Oracle® Fusion Middleware

Deployment Guide for Oracle Service Bus

11*g* Release 1 (11.1.1.4.0)

E15022-02

January 2011

ORACLE®

Oracle Fusion Middleware Deployment Guide for Oracle Service Bus, 11*g* Release 1 (11.1.1.4.0)

E15022-02

Primary Authors: Floyd Jones, Legacy authors

Contributing Author:

Contributor:

Contents

List of Figures

List of Tables

Preface

This preface describes the document accessibility features and conventions of this guide--*Oracle Fusion Middleware Deployment Guide for Oracle Service Bus*.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/support/contact.html or visit http://www.oracle.com/accessibility/support.html if you are hearing impaired.

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.

Convention	Meaning
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 document describes how to deploy Oracle Service Bus configurations in a production environment. The following sections introduce key concepts and tasks for deploying Oracle Service Bus in your organization:

- Section 1.1, "Deployment Goals"
- Section 1.2, "Oracle Service Bus Deployment Topology"
- Section 1.3, "Key Deployment Tasks"
- Section 1.4, "Roles in Oracle Service Bus Deployment"
- Section 1.5, "Key Deployment Resources"

This document focuses on the deployment phase of the Oracle Service Bus software lifecycle. For a general overview of Oracle Service Bus, see *Oracle Fusion Middleware Concepts and Architecture for Oracle Service Bus*.

1.1 Deployment Goals

Oracle Service Bus combines intelligent message brokering with service monitoring and administration to provide a unified software product for implementing and deploying your Service-Oriented Architecture (SOA). When deploying Oracle Service Bus configurations, consider the following goals:

- *High Availability*. A deployment must be sufficiently available and accessible, with provisions for failover in the event of hardware or network failures.
- *Performance*. A deployment must deliver sufficient performance at peak and off-peak loads.
- *Scalability*. A deployment must be capable of handling anticipated increases in loads simply by using additional hardware resources, rather than requiring code changes.
- *Security*. A deployment must sufficiently protect data from unauthorized access or tampering.

You can achieve these goals and others with every Oracle Service Bus configuration.

1.2 Oracle Service Bus Deployment Topology

The term topology refers to a particular configuration of servers, clusters, applications, products in a domain, and where applications and products are deployed. The term Oracle Service Bus deployment topology refers to which servers/clusters the Oracle Service Bus product is deployed on.

Oracle Service Bus supports different deployment topologies, including coexistence with non-Oracle Service Bus servers/clusters in a domain. Following are the supported deployment topologies for Oracle Service Bus:

- Admin-only topology – In this topology, Oracle Service Bus is deployed on only the admin server.
- Admin + managed server topology – In this topology, Oracle Service Bus is deployed on the admin server and one non-clustered (stand-alone) managed server. Management features are deployed on the admin server, and run-time features are deployed on the managed server.
- Cluster topology – In this topology, Oracle Service Bus is deployed on the admin server and one cluster. Management features are deployed on the admin server, and run-time features are deployed on the cluster. Only one Oracle Service Bus cluster is allowed in a domain.

The first two topologies, Admin-only and Admin + managed server, are sometimes referred to as “non-clustered" Oracle Service Bus topologies.

Non-Oracle Service Bus servers/clusters can also coexist in an Oracle Service Bus deployment topology. You can create a domain using one of the Oracle Service Bus deployment topologies and create as many stand-alone non-Oracle Service Bus managed servers or clusters as you wish. These non-Oracle Service Bus servers/clusters can contain other Oracle products or user applications.

Oracle Service Bus can also be co-located with other Oracle products, with restrictions. Co-location means two products are deployed on the same servers/clusters. Currently Oracle Service Bus can be co-located with Oracle SOA Suite, though only in the Admin + managed server topology.

1.2.1 Oracle Service Bus Deployment Topology and the Oracle Service Bus Domain Extension Template

You create a particular Oracle Service Bus topology by applying an Oracle Service Bus domain extension template using the Oracle Fusion Middleware Configuration Wizard user interface or scripts. The use of domain extension templates automates most of the deployment tasks and greatly simplifies domain creation.

Oracle Service Bus provides two domain extension templates:

- Oracle Service Bus – This template lets you create all Oracle Service Bus deployment topologies.
- Oracle Service Bus for developers – This template is provided as a more convenient way of creating an Admin-only Oracle Service Bus topology; for example, in a development environment.

For information on creating Oracle Service Bus domains with the Oracle Fusion Middleware Configuration Wizard, including co-location with Oracle SOA Suite, see "Installing and Configuring Oracle Service Bus 11*g*" in the *Oracle Fusion Middleware Installation Guide for Oracle Service Bus*.

1.3 Key Deployment Tasks

Deploying Oracle Service Bus may require completing some or all of the following tasks:

1. Define the goals for your Oracle Service Bus deployment, as described in Section 1.1, "Deployment Goals."

2. Create your deployment topology, as described in Chapter 1.2, "Oracle Service Bus Deployment Topology."
3. Deploy your Oracle Service Bus configuration, as described throughout this guide.
4. Set up security for your Oracle Service Bus deployment as described in "Security" in the *Oracle Fusion Middleware Developer's Guide for Oracle Service Bus*.

1.4 Roles in Oracle Service Bus Deployment

To deploy an integrated solution successfully, a deployment team must include people who perform the following roles:

- Section 1.4.1, "Deployment Specialists"
- Section 1.4.2, "Oracle WebLogic Server Administrators"
- Section 1.4.3, "Database Administrators"

One person can assume multiple roles, and all roles are not equally relevant in all deployment scenarios. A successful deployment, however, requires input by people in each role.

1.4.1 Deployment Specialists

Deployment specialists coordinate the deployment effort. They are knowledgeable about the features of the Oracle Service Bus product. They provide expertise in designing the deployment topology for an ESB solution, based on their knowledge of how to configure various Oracle Service Bus features on one or more servers. Deployment specialists have experience in the following areas:

- Resource requirements analysis
- Deployment topology design
- Project management

1.4.2 Oracle WebLogic Server Administrators

Oracle WebLogic Server administrators provide in-depth technical and operational knowledge about Oracle WebLogic Server deployments in an organization. They have experience and expertise in the following areas:

- Hardware and platform knowledge
- Managing all aspects of a Oracle WebLogic Server deployment, including installation, configuration, monitoring, security, performance tuning, troubleshooting, and other administrative tasks.

1.4.3 Database Administrators

Database administrators provide in-depth technical and operational knowledge about database systems deployed in an organization. They have experience and expertise in the following areas:

- Hardware and platform knowledge
- Managing all aspects of a relational database (RDBMS), including installation, configuration, monitoring, security, performance tuning, troubleshooting, and other administrative tasks

1.5 Key Deployment Resources

This section provides an overview of resources that can be modified at deployment time. The term *resource* is used in this document to refer to technical assets in general, except in discussions of security where it is used to refer only to Oracle WebLogic Server entities that can be protected from unauthorized access using security roles and security policies.

The key resources that may be modified at deployment time are:

- Section 1.5.1, "Oracle WebLogic Server Resources"
- Section 1.5.2, "Oracle Service Bus Configuration Resources"
- Section 1.5.3, "Relational Database Management System Resources"
- Section 1.5.4, "Hardware, Operating System, and Network Resources"

1.5.1 Oracle WebLogic Server Resources

This section provides general information about Oracle WebLogic Server resources that are most relevant to the deployment of an Oracle Service Bus solution. You can configure these resources from the Oracle WebLogic Server Administration Console or through J2EE and WebLogic resource descriptors.

Oracle WebLogic Server provides many configuration options and tunable settings for deploying Oracle Service Bus solutions in any supported environment. The configurable Oracle WebLogic Server features that are most relevant to Oracle Service Bus deployments are:

- Section 1.5.1.1, "Clustering"
- Section 1.5.1.2, "Java Message Service"
- Section 1.5.1.3, "EJB Pooling and Caching"
- Section 1.5.1.4, "JDBC Connection Pools"
- Section 1.5.1.5, "Execution Thread Pool"
- Section 1.5.1.6, "J2EE Connector Architecture"

1.5.1.1 Clustering

A cluster is a group of servers that can be managed as a single unit. Clustering provides a deployment platform that is more scalable than a single server. To increase workload capacity, you can run Oracle WebLogic Server on a cluster. For more information about clustering, see Chapter 3, "Understanding Oracle Service Bus Clusters."

1.5.1.2 Java Message Service

The WebLogic Java Message Service (JMS) enables Java applications sharing a messaging system to exchange (create, send, and receive) messages. WebLogic JMS is based on *Java Message Service Specification* version 1.0.2 at `http://java.sun.com/products/jms/docs.html`.

JMS servers can be clustered and connection factories can be deployed on multiple instances of Oracle WebLogic Server. For more information about WebLogic JMS, see the following topics:

- "Understanding WebLogic JMS" in *Oracle Fusion Middleware Programming JMS for Oracle WebLogic Server*.

