrumentation, the instrumentation can be turned off or set selectively to a minimum level. Also, unnecessary auditing can be turned off for services to improve performance.

Purging

Some of the data gets accumulated over time. Purging or archiving unnecessary data helps system performance.

RSB Service Activity Table

Service Activity table adds a row for each invocation of the service for which trace is enabled. If audit is enabled, the payload is also persisted into the database. Depending on the business need, unnecessary growth of this table should be controlled by regular purging.

RSB Service Activity is a high volume, denormalized table and is expected to grow into a significant size, measures should be taken to manage and monitor the table on frequent basis depending on the consumer's Data lifecycle management policies. Un-monitored, undamaged RSB Service activity table overtime can lead to significant issues not limited to slow system response, database instance issue, database crash and others.

Some of the recommendations are:

- Partitioning strategy: Date Partitioning on the table (one of the partition schemes).
- Data Purging. Purging of RSB SERVICE ACTIVITY should be done at the partition level. Dropping or truncating a partition would perform far better than a delete statement. Global indexes will have to be maintained/updated as well;
- Collection of statistics on a frequent basis.
- Data Archiving.
- Parallelization of queries.

How to Calculate Schema Size?

The service activity table can grow fast depending on the number of transaction and trace and audit configuration. The growth rate and size requirement varies by the type of retail business. It is recommended to estimate the size requirement for your enterprise in the beginning.

Size required (MB) = number of days of data retention * average number of transactions per day * average record size (MB)

You may use the following SQL to calculate the average size of a row, if you have sufficient data in the service activity table.

```
select
 avg(nvl(vsize(activity_num),0) +
     nvl(vsize(activity_state),0) +
     nvl(vsize(application_name),0) +
     nvl(vsize(business_operation_name),0) +
     nvl(vsize(BUSINESS_SERVICE_NAME),0) +
     nvl(vsize(BUSINESS_SERVICE_URI),0) +
     nvl(vsize(ERROR_CODE),0) +
     nvl(vsize(ERROR_DETAIL),0) +
```

```
     nvl(vsize(ERROR_REASON),0) +
     nvl(vsize(MESSAGE_ECID),0) +
     nvl(vsize(MESSAGE_FAMILY),0) +
     nvl(vsize(PROXY_OPERATION_NAME),0) +
     nvl(vsize(PROXY_SERVICE_NAME),0) +
     nvl(vsize(PROXY_SERVICE_URI),0) +
     nvl(vsize(REQUEST_TIMESTAMP),0) +
     nvl(dbms_lob.getlength(REQUEST_XML),0) +
     nvl(vsize(RESPONSE_TIMESTAMP),0) +
     nvl(dbms_lob.getlength(RESPONSE_XML),0)
     ) "Average Size(Bytes)"
from
    rsb_soainfra.rsb_service_activity
```

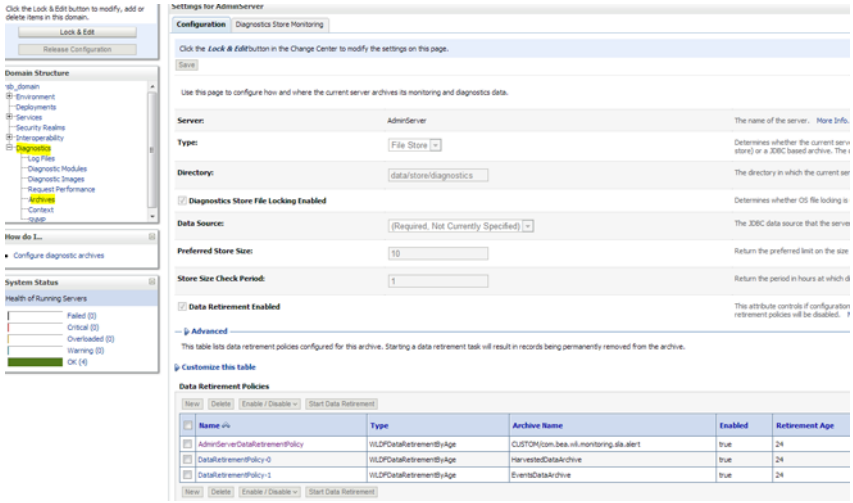
WebLogic Data Retirement Policy

WebLogic data retirement policy may have impact on the performance of server. If the RSB system has a lot of alerts and statistical information accumulated over time, the old data needs to be retired to keep the performance optimal. RSB ships with a default data retirement policy. This is disabled by default.

The default data retirement policy is set with a Retirement Age of 72, Retirement Time of 1 and Retirement Period 12. This means retire data 72 hours or older. Start the first run from 1:00 am and repeat the process every 12 hours after that.

If the customer wants to use this policy, they can enable the policy through WebLogic Admin Console. Following steps show how to enable the data retirement policy.