- "Configuring WebLogic JMS Clustering" and "Monitoring JMS Statistics and Managing Messages" in *Oracle Fusion Middleware Configuring and Managing JMS for Oracle WebLogic Server*.

1.5.1.3 EJB Pooling and Caching

The number of EJBs in an Oracle Service Bus deployment affects system throughput. You can tune the number of EJBs in the system through either the EJB pool or the EJB cache, depending on the type of EJB. The following table describes types of EJBs and their associated tunable parameter.

Table 1–1 Parameters for Tuning EJBs

EJB Type	Tunable Parameter Name	Tunable Parameter Description
Message-Driven Beans	`max-beans-in-free-pool`	The maximum number of listeners that pull work from a queue.
Stateless Session Beans	`max-beans-in-free-pool`	The maximum number of beans available for work requests.
Stateful Session Beans and Entity Beans	`max-beans-in-cache`	`The number of beans that can be active at once. A setting that is too low results in CacheFullExceptions`. A setting that is too high results in excessive memory consumption.

For more information about controlling throughput, see "Using the WebLogic Persistent Store" in *Oracle Fusion Middleware Configuring Server Environments for Oracle WebLogic Server*.

1.5.1.4 JDBC Connection Pools

Java Database Connectivity (JDBC) enables Java applications to access data stored in DBMS. To reduce the overhead associated with establishing database connections, WebLogic JDBC provides ready-to-use connection pools.

JDBC connection pools are used to optimize DBMS connections. If you are using the Oracle Service Bus JMS Reporting Provider, you can tune Oracle Service Bus performance by configuring the size of JDBC connection pools. A setting that is too low may result in delays while Oracle Service Bus waits for connections to become available. A setting that is too high may result in slower DBMS performance.

For more information about WebLogic JDBC connection pools, see:

- "How Connection Pools Enhance Performance" under "Performance Tuning Your JDBC Application" in *Oracle Fusion Middleware Programming JDBC for Oracle WebLogic Server*.
- "Connection Pool Features" under "Configuring JDBC Data Sources" in *Oracle Fusion Middleware Configuring and Managing JDBC for Oracle WebLogic Server*.

1.5.1.5 Execution Thread Pool

The *execution thread pool* controls the number of threads that can execute concurrently on Oracle WebLogic Server. A setting that is too low may result in sequential processing and possible deadlocks. A setting that is too high may result in excessive memory consumption, and may cause thrashing.

The number of execute threads also determines the number of threads that read incoming socket messages (socket-reader threads). This number is, by default,

one-third of the number of execute threads. A number that is too low can result in contention for threads to read sockets and can sometimes lead to a deadlock.

Set the execution thread pool high enough to run all candidate threads, but not so high that performance is hampered due to excessive context switching in the system. Monitor your running system to empirically determine the best value for the execution thread pool.

Note: Most production applications require an execution thread count greater than the default value. A thread count of 50 is a commonly used value. Be sure to adjust your JDBC connection pool to match your thread count value.

1.5.1.6 J2EE Connector Architecture

The WebLogic J2EE Connector Architecture (JCA) integrates the J2EE Platform with one or more heterogeneous Enterprise Information Systems (EIS). The WebLogic JCA is based on the *J2EE Connector Specification*, Version 1.0, from Sun Microsystems, Inc.

For information about the WebLogic J2EE-CA, see "J2EE Connector Architecture" in *Oracle Fusion Middleware Programming Resource Adapters for Oracle WebLogic Server*.

1.5.2 Oracle Service Bus Configuration Resources

Oracle Service Bus configuration resources contain environment-specific settings that you will want to change or tune when deploying the configuration to a new domain. The following sections describe the resources that you may need to reconfigure after deploying a configuration.

1.5.2.1 Business Services

Business services are Oracle Service Bus definitions of the enterprise information services with which you want to exchange messages. Business services in a production environment could specify multiple endpoints (URLs) for load balancing purposes and high availability. For information on how to add endpoints to a business service, see "Editing Business Service Configurations" the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*. For information on how to update the value of existing endpoints, see "Finding and Replacing Environment Values" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

1.5.2.2 Proxy Services

Proxy services are Oracle Service Bus definitions of intermediary Web services that Oracle Service Bus implements locally on Oracle WebLogic Server. While the majority of the metadata that defines a proxy service can be deployed without change in a new environment, there is some information you may need to update:

- Proxy service message flows route messages to named destinations (business services, other proxy services, and so on). Message routing definitions may need to be updated in a new environment.
- Definitions of proxy services for File, FTP, and Email message types must specify a single managed server for deployment of polling runtime components in a cluster. The list of managed servers appears in the Oracle Service Bus Console only for clustered Oracle Service Bus domains.
- Proxy service definitions can include directory names that may need to be updated for a new environment. For information on how to configure this resource

appropriately for your environment, see "Finding and Replacing Environment Values" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

- Proxy service definitions include references to other Oracle Service Bus resources. It is important to verify the validity of these references in a new environment.
- JMS queues and connection factories in the proxy service URL may need to be updated.
- Each proxy service relies on an instance of Oracle WebLogic Server Work Manager for its dispatch policy. You can tune the Work Manager instance to meet the requirements of your production environment. For more information, see "Using Work Manager to Optimize Scheduled Work" in *Oracle Fusion Middleware Configuring Server Environments for Oracle WebLogic Server*.

For more information about proxy services, see "Proxy Services: Creating and Managing" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

1.5.2.3 WSDLs

Oracle Service Bus uses WSDL (Web Service Definition Language) to describe proxy services and business services. WSDL is used to describe what a Web service can do, where it resides, and how to invoke it.

You can base the definition of proxy services and business services on existing WSDL files, and complete configuring the service using the Oracle Service Bus Console. WSDL files used as the basis for defining services are stored as Oracle Service Bus resources. These resources are unlikely to require updating when deployed to a new environment, because Oracle Service Bus does not use the URLs in these WSDL files at run time.

Note: Oracle Service Bus creates a new WSDL file for each HTTP proxy service. You can view the contents of this WSDL file by appending `?wsdl` to the endpoint for the service. For example, when running the Oracle Service Bus Examples Server (**Start** > **All Programs** > **Oracle Service Bus** > **Examples** > **Start Examples Server**), you can view the WSDL for the loangateway2 proxy service at http://localhost:7021/crejws_basic_ejb/loangateway2?wsdl.

1.5.2.4 Schemas

A schema is a document that defines valid content for an XML document. Schemas are used to add XML information to messages exchanged in Oracle Service Bus. These resources are unlikely to require updating when deployed to a new environment.

1.5.2.5 Service Accounts

Oracle Service Bus uses service accounts to provide authentication when connecting to a service or server. For information about using this resource appropriately in your production environment, see "Security" in the *Oracle Fusion Middleware Developer's Guide for Oracle Service Bus*.

1.5.2.6 Service Key Providers

Oracle Service Bus uses service key providers to supply credential-level validation to proxy services. The following types of security are available:

- SSL client authentication
- Digital signature

- Encryption
- Web services security X509 token

For information about how to configure this resource appropriately for your environment, see "Service Key Providers" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

1.5.2.7 WS-Policies

Oracle Service Bus uses Web Service Policies (WS-Policies) to associate Web service security policy with proxy services and business services. For information about how to configure this resource appropriately for your production environment, see "Security" in the *Oracle Fusion Middleware Developer's Guide for Oracle Service Bus*.

These resources are unlikely to require updating when deployed to a new environment.

1.5.2.8 XQuery and XSLT Transformations

Transformation maps describe the mapping between two data types. Oracle Service Bus supports data mapping using either XQuery or the eXtensible Stylesheet Language Transformation (XSLT) standard. These resources are unlikely to require updating when deployed to a new environment.

1.5.2.9 MFLs

Message Format Language (MFL) is an Oracle proprietary language used to define rules to transform formatted binary data into XML data. MFL documents are unlikely to require updating when deployed to a new environment.

1.5.2.10 JARS

JARs (Java ARchive) are zipped files that contain a set of Java classes. They are used to store compiled Java classes and associated metadata that can constitute a program. JARs act as callable program libraries for Java code elements (so that a single compilation link provides access to multiple elements, rather than requiring bindings for each element individually). JARs are unlikely to require updating when deployed to a new environment.

1.5.2.11 Alert Destinations

An Alert Destination resource captures a list of recipients that can receive alert notifications from the Oracle Service Bus. In typical system monitoring contexts, alerts generated by Oracle Service Bus bear significance to a finite set of users. In Oracle Service Bus, each Alert Destination resource may be configured to include a set of recipients according to a given context. Alert Destinations are used by Alert actions configured in the message flow, and also by SLA alert rules. An Alert destination could include one or more of the following types of destinations: Console, Reporting Data stream, SNMP trap, Alert log, E-mail, JMS queue, or JMS topic. In the case of E-mail and JMS destinations, a destination resource could include a list of E-mail addresses or JMS URIs, respectively. Alert Destinations are unlikely to require updating when deployed to a new environment.

1.5.2.12 UDDI Registries

A UDDI Registry resource is a global resource that stores information about UDDI Registries used by Oracle Service Bus. After the UDDI Registry resource is configured, you can then publish Oracle Service Bus proxy services to the associated registry, or

import business services from the registry to be used by an Oracle Service Bus proxy service. You must be in an active session to configure the UDDI registry resource. UDDI Registry resources must be updated or reconfigured when deployed to a new environment. For more information on updating UDDI Registry resources during deployment, see Section 1.5.2.15, "Best Practices to Follow When Migrating Global Resources."

1.5.2.13 JNDI Providers

JNDI Provider resources are definitions of JNDI providers that describe the URL (or list of URLs in the case of clustered deployments) of the JNDI providers used by Oracle Service Bus. They are global resources and can be re-used across projects with an Oracle Service Bus domain. If the JNDI provider is secured, then the JNDI Provider resource description also carries a user name and password to gain access. JNDI Provider resources must be updated or reconfigured when deployed to a new environment. For more information on updating JNDI Provider resources during deployment, see Section 1.5.2.15, "Best Practices to Follow When Migrating Global Resources."

1.5.2.14 SMTP Servers

SMTP Server resources are definitions of SMTP Servers that describe the URL and port for the SMTP Servers used by the Oracle Service Bus. They are global resources and can be re-used across projects with an Oracle Service Bus domain. If the SMTP Server is secured, then the SMTP Server resource description also carries a user name and password to gain access. SMTP Server resources are used while configuring Alert Destination resources and E-mail transport-based Business Services. SMTP Server resources must be updated or reconfigured when deployed to a new environment. For more information on updating SMTP Server resources during deployment, see Section 1.5.2.15, "Best Practices to Follow When Migrating Global Resources."

1.5.2.15 Best Practices to Follow When Migrating Global Resources

Global resources include the UDDI Registry, JNDI Provider, and SMTP Server resources. These resources typically have different values in testing environments from those that will be used in staging or production environments. When you migrate a deployment from a test environment to a production environment, use one of the following methods:

1. Create new global resources on the production domain with the same names as in the test environment. Alternatively, import these resources directly from the test environment for the first time, and then customize them (rather than spend time creating them manually).
2. When exporting artifacts from the test environment, exclude the global resources. Then simply import the `config.jar` file. If there are any resources in the `config.jar` file that references Alert Destinations, JNDI providers, or SMTP providers, these references will simply snap into place when the import is complete.
3. Alternatively, export all the artifacts (including the global resources), but exclude the global resources when importing the `config.jar` file.

1.5.3 Relational Database Management System Resources

Oracle Service Bus relies on database resources for storing message-reporting data by the JMS Reporting Provider. Database performance is a factor in overall Oracle Service Bus performance.

For additional information on tuning your database, see your database vendor's documentation.

1.5.4 Hardware, Operating System, and Network Resources

Hardware, operating system, and network resources play a crucial role in Oracle Service Bus performance. Deployments must comply with the hardware and software requirements described in the "Oracle Fusion Middleware Supported System Configurations" at http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html.

2

Configuring a Non-Clustered Deployment

This section describes the tasks that you must perform to configure Oracle Service Bus for deployment in a non-clustered Oracle WebLogic Server domain. A non-clustered Oracle Service Bus deployment refers to one of the following Oracle Service Bus deployment topologies: Admin-only and Admin + managed server, as described in Chapter 1.2, "Oracle Service Bus Deployment Topology."

To set up and deploy Oracle Service Bus in a non-clustered configuration, complete the following steps:

- Section 2.1, "Step 1. Configure a Database for the JMS Reporting Provider Data Store"
- Section 2.2, "Step 2. Prepare an Oracle Service Bus Domain"
- Section 2.3, "Step 3. Configure Oracle Service Bus Security"
- Section 2.4, "Step 4. Deploy an Oracle Service Bus Configuration"
- Section 2.5, "Step 5. Update Your Domain as Your Production Environment Changes"

2.1 Step 1. Configure a Database for the JMS Reporting Provider Data Store

Oracle Service Bus requires a database for the JMS Reporting Provider. The local copy of the Apache Derby database that is installed with Oracle WebLogic Server is not recommended for production use.

For information on configuring a database for the Oracle Service Bus JMS reporting provider, see the *Oracle Fusion Middleware Installation Guide for Oracle Service Bus*.

For a complete list of supported databases, see the "Oracle Fusion Middleware Supported System Configurations" at `http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html`.

Note: It is important to configure your database appropriately for production use. You must provide adequate space to log messages and follow best practices for administering your database.

2.2 Step 2. Prepare an Oracle Service Bus Domain

To prepare a Oracle Service Bus environment, complete the tasks described in the following sections:

2.2.1 Creating an Oracle Service Bus Domain Using the Oracle Fusion Middleware Configuration Wizard

For information on creating an Oracle Service Bus domain with the Oracle Fusion Middleware Configuration Wizard, see "Installing and Configuring Oracle Service Bus 11*g*" in the *Oracle Fusion Middleware Installation Guide for Oracle Service Bus*.

2.2.2 Configuring JMS Resources

In addition to configuring JMS file stores in the Oracle Fusion Middleware Configuration Wizard, proxy services and business services that use JMS require configuration of the following resources:

- JMS connection factories. You must configure XA or non-XA JMS connection factories for all business services and proxy services implemented using JMS.
- JMS queues/topics. Oracle Service Bus automatically configures JMS queues for proxy services that are implemented using JMS. You must configure JMS queues/topics for all business services using JMS and for any proxy services that are implemented using non- JMS.

If you want to concentrate all Oracle Service Bus JMS resources in a single JMS module, use the Oracle WebLogic Server Administration Console to create a new JMS module containing the destination to be used for the proxy services' endpoint.

For more information about configuring JMS resources, see "Methods for Configuring JMS Resources" in the *Oracle Fusion Middleware Configuring and Managing JMS for Oracle WebLogic Server*.

2.3 Step 3. Configure Oracle Service Bus Security

Oracle Service Bus leverages the security features of Oracle WebLogic Server to ensure message confidentiality and integrity (message-level security), secure connections between clients and Oracle WebLogic Server (transport-level security), and authentication and authorization (access control). For information on how to configure security for Oracle Service Bus, see "Security" in the *Oracle Fusion Middleware Developer's Guide for Oracle Service Bus*.

Note: You must configure security separately for each Oracle Service Bus domain. Oracle Service Bus does not export or import security configurations.

2.4 Step 4. Deploy an Oracle Service Bus Configuration

Once you have configured your Oracle Service Bus domain, secured it, and added any JMS resources required for its services, you are ready to import the JAR file that contains your Oracle Service Bus configuration. After you have imported the configuration metadata, you can update environment-specific information for your domain.

The following steps describe the basic procedure for deploying the contents of configuration JAR file:

1. Create a session in Oracle Service Bus.
2. Import all or selected objects from a configuration JAR file.
3. Update environment-specific information such as service endpoint URIs and directory names.
4. Activate the Session.

You can perform these steps manually or programmatically:

- To import and update a configuration manually, use the Oracle Service Bus Console as described under the following topics in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*:
  - "Importing Resources"
  - "Finding and Replacing Environment Values"
- To import and update a configuration programmatically, use the WebLogic Scripting Tool (WLST) and the Oracle Service Bus `deploymentMBean` as described in Appendix A, "Using the Deployment APIs."

In addition to service endpoint URIs, directory names, and security configuration, your Oracle Service Bus configuration may contain other settings that must be updated to operate correctly in the new environment. Items that commonly require update include the following:

- Service references

  For information about service references, see "Viewing References to Resources" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

- Routing destinations

  For information about routing configuration, see "Proxy Services: Message Flow" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

- Load balancing settings

  For information about load balancing, see "Transport Configuration Page" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

Use the Oracle Service Bus Console to confirm and change your configuration, as necessary.

2.5 Step 5. Update Your Domain as Your Production Environment Changes

Production environments change over time and as application use increases. This section describes how to update your domain in response to common production environment change scenarios:

- Section 2.5.1, "Changing a Business Service"
- Section 2.5.2, "Installing a New Version of a Proxy Service"
- Section 2.5.3, "Online Configuration Updates"

2.5.1 Changing a Business Service

Enterprise information services (EIS) are sometimes phased out, and new instances (possibly with new versions of EIS software, new hardware, and so on) are brought online. When this happens, Oracle Service Bus administrators need to gracefully

transition to the new EIS instance by modifying any affected Oracle Service Bus business services.

This situation is similar to an EIS instance failure, but not as urgent. For a description of deployment considerations, see Section 5.2.2, "EIS Instance Failover." For information about using the Oracle Service Bus Console to change an endpoint URI for a business service, see "Transport Configuration Page" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

2.5.2 Installing a New Version of a Proxy Service

As your business requirements change, you may need to make changes to your proxy services. If the changes you need to make are backward compatible, you can dynamically make changes online using the Oracle Service Bus Console to create a new version of the proxy service. Changes are backward compatible if they meet one of the following criteria:

- The interface of the changed object is unchanged.
- Old and new clients will work with the interface.