1. Click **Diagnostics**.
2. Click **Archives**.
3. Click **AdminServer**.
4. Click **Lock & Edit**, if needed.
5. Select the policies **BusinessAlertDataRetirePolicy** and **SLAAlertDataRetirePolicy**.
6. Click b.
7. Click **Release Configuration**, if needed.



20

Troubleshooting

RIC can be used to assist both technical and business troubleshooting. Here are some of the features that can help in analysis of issues.

- Health check (WSDL accessibility). The ping operation can be used to check if there are any connectivity issues in calling services. This page allows testing database connectivity too.
- Log files. RIC shows a consolidated log from all the managed servers. You can filter the log based on severity, timestamp and server to analyze the log.
- Historical trend. Historical trend graphs show the trend of service invocation in the system. These graphs can be used to identify spikes and dips in service invocation counts to see any pattern that can indicate potential issues or risks. There are various levels of aggregation by app, message family, service and so on.
- Recent Activity in RSB Integration Summary shows the recent transactions. This can be used to check if an invocation went through the system.
- Top Problematic Interface in RSB Integration Summary page shows the interfaces that generate most faults.
- Trace logs show the log of transactions for services enabled for trace.

21

External Systems Integration

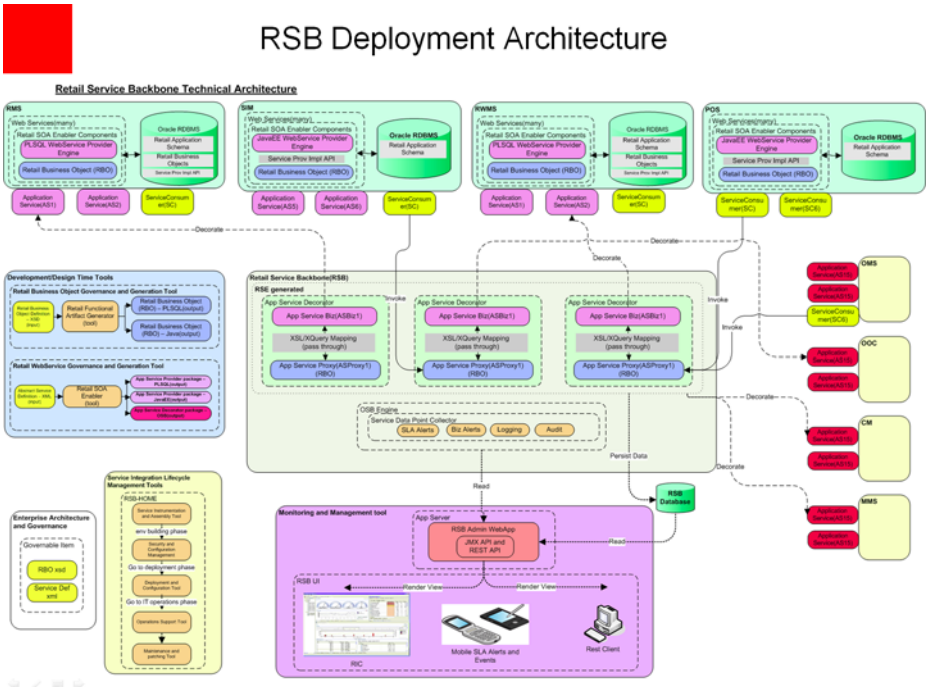
Enterprise Manager

Enterprise Manager for Fusion Middleware Control (EM) is useful in managing web service security policies, keystores and credential wallets, in addition to domain and server management features it provides.

EM is an optional component for RSB. It is installed during the domain creation. If installed, it allows for managing keystore and wallet files (especially for Policy B), managing OWSM policies and authoring web service policies.

RSB Architecture

The following diagram shows a deployment architecture diagram of a typical RSB implementation.



A

List of RSB Decorator Paks

- RsbAppServiceDecoratorPak19.0.000ForRm19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForCm19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForRms19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForFin19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForRpm19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForIgs19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForRwms19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForSim19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForCo19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForRce19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForRob19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForOoc19.0.000_eng_ga.zip
- RsbAppServiceDecoratorPak19.0.000ForMms19.0.000_eng_ga.zip