If the changes you need to make are not backward compatible, there are two alternatives to consider that would enable you to make the changes online:

- Create and deploy a new proxy service having a different name and URL from that of the earlier version. Clients upgrade by accessing the new proxy service. This enables you to run the old and new versions of a proxy service in parallel, and supports a gradual migration to the new proxy service.
- Force backwards compatibility by changing the proxy service interface to support both the new interface and the old interface (for example, using XML schema choice) and perform different logic in the message flow based on the document received. Clients continue to access the proxy service by using its original URL.

Oracle Service Bus cluster domains have additional system administration requirements for deployment of proxy services that are not backward compatible. For more information, see Section 4.6.4, "Installing a New Version of a Proxy Service in a Cluster."

2.5.3 Online Configuration Updates

Oracle Service Bus allows you to dynamically change the configuration information for a system without the need to restart the server for changes to take affect.

You can change a resource, a project, or a number of resources (related or unrelated) using the Oracle Service Bus Console using the following procedure:

1. Create a session. All changes to Oracle Service Bus configurations require a session. (Security-related changes are the exception.)
2. Modify resources in the session.or import all or selected objects from a configuration JAR file.
3. Update environment-specific information such as service endpoint URIs and directory names.
4. Activate the session.

   The changes are consolidated and sent to all servers (administration and managed servers, if you are working in a cluster environment). These changes update the persisted configuration data and also cause other run-time tasks to be performed (such as, creating proxy services and JMS queues, compiling XQueries, and so on).

You can perform these steps manually or programmatically:

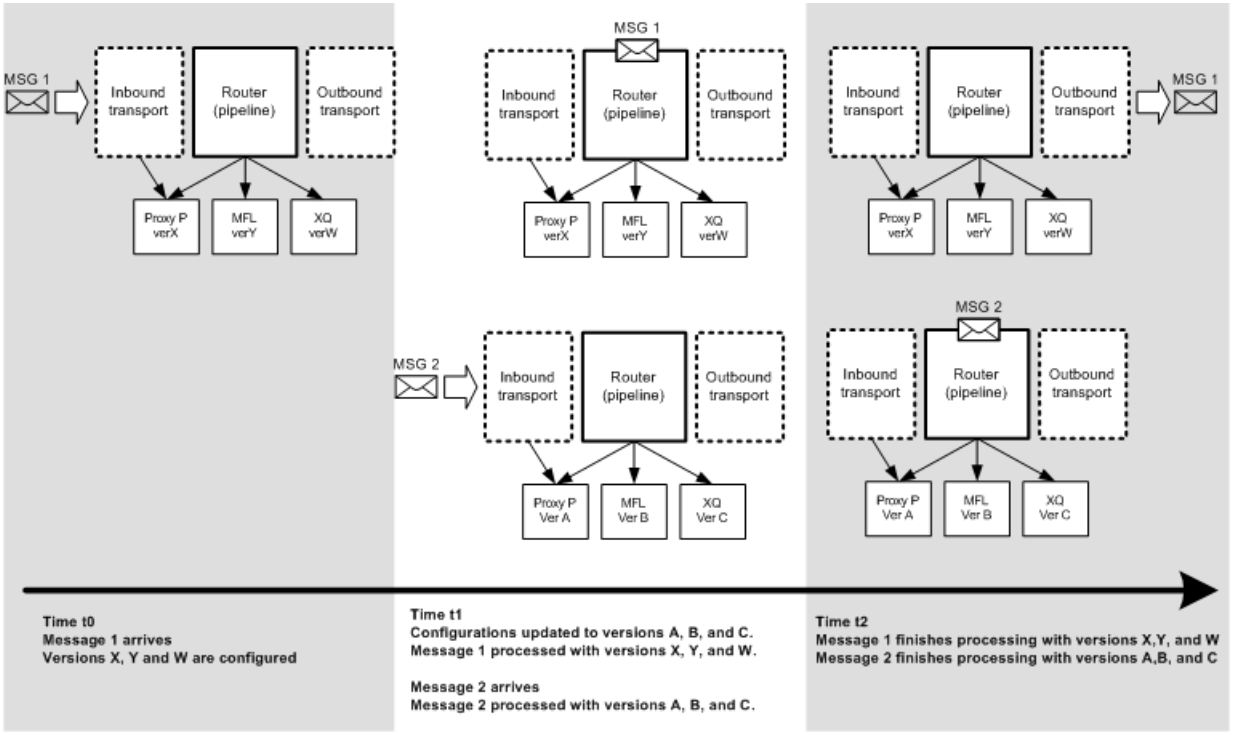
- To import and update a configuration manually, use the Oracle Service Bus Console as described in the following topics in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*:
  - "Importing Resources"
  - "Finding and Replacing Environment Values"
- To import and update a configuration programmatically, use the WebLogic Scripting Tool (WLST) and the Oracle Service Bus `deploymentMBean` as described in Appendix A, "Using the Deployment APIs."

Figure 2–1 illustrates how the system behaves to process messages in the event that the configuration is updated while messages are being processed through the system. Table 2–1 describes the versions for the resources for the sample system illustrated in Figure 2–1.

Table 2–1 Initial and Updated Configuration for a Sample System

Resource	Initial Version	Updated Version
Proxy Service	X	A
MFL	Y	B
XQuery	W	C

Figure 2–1 Sample Online Update Scenario



Note the following characteristics of the message processing illustrated in the preceding figure:

- Message 1 is already in the system at t1 (the time the configuration is updated)
- Message 1 completes processing by the original (pre-update) resources (X, Y, W)

- Message 2 starts and completes processing with the new configuration (resources A, B, C)

Oracle Service Bus tries to execute messages with the version of the proxy service and artifacts available when the messages enters the proxy service.

This ensures that a message has a consistent view of the artifacts. If the message processor cannot guarantee this behavior for a message, it will reject it rather than process it incorrectly. If you want the system to retry rejected messages, use a JMS proxy service with retries.

2.5.3.1 Best Practices for Successful Online Configuration Updates

This section describes best practices to follow and limitations to be aware of when you update a configuration in a running Oracle Service Bus system.

- If you are concerned about message rejection by Oracle Service Bus, use the JMS transport protocol with retries. In this case, any messages that are rejected because the system cannot guarantee their processing by compatible resources will be retried.
- Security-related configuration updates must be performed first, then update the Oracle Service Bus resources in your system. To learn about updating security resources, see "Overview of Security Management" in *Oracle Fusion Middleware Securing Oracle WebLogic Server*.
- Updates must be compatible with existing clients using the system. See Section 2.5.2, "Installing a New Version of a Proxy Service."
- If you are updating the configuration to a cluster, it is possible that the updates are done at different times on different managed servers. Consequently, it is possible that messages are processed by different versions of a proxy service, depending on which managed server gets the message to process. This is dependent on load balancing across managed servers.
- During online deployment, Oracle Service Bus checks whether the correct versions of referenced resources are used for message processing. If this is temporarily not true, an error is returned. However, if the interface artifact of an invoked service changes (for example: MFL, WSDL), the invoking proxy service may not return an error although it temporarily sees a version of the artifact that does not correlate with the proxy service version.

3

Understanding Oracle Service Bus Clusters

The following sections describe how Oracle Service Bus is configured and deployed in a clustered environment. It contains the following topics:

3.1 Understanding Oracle Service Bus Clusters

Clustering allows Oracle Service Bus to run on a group of servers that can be managed as a single unit. In a clustered environment, multiple machines share the processing load. Oracle Service Bus provides load balancing so that resource requests are distributed proportionately across all machines. An Oracle Service Bus deployment can use clustering and load balancing to improve scalability by distributing the workload across nodes. Clustering provides a deployment platform that is more scalable than a single server.

An Oracle WebLogic Server cluster domain consists of only one administration server, and one or more managed servers. The managed servers in an Oracle Service Bus domain can be grouped in a cluster. When you configure Oracle Service Bus clusterable resources, you normally target the resources to the named cluster. The advantage of specifying a cluster as the target for resource deployment is that it makes it possible to dynamically increase capacity by adding managed servers to your cluster.

> **Note:** Oracle Service Bus domains can support a single cluster, and all managed servers in the domain must belong to that cluster.

The topics in this section provide the information you need to configure Oracle Service Bus in a clustered environment. Although some background information about how Oracle WebLogic Server supports clustering is provided, the focus is on procedures that are specific to configuring Oracle Service Bus for a clustered environment.

Before proceeding, we recommend that you review the following sections of the Oracle WebLogic Server documentation to obtain a more in-depth understanding of clustering:

- *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*

- *Oracle Fusion Middleware Configuring Server Environments for Oracle WebLogic Server*

3.2 Designing a Clustered Deployment

The following sections provide the information you need to design a clustered deployment:

- Section 3.2.1, "Introducing Oracle Service Bus Domains"
- Section 3.2.2, "Oracle Service Bus Deployment Resources"
- Section 3.3, "Load Balancing in a Oracle Service Bus Cluster"

3.2.1 Introducing Oracle Service Bus Domains

Before you begin designing the architecture for your clustered domain, you need to learn how Oracle WebLogic Server clusters operate.

3.2.1.1 Creating Domains

Domain and cluster creation are simplified by the Oracle Fusion Middleware Configuration Wizard, which lets you generate domains from basic and extension domain templates. Based on responses to user queries, the Oracle Fusion Middleware Configuration Wizard generates a domain, server, and enterprise application with the appropriate components preconfigured and assets included. For information about creating Oracle Service Bus domains using the Oracle Fusion Middleware Configuration Wizard, see the *Oracle Fusion Middleware Installation Guide for Oracle Service Bus*.

3.2.1.2 Clustered Servers

A server can be either a managed server or an administration server. An Oracle WebLogic Server running the administration service is called an *administration server* and hosts the Oracle WebLogic Server Administration Console. In a domain with multiple Oracle WebLogic Servers, only one server is the administration server; the other servers are called *managed servers*. Each managed server obtains its configuration at startup from the administration server.

Note: Managed servers that start without an administration server operate in Managed Server Independence (MSI) mode. For information about MSI mode, see "Managed Server Independence Mode" in *Oracle Fusion Middleware Managing Server Startup and Shutdown for Oracle WebLogic Server*.

You can enable an SSL port, an HTTP cleartext port, or both ports for the administration server. In secure installations, the HTTP cleartext port can be disabled. However, when Oracle Service Bus is used in combination with a UDDI registry (for example, Oracle Service Registry), you must ensure that the server's HTTP cleartext port is enabled.

For general information about WebLogic clusters, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*. This document includes details regarding recommended basic, multi-tiered, and proxy architectures. For information about security considerations in the design of WebLogic clusters, see "Security Options for Cluster Architectures" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

3.2.2 Oracle Service Bus Deployment Resources

For each server in a clustered domain, you can configure a variety of attributes that define the functionality of the server in the domain. These attributes are configured automatically when you create an Oracle Service Bus domain using the Oracle Fusion Middleware Configuration Wizard. Advanced users can also configure these attributes manually using the Servers node in the Oracle WebLogic Server Administration Console.

For a list of configurable Oracle Service Bus deployment resources, see Appendix B, "Oracle Service Bus Deployment Resources." It describes the default targeting of each resource in a clustered Oracle Service Bus domain and provides instructions on how to navigate to each resource in the Oracle WebLogic Server Administration Console.

3.2.2.1 Singleton Resources

While most resources used by Oracle Service Bus are deployed homogeneously across the cluster, there are a number of resources that must be pinned to a single managed server in order to operate correctly. The following table lists these components, describes how they are targeted, and provides a link for more information.

Table 3–1 Oracle Service Bus Singleton Resources

Resource	How and Where to Target	For more information, see...
File, FTP, and E-mail pollers for proxy services	Specified manually in the proxy service definition using the Oracle Service Bus Console. The poller is deployed on all managed servers, but the poller on only one managed server will poll for a given proxy service.	*Oracle Fusion Middleware Installation Guide for Oracle Service Bus*
ALSB Domain Singleton Marker Application	Targeted in the Oracle Fusion Middleware Configuration Wizard	*Oracle Fusion Middleware Installation Guide for Oracle Service Bus*
SLA Manager	Targeted in the Oracle Fusion Middleware Configuration Wizard	*Oracle Fusion Middleware Installation Guide for Oracle Service Bus*
Oracle Service Bus JMS server on each managed server `wlsbJMSServer_auto_1-n`	Automatically targeted in the Oracle Fusion Middleware Configuration Wizard	*Oracle Fusion Middleware Installation Guide for Oracle Service Bus*

3.2.2.2 Monitoring and Alert Resources in a Cluster

The ALSB Domain Singleton Marker Application (domainsingletonmarker.ear) runs as a singleton service on one of the managed servers in the cluster. Data collection is performed on each of the managed servers in the domain. The ALSB Domain Singleton Marker Application is responsible for the collection and aggregation of data from all managed servers in the domain. The aggregated data are processed and classified by each Oracle Service Bus configuration.

Oracle Service Bus configurations can include rules defining Service Level Agreements (SLAs) for system performance. The Alert Manager is responsible for storing rules, and evaluates these rules against the data aggregated for the cluster. When a rule evaluates to `true`, the Alert Manager sends an e-mail message, posts a message on a JMS queue, or logs a message according to the action associated with the rule.

For more information about these features, see "Monitoring" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

3.2.2.3 Cluster Configuration Changes and Deployment Requests

When a managed server is down during session activation, configuration changes from the activation are not reflected on that server. In addition, the task status of the session activation is listed as *Partially Activated* to indicate that the activation was not completed on all managed servers. After the managed server is restarted, it synchronizes with the information available with the administration server, and any unactivated changes are activated on the managed server. For more information about session activations, see "Using the Change Center" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

You can only re-configure a cluster (for example, add new nodes to the cluster or modify business service configuration) when its administration server is active.

If the administration server for a cluster is down, deployment or undeployment requests are interrupted, but managed servers continue serving requests. You can boot or reboot managed servers using an existing configuration, as long as the required configuration files (`msi-config.xml`, `SerializedSystemIni.dat`, and optionally `boot.properties`) exist in each managed server's root directory.

3.3 Load Balancing in a Oracle Service Bus Cluster

One of the goals of clustering your Oracle Service Bus application is to achieve scalability. For a cluster to be scalable, each server must be fully utilized. Load balancing distributes the workload proportionately across all the servers in a cluster so that each server can run at full capacity. The following sections describe inbound message processing load balancing for Oracle Service Bus clusters:

- Section 3.3.1, "Load Balancing HTTP Functions in a Cluster"
- Section 3.3.2, "Load Balancing JMS Functions in a Cluster"

For more information about inbound message load balancing, see "Load Balancing in a Cluster" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*. For information about configuring load balancing for business services, see "Transport Configuration" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

3.3.1 Load Balancing HTTP Functions in a Cluster

Web services (SOAP or XML over HTTP) can use HTTP load balancing. External load balancing can be accomplished through the WebLogic HttpClusterServlet, a WebServer plug-in, or a hardware router. For an overview of a cluster topology that includes load balancing, see Figure 5–1. Oracle WebLogic Server supports load balancing for HTTP session states and clustered objects. For more information, see "Communications in a Cluster" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

3.3.2 Load Balancing JMS Functions in a Cluster

Most JMS queues used by Oracle Service Bus are configured as distributed destinations. Exceptions are JMS queues that are targeted to single managed servers.

For detailed information on JMS load balancing, see "Controlling the Flow of Messages on JMS Servers and Destinations" in *Oracle Fusion Middleware Performance and Tuning for Oracle WebLogic Server*.

3.4 High Availability in an Oracle Service Bus Cluster

Message-driven beans consume messages from JMS destinations. A number of message-driven beans are deployed on each Oracle Service Bus destination.

3.4.1 Highly Available JMS for Oracle Service Bus

The ability to configure multiple physical destinations as members of a single distributed destination set provides a highly available implementation of WebLogic JMS. Specifically, for each node in a cluster, an administrator should configure one physical destination for a distributed destination. If one node in the cluster fails, making the physical destination for that node unavailable, then other physical destinations configured as members of the distributed destination can provide services to JMS producers and consumers. (This is how the Oracle Fusion Middleware Configuration Wizard generates domains for a cluster.)

Message-driven beans consume messages from distributed destinations. Distributed destinations contain one physical destination for each instance of Oracle WebLogic Server. A single message producer on a distributed queue is bound to a single physical destination. Message-driven beans are bound to the physical destination in the server on which they are deployed (server affinity).

For more information, see Chapter 5, "Understanding Oracle Service Bus High Availability."

3.5 Deploying Configurations

Configurations are deployed by importing the contents of one or more JAR files exported from the Oracle Service Bus Console. You deploy an Oracle Service Bus configuration in a clustered environment following the same procedure as for a non-clustered deployment. For a description of the deployment procedure, see Section 2.4, "Step 4. Deploy an Oracle Service Bus Configuration."

Preparations for deployment of an Oracle Service Bus configuration in a production cluster environment involve more system administration tasks than for a non-clustered testing or staging environment. For a full description of the steps involved in a production cluster deployment, see Chapter 4, "Configuring a Clustered Deployment."

4

Configuring a Clustered Deployment

This section describes the tasks that you must perform to configure Oracle Service Bus for deployment in a clustered environment.

> **Note:** If you want to run Oracle Service Bus on a managed server in a non-clustered environment, see Chapter 2, "Configuring a Non-Clustered Deployment."