B

Service Integration Flows

RsbServiceIntegrationFlowPak19.0.000ForRibOmsToRsbOmsRouting_eng_ga.zip

C

RSB Security Configuration Sample Script

RsbAppServiceSecuritySetupSamplesPak19.0.000ForAll19.0.000Apps_eng_ga.zip

D

RSB_SERVICE_ACTIVITY Table

Create Table

The recommended partitioning strategy is interval partitioning based on the request_ timestamp column, with a single partition per day.

```
CREATE TABLE RSB_SERVICE_ACTIVITY
(
  ACTIVITY_NUM          NUMBER(19)       NOT NULL,
  ACTIVITY_STATE        VARCHAR2(8),
      APPLICATION_NAME          VARCHAR2(8),
  BUSINESS_OPERATION_NAME  VARCHAR2(256),
  BUSINESS_SERVICE_NAMEVARCHAR2(256),
  BUSINESS_SERVICE_URI VARCHAR2(256),
  ERROR_CODE            VARCHAR2(64),
 ERROR_DETAIL          VARCHAR2(2048),
 ERROR_REASON          VARCHAR2(1024),
 MESSAGE_ECID          VARCHAR2(64)NOT NULL,
 MESSAGE_FAMILY        VARCHAR2(25),
 PROXY_OPERATION_NAME VARCHAR2(64),
 PROXY_SERVICE_NAME    VARCHAR2(256),
 PROXY_SERVICE_URI     VARCHAR2(256),
 REQUEST_TIMESTAMP     TIMESTAMP(6),
 REQUEST_XML           CLOB,
 RESPONSE_TIMESTAMP    TIMESTAMP(6),
 RESPONSE_XML          CLOB
 )
```

Indexes

Indexes were added to assist in obtaining optimal performance of the active, real-time console. Six indexes were added, naming of indexes should be updated from what is listed below. Recommended indexes are as follows. All indexes are LOCAL partitioned indexes.

```
CREATE INDEX RSB_SERVICE_ACTIVITY _I1 ON RSB_SERVICE_ACTIVITY
(REQUEST_TIMESTAMP, APPLICATION_NAME, MESSAGE_FAMILY, PROXY_SERVICE_NAME)
LOGGING
TABLESPACE RETAIL_INDEX
LOCAL
INITRANS   12;

CREATE INDEX RSB_SERVICE_ACT_TEST_I3 ON RSB_SERVICE_ACTIVITY
(REQUEST_TIMESTAMP, APPLICATION_NAME, ACTIVITY_STATE)
```

```
TABLESPACE RETAIL_INDEX
INITRANS   12
LOGGING
LOCAL;

CREATE INDEX RSB_SERVICE_ACTIVITY_I4 ON RSB_SERVICE_ACTIVITY
(REQUEST_TIMESTAMP, PROXY_SERVICE_NAME, ACTIVITY_STATE)
  TABLESPACE RETAIL_INDEX
  INITRANS   12
LOGGING
LOCAL;

CREATE INDEX RSB_SERVICE_ACTIVITY_I5 ON RSB_SERVICE_ACTIVITY
(REQUEST_TIMESTAMP, ACTIVITY_STATE)
  TABLESPACE RETAIL_INDEX
  INITRANS   12
LOGGING
LOCAL;

CREATE INDEX RSB_SERVICE_ACTIVITY_I7 ON RSB_SERVICE_ACTIVITY
(RESPONSE_TIMESTAMP, PROXY_SERVICE_NAME)
  TABLESPACE RETAIL_INDEX
  INITRANS   12
LOCAL
LOGGING;

CREATE INDEX RSB_SERVICE_ACTIVITY_I8 ON RSB_SERVICE_ACTIVITY
(REQUEST_TIMESTAMP, RESPONSE_TIMESTAMP, APPLICATION_NAME)
  TABLESPACE RETAIL_INDEX
  INITRANS   12
  LOCAL
LOGGING;
```

Constraints

```
ALTER TABLE RSB_SERVICE_ACTIVITY ADD CONSTRAINT PK_RSB_SERVICE_ACT_TEST PRIMARY
KEY
(ACTIVITY_NUM)
USING INDEX
  TABLESPACE RETAIL_INDEX
  INITRANS   12;
```

Purging / Maintenance

Purging of this table should be done at the partition level. Dropping or truncating a partition would perform far better than a delete statement. Global indexes must be maintained/updated as well; there is only 1 global index at this time and that is on the primary key of the table. Updating of global indexes could be expensive with the potential size of this table. It will be a good idea to limit the number of global indexes used on this table.

References

OSB

- Oracle® Fusion Middleware Concepts and Architecture for Oracle Service Bus 12c Release 2 (12.2.1.3.0) https://docs.oracle.com/middleware/12213/osb/develop/
- Oracle® Fusion Middleware Administrator's Guide for Oracle Service Bus 12c Release 2 (12.2.1.3.0)

  https://docs.oracle.com/middleware/12213/osb/administer/

WebLogic

- Oracle® Fusion Middleware Using Clusters for Oracle WebLogic Server 12c Release 2 (12.2.1.3.0)

  https://docs.oracle.com/middleware/12213/wls/index.html
- Oracle® Fusion Middleware Introducing Web Services 12c Release 2 (12.2.1.3.0)

  https://docs.oracle.com/middleware/12213/owsm/
- Oracle® Fusion Middleware Performance and Tuning for Oracle WebLogic Server 12c Release 2 (12.2.1.3.0)

  https://docs.oracle.com/middleware/12213/wls/WLCAG/weblogic_ca_intro.htm#WLCAG107

OWSM

- Oracle® Fusion Middleware Security and Administrator's Guide for Web Services 12c Release 2 (12.2.1.3.0)

  https://docs.oracle.com/middleware/12213/owsm/