After planning the architecture of your clustered domain, as described in Section 3.2, "Designing a Clustered Deployment," you are ready to set up Oracle Service Bus in a clustered environment. To do this, you must configure an administration server and managed servers, and then deploy Oracle Service Bus resources to the servers. You also need a router (hardware or software), if you need inbound HTTP load balance functions. The persistent configuration for a domain of Oracle WebLogic Server instances and clusters is stored in an XML configuration file (`config.xml`) in the `config` directory of the root directory of your Oracle Service Bus domain.

To set up and deploy Oracle Service Bus in a clustered domain, complete the following steps:

- Section 4.1, "Step 1. Comply with Configuration Prerequisites"
- Section 4.2, "Step 2. Prepare an Oracle Service Bus Domain"
- Section 4.3, "Step 3. Configure Oracle Service Bus Security"
- Section 4.4, "Step 4. Starting, Stopping, and Monitoring Managed Servers"
- Section 4.5, "Step 5. Deploy an Oracle Service Bus Configuration"
- Section 4.6, "Step 6. Update Your Domain as Your Production Environment Changes"

For information about deploying Oracle Service Bus in a non-clustered environment, see Chapter 2, "Configuring a Non-Clustered Deployment."

4.1 Step 1. Comply with Configuration Prerequisites

This section describes prerequisites for configuring Oracle Service Bus to run in a clustered environment:

- Obtain an IP address for the administration server you will use for the cluster.

  All Oracle WebLogic Server instances in a cluster use the same administration server for configuring and monitoring. When you add servers to a cluster, you must specify the administration server that each will use.

- Define a multicast address for each cluster.

  Note: You are prompted to provide a multicast address when you create an Oracle Service Bus domain using the Oracle Fusion Middleware Configuration Wizard. (See Section 4.2, "Step 2. Prepare an Oracle Service Bus Domain.")

  The multicast address is used by cluster members to communicate with each other. Clustered servers must share a single, exclusive, multicast address. For each cluster on a network, the combination of multicast address and port must be unique. If two clusters on a network use the same multicast address, they should use different ports. If the clusters use different multicast addresses, they can use the same port or accept the default port, `7001`. To support multicast messages, the administration server and the managed servers in a cluster must be located on the same subnet.

- Define IP addresses for the servers in your cluster. You can do this in a number of ways:

  Note: You are prompted to provide listen addresses for servers when you create a Oracle Service Bus domain using the Oracle Fusion Middleware Configuration Wizard. (See Section 4.2, "Step 2. Prepare an Oracle Service Bus Domain.")

  - Assign a single IP address and different listen port numbers to the servers in the cluster.

    By assigning a single IP address for your clustered servers with a different port number for each server, you can set up a clustered environment on a single machine without the need to make your machine a multi-homed server.

    To access such an IP address from a client, structure the IP address and port number in your URL in one of the following ways:

    * ipAddress:portNumber-portNumber – When the port numbers are sequential. For example:

      127.0.0.1:7003-7005

    * ipAddress:portNumber+...+portNumber – When the port numbers are not sequential. For example:

      127.0.0.1:7003+7006+7008

    * ipAddress:portNumber,ipAddress:portNumber,... – Verbose, explicit specification. For example:

      127.0.0.1:7003,127.0.0.1:7004,127.0.0.1:7005

  - Assign a static IP address for each Oracle WebLogic Server instance to be started on each machine in the cluster.

    In this case, when multiple servers are run on a single machine, that machine must be configured as a multi-homed server, that is, multiple IP addresses are assigned to a single computer. Under these circumstances, structure the cluster address as a comma-separated list of IP addresses.

For example, the following listing is an example of a cluster address specified in a `config.xml` file. It specifies a static IP address for each of the four servers in a cluster named `MyCluster`:

```
<Cluster
ClusterAddress="127.0.0.1:7001,127.0.0.2:7001,127.0.0.3,127.0.0.4:7001"
Name="MyCluster"/>
```

You can also use a DNS approach to identifying servers.

For more information on addressing issues, see "Avoiding Listen Address Problems" in "Setting Up WebLogic Clusters" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

Note: In test environments, it is possible to have multiple Oracle WebLogic Server instances on a single machine. In these circumstances, you can have some Oracle WebLogic Server instances on the same node with different port numbers and some on different nodes with the same port number.

- Configure one of the following databases for your clustered domain:
  - Microsoft SQL Server
  - Oracle

Note: The local copy of the Apache Derby database that is installed with Oracle WebLogic Server is for evaluation purposes only.

It is important to configure your database appropriately for production use. You must provide adequate space to store data and log messages, and follow best practices for administering your database.

Note: You can configure your database to use concurrent access.

For the latest information about issues regarding specific databases, see the product release notes.

- Include a shared file system. A shared file system is required for any cluster you want to be highly available. We recommend either a Storage Area Network (SAN) or a multiported disk system.

  For information about configuring a highly available cluster, see "Configuring WebLogic JMS Clustering" in "Configuring Advanced JMS System Resources" in *Oracle Fusion Middleware Configuring and Managing JMS for Oracle WebLogic Server*.

- Configure a hardware or software router for your system. Load balancing can be accomplished using either the built-in load balancing capabilities of a WebLogic proxy plug-in or separate load balancing hardware.

  For information about hardware and software routers, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

> **Note:** Additional requirements apply when you design your domain to include one or more firewalls. For a description of how to add firewall information to your domain configuration file, see Section 4.2.1.1, "Adding Proxy Server or Firewall Information to your Domain Configuration." For additional information, see "Communications in a Cluster" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

- Oracle Service Bus load balances File, Email, and FTP transport processing across the managed servers in a cluster. All managed servers in the cluster should be able to access the Archive, Stage, and Error directories specified in any File, Email, or FTP proxy service configuration. These directories should be configured in a shared file system such as NFS. By using a shared file system, users and programs can access files on remote systems almost as if they were local files.

For more information about setting up clustered Oracle WebLogic Server instances, see "Setting Up WebLogic Clusters" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

4.2 Step 2. Prepare an Oracle Service Bus Domain

When preparing an Oracle Service Bus domain, you must add a definition for each managed server to the domain configuration file (`config.xml`), assign all managed servers to a cluster, specify the Oracle Service Bus components on the servers in your domain, and so on.

To prepare an Oracle Service Bus environment in a clustered domain, complete the tasks described in the following sections:

- Section 4.2.1, "Creating an Oracle Service Bus Domain Using the Oracle Fusion Middleware Configuration Wizard"
- Section 4.2.2, "Configuring JMS Resources"

4.2.1 Creating an Oracle Service Bus Domain Using the Oracle Fusion Middleware Configuration Wizard

For information on creating an Oracle Service Bus domain with the Oracle Fusion Middleware Configuration Wizard, see "Installing and Configuring Oracle Service Bus 11*g*" in the *Oracle Fusion Middleware Installation Guide for Oracle Service Bus*.

4.2.1.1 Adding Proxy Server or Firewall Information to your Domain Configuration

If you will be using Web services behind a proxy server or firewall, you must edit the `config.xml` file to include information about that proxy server or firewall.

To add proxy server or firewall information to your domain configuration, complete the following steps:

1. Open `config.xml` with an ASCII editor.
2. Find the line that starts with the following tag in the `config.xml` file:

   ```
   <Cluster
   ```

3. Add the following three attributes to the Cluster attribute list:

   ```
   FrontendHTTPPort="proxyPort" FrontendHTTPSPort="proxySSLPort"
   ```

```
FrontendHost="proxyServerHost"
```

For example, the following listing is an example of a cluster address with a firewall specified in a `config.xml` file for a cluster named `MyCluster` and a proxy server named `MyProxy`:

```
<Cluster
ClusterAddress="127.0.0.1:7001,127.0.0.2:7001,127.0.0.3,127.0.0.4:7001"
FrontendHTTPPort="7006" FrontendHTTPSPort="7007" FrontendHost="MyProxy"
MulticastAddress="127.0.0.5" MulticastPort="7010" Name="MyCluster"/>
```

4. Save your changes and close the `config.xml` file.

4.2.2 Configuring JMS Resources

In addition to configuring JMS file stores in the Oracle Fusion Middleware Configuration Wizard, proxy services and business services that use JMS require configuration of the following resources:

- JMS queues/topics. Oracle Service Bus automatically configures JMS queues for proxy services that are implemented using JMS. You must configure JMS queues/topics for all business services using JMS and for any proxy services that are implemented using non- JMS.

  Proxy services can consume messages from a remote queue on a separate domain. In this case, Oracle Service Bus will not create the queue for you. The JMS queues can be created for proxy services only if the queues are on the same local Oracle Service Bus domain.

- JMS connection factories. You must configure JMS connection factories for all business services and proxy services implemented using JMS.

For information about configuring JMS resources, see "Configuring Advanced JMS System Resources" in *Oracle Fusion Middleware Configuring and Managing JMS for Oracle WebLogic Server*.

4.3 Step 3. Configure Oracle Service Bus Security

Oracle Service Bus leverages the security features of Oracle WebLogic Server to ensure message confidentiality and integrity (message-level security), secure connections between clients and Oracle WebLogic Server (transport-level security), and authentication and authorization (access control). For information about the tasks you must complete, see "Security" in the *Oracle Fusion Middleware Developer's Guide for Oracle Service Bus*.

> **Note:** You must configure security separately for each Oracle Service Bus domain. Oracle Service Bus does not export or import security configurations.

If you want to configure SSL for your cluster, you can do so when creating your domain or by using the Oracle WebLogic Server Administration Console. For a domain in which security functionality is deployed in a multi-node cluster, you also need to configure keystores, server certificate and private key for each managed server, and so on, for every machine in a cluster. You either need to use a separate keystore for each machine or you can use a single keystore if it is available to all machines.

4.4 Step 4. Starting, Stopping, and Monitoring Managed Servers

This section describes the basic management tasks for the managed servers in your clustered domain:

- Section 4.4.1, "Starting and Stopping Managed Servers"
- Section 4.4.2, "Monitoring Your Servers"

4.4.1 Starting and Stopping Managed Servers

Node Manager is a utility that enables you to start, stop, and migrate your Oracle WebLogic Server instances. You can start your managed servers using Node Manager in conjunction with the Oracle WebLogic Server Administration Console, or you can create WLST scripts to automate Node Manager functionality.

Tip: Run the `setDomainEnv` script before the `startNodemanager` script to ensure that the Oracle Service Bus classes are available to spawned servers. Alternatively, explicitly set classpath in the Oracle WebLogic Server Administration Console before attempting to start the server.

By default, when the Oracle Fusion Middleware Configuration Wizard generates an Oracle Service Bus cluster domain:

- The Domain Singleton Marker Application (`domainsingletonmarker.ear`) is targeted to the first managed server in the cluster. For data aggregation to function properly, the server that `domainsingletonmarker.ear` is targeted to must be started first and must be available when other managed servers are started.
- The PurgingMDB (`msgpurger.ear`) is targeted to the first managed server in the cluster. For message purging to function properly, the server where `msgpurger.ear` is targeted must be available.

For more information on Node Manager, see "Using Node Manager to Control Servers" in *Oracle Fusion Middleware Node Manager Administrator's Guide for Oracle WebLogic Server*. For a complete overview of methods to start and stop managed servers, see "Starting and Stopping Servers" in *Oracle Fusion Middleware Managing Server Startup and Shutdown for Oracle WebLogic Server*.

4.4.2 Monitoring Your Servers

Once startup is complete, you can use the Oracle Service Bus Console to verify the status of servers. For information about using Oracle Service Bus Console to monitor your servers, see "Listing and Locating Servers" in "Monitoring" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

4.5 Step 5. Deploy an Oracle Service Bus Configuration

You deploy an Oracle Service Bus configuration in a clustered environment following the same procedure as for a non-clustered deployment. For a description of the deployment procedure, see Section 2.4, "Step 4. Deploy an Oracle Service Bus Configuration."

Note: If you have imported a configuration from a non-clustered environment and that configuration includes proxy services that use File, FTP, or Email transports, you must specify a Managed Server for each of those proxy services. The Managed Server list appears in the Oracle Service Bus Console in clustered Oracle Service Bus domains only.

4.6 Step 6. Update Your Domain as Your Production Environment Changes

Production environments change over time and as application use increases. This section describes how to update your domain in response to common production environment change scenarios:

- Section 4.6.1, "Adding a Managed Server"
- Section 4.6.2, "Deleting a Managed Server"
- Section 4.6.3, "Changing a Business Service in a Cluster"
- Section 4.6.4, "Installing a New Version of a Proxy Service in a Cluster"

4.6.1 Adding a Managed Server

As use of Oracle Service Bus grows, you can add new managed servers to your Oracle Service Bus cluster to increase capacity. For detailed instructions, see "Scaling the Topology" in the *Oracle Fusion Middleware High Availability Guide*.

After you scale your topology, use the instructions in the following topics to update business and proxy services.

4.6.1.1 Updating Business Service Configurations for an Expanded Cluster

If your Oracle Service Bus configuration includes one or more business services that use JMS request/response functionality, then you must also perform the following procedure using the Oracle Service Bus Console after adding the new managed server to the cluster:

1. In the Change Center, click Create to create a session.
2. Using the Project Explorer, locate and select a business service that uses JMS request/response. Business services of this type display Messaging Service as their Service Type.
3. At the bottom of the View Details page, click **Edit**.
4. If there is a cluster address in the endpoint URI, add the new server to the cluster address.
5. On the Edit a Business Service - Summary page, click **Save**.
6. Repeat the previous steps for each remaining business service that uses JMS request/response.
7. In the Change Center, click **Activate**.

The business services are now configured for operation in the extended domain.

Note: For business services that use a JMS MesageID correlation scheme, you must edit the connection factory settings to add an entry to the table mapping managed servers to queues. For information on how to configure queues and topic destinations, see "JMS Server Targeting" in *Oracle Fusion Middleware Configuring and Managing JMS for Oracle WebLogic Server*.

4.6.1.2 Updating Proxy Service Configurations for an Expanded Cluster

If your Oracle Service Bus configuration includes one or more proxy services that use JMS endpoints with cluster addresses, then you must also perform the following procedure using the Oracle Service Bus Console after adding the new managed server to the cluster:

1. In the Change Center, click Create to create a session.
2. Using the Project Explorer, locate and select a proxy service that uses JMS endpoints with cluster addresses.
3. At the bottom of the View Details page, click **Edit**.
4. If there is a cluster address in the endpoint URI, add the new server to the cluster address.
5. On the Edit a Proxy Service - Summary page, click **Save**.
6. Repeat the previous steps for each remaining proxy service that uses JMS endpoints with cluster addresses.
7. In the Change Center, click **Activate**.

The proxy services are now configured for operation in the extended domain.

4.6.2 Deleting a Managed Server

Using Oracle WebLogic Server administration tools, you can delete a managed server from your Oracle Service Bus cluster. Before deciding to delete a managed server, you should take into account the following considerations:

- If your Oracle Service Bus configuration includes one or more proxy services that use File, FTP, or Email transports that have pinned transport pollers to the managed server that you want to remove from the cluster, then you must select a different managed server for each of those proxy services *before* removing the managed server from the cluster.
- The managed server that hosts the Domain Singleton Marker Application must not be deleted from the cluster. If the managed server that hosts the Domain Singleton Marker Application fails, you must perform a manual migration.
- If you want to delete the managed server that hosts the Message Reporting Purger application, you must select a different manager server for the Message Reporting Purger and its associated queue (`wli.reporting.purge.queue`).

In the following procedure, you will delete resources associated with a managed server, and finally delete the managed server itself. The names of most affected resources end with *_auto_x*, where x is the number of the managed server you want to delete. For example, the JMS reporting provider queue for managed server 2 is called wli.reporting.jmsprovider.queue_auto_2.

Note: The Oracle WebLogic Server Console by default displays only a partial list of resources. To display a full list of resources, click **Customize this table** and increase the **Number of rows displayed per page**.

To delete a managed server from a cluster:

1. After you have read the considerations earlier in this section, stop all managed servers in the cluster. Keep the Admin server running.
2. Log into the Oracle WebLogic Server Console, and click **Lock & Edit** in the Change Center.
3. Delete all **_auto_x* queue members from distributed queues.
   a. In the Domain Structure pane, select **Services** > **Messaging** > **JMS Modules** to display a list of JMS modules.
   b. Click the **jmsResources** module. You will see a list of distributed queues whose names begin with *dist_*. For example, dist_QueueIn_auto.
   c. Click a distributed queue and go to its **Members** tab. Delete the queue for the managed server. For example, if you are deleting managed server 2, delete QueueIn_auto_2 from the dist_QueueIn_auto distributed queue.
   d. Repeat these steps for each distributed queue, deleting the relevant **_auto_x* resource for the managed server you are deleting.
4. Delete all **_auto_x* queues.
   a. In the list of resources for the **jmsResources** module, delete all queues whose names end with **_auto_x* for the managed server you are deleting. For example, delete wli.reporting.jmsprovider.queue_auto_2 if you are deleting managed server 2.
   b. Delete all **_auto_x* queues from the **Services** > **Messaging** > **JMS Modules** > **WseeJmsModule** for the managed server you are deleting. For example, delete DefaultCallbackQueue-WseeJmsServer_auto_2 and DefaultQueue-WseeJmsServer_auto_2 if you are deleting managed server 2.
5. Delete all JMS subdeployments for the managed server you are deleting.
   a. Select **Services** > **Messaging** > **JMS Modules**.
   b. Click the **configwiz-jms** module, and click the **Subdeployments** tab.

      In the Targets column, note the **_auto_x* targets. Delete all subdeployments targeted to **_auto_x* for the managed server you are deleting.
   c. As with the previous step, delete all **_auto_x* subdeployments from the **jmsResources** and **WseeJmsModule** modules for the managed server you are deleting.
6. Delete all JMS servers targeted to the managed server you are deleting.
   a. Select **Services** > **Messaging** > **JMS Servers**.
   b. Delete all **_auto_x* JMS servers for the managed server you are deleting.
7. Delete all store-and-forward agents targeted to the managed server you are deleting.
   a. Select **Services** > **Messaging** > **Store-and-Forward Agents**.

b. Delete all *_*auto_x* store-and-forward agents for the managed server you are deleting.

8. Delete all persistent stores targeted to the managed server you are deleting.
   a. Select **Services** > **Persistent Stores**.
   b. Delete all *_*auto_x* persistent stores for the managed server you are deleting.
9. Delete all migratable targets for the managed server you are deleting.
   a. Select **Environment** > **Migratable Targets**.
   b. Delete *server_name (migratable)* for the managed server you are deleting. For example, if the managed server you are deleting is named osb_server2, delete osb_server2 (migratable).
10. Untarget the managed server from the ALSB Framework Starter Application.
    a. In the Domain Structure pane, click **Deployments**.
    b. Click **ALSB Framework Starter Application**, and click the **Targets** tab.
    c. Select the **ALSB Framework Starter Application** check box, and click **Change Targets**.
    d. Remove the managed server as a deployment target.

    Note: If the **All servers in the cluster** targeting option is selected, you do not need to untarget the managed server.

11. Remove the managed server from the cluster.
    a. Select **Environment** > **Clusters**, and click the name of the cluster.
    b. Click the **Servers** tab, and delete the managed server from the cluster.
    c. Go to the **Configuration** > **General** tab for the cluster.

       In the **Cluster Address** field, remove the managed server from the cluster address.

       In the **Number Of Servers In Cluster Address** field, reduce the number of servers by one.
12. Delete the managed server. For instructions, see "Delete Managed Servers" in the *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Online Help*.
13. Restart the remaining managed servers.

4.6.3 Changing a Business Service in a Cluster

The procedure for changing a business service is the same in both non-clustered and cluster environments. For information about changing a business service, see Section 2.5.1, "Changing a Business Service."

However, the procedure for deploying changes to a business service in a cluster depends on the types of changes made to the business service and the nature of any other changes that might be deployed simultaneously. For more information, see the description of installation strategies in the following section.

4.6.4 Installing a New Version of a Proxy Service in a Cluster

As your business requirements change, you may need to make changes to your proxy services. You can make these changes dynamically online, partially offline, or completely offline. If your changes are backward compatible (that is, you are making no changes to interfaces), you can make your changes dynamically online using the Oracle Service Bus Console. Making other types of changes should be done partially or completely offline, which requires additional system administration steps.

For information about performing online updates, see Section 2.5.3, "Online Configuration Updates."

Making changes that include non-backward compatible changes to proxy service interfaces requires complete offline deployment. To install the new version, follow the procedure below while all servers are operational:

1. Quiesce all inbound messages.
2. Confirm all asynchronous backlogged messages have been processed.
3. Make the necessary changes in the proxy service, and test to verify the proxy service operates as required.
4. Resume accepting inbound messages.

For more information about backward compatibility and installation strategies, see Section 2.5.2, "Installing a New Version of a Proxy Service."

5

Understanding Oracle Service Bus High Availability

A clustered Oracle Service Bus domain provides high availability. A highly available deployment has recovery provisions in the event of hardware or network failures, and provides for the transfer of control to a backup component when a failure occurs.

The following sections describe clustering and high availability for a Oracle Service Bus deployment:

- Section 5.1, "About Oracle Service Bus High Availability"
- Section 5.2, "Oracle Service Bus Failure and Recovery"
- Section 5.3, "High Availability for Poller-Based Transports"

Note: For detailed instructions on configuring Oracle Service Bus for high availability, see the "Architecting OSB for High Availability and Whole Server Migration" document, available on the Oracle Service Bus product page at `http://www.oracle.com/technology/products/integration/service-bus/index.html`.

5.1 About Oracle Service Bus High Availability

For a cluster to provide high availability, it must be able to recover from service failures. Oracle WebLogic Server supports failover for clustered objects and services pinned to servers in a clustered environment. For information about how Oracle WebLogic Server handles such failover scenarios, see "Communications in a Cluster" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

5.1.1 Recommended Hardware and Software

The basic components of a highly available Oracle Service Bus environment include the following:

- An administration server
- A set of managed servers in a cluster
- An HTTP load balancer (router)
- Physically shared, highly-available disk subsystems for managed server data—Whole server migration requires that all data on a managed server be located on a multi-ported disk. A typical and recommended way to do this is by using a multi-ported disk subsystem or SAN, and allowing two or more servers to

mount file systems within the disk subsystem. The file system does not need to be simultaneously shared; it is only necessary for one server to mount a file system at any one time.

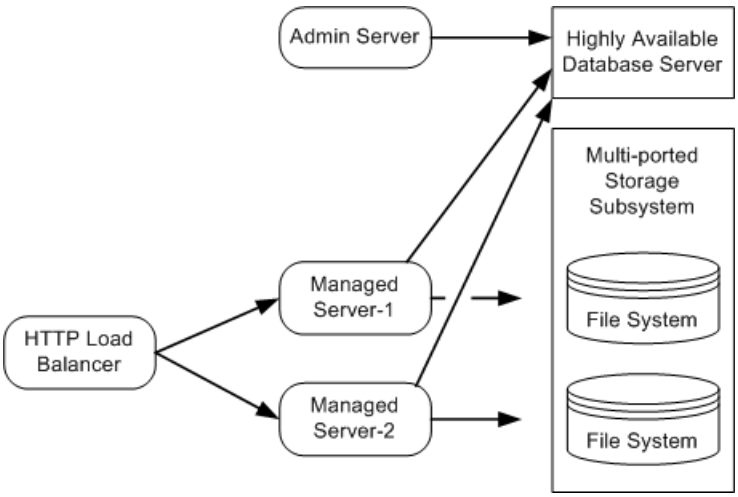
- A Microsoft SQL Server or Oracle database configured for failover with a cluster manager—You should take advantage of any high availability or failover solutions offered by your database vendor in addition to using a commercial cluster manager. (For database-specific information, see your database vendor's documentation.)

> **Note:** For information about availability and performance considerations associated with the various types of JDBC drivers, see "Configure Database Connectivity" in the *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Online Help*.

A full discussion of how to plan the network topology of your clustered system is beyond the scope of this section. For information about how to fully utilize inbound load balancing and failover features for your Oracle Service Bus configuration by organizing one or more Oracle WebLogic Server clusters in relation to load balancers, firewalls, and Web servers, see "Cluster Architectures" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*. For information on configuring outbound load balancing, see "Transport Configuration Page" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

For a simplified view of a cluster, showing the HTTP load balancer, highly available database and multi-ported file system, see the following figure.

Figure 5–1 Simplified View of a Cluster



5.1.1.1 Regarding JMS File Stores

The default Oracle Service Bus domain configuration uses a file store for JMS persistence to store collected metrics for monitoring purposes and alerts. The configuration shown relies on a highly available multi-ported disk that can be shared between managed servers to optimize performance. This will typically perform better than a JDBC store.

For information about configuring JMS file stores, see "Using the WebLogic Persistent Store" in *Oracle Fusion Middleware Configuring Server Environments for Oracle WebLogic Server*.

5.1.2 What Happens When a Server Fails

A server can fail due to software or hardware problems. The following sections describe the processes that occur automatically in each case and the manual steps that must be taken in these situations.

5.1.2.1 Software Faults

If a software fault occurs, the Node Manager (if configured to do so) will restart the Oracle WebLogic Server. For more information about Node Manager, see "Using Node Manager to Control Servers" in *Oracle Fusion Middleware Node Manager Administrator's Guide for Oracle WebLogic Server*. For information about how to prepare to recover a secure installation, see "Directory and File Back Ups for Failure Recovery" in *Oracle Fusion Middleware Managing Server Startup and Shutdown for Oracle WebLogic Server*.

5.1.2.2 Hardware Faults

If a hardware fault occurs, the physical machine may need to be repaired and could be out of operation for an extended period. In this case, the following events occur:

- The HTTP load balancer will detect the failed server and will redirect to other managed servers. (The actual algorithm for doing this will depend on the vendor for the http load balancer.)
- All new internal requests will be redirected to other managed servers.
- All in-flight transactions on the failed server are terminated.
- JMS messages that are already enqueued are not automatically migrated, but must be manually migrated. For more information, see Section 5.1.2.3, "Server Migration."
- If your Oracle Service Bus configuration includes one or more proxy services that use File, FTP or Email transports with transport pollers pinned to a managed server that failed, you must select a different managed server in the definition of each of those proxy services in order to resume normal operation.
- If the managed server that hosts the aggregator application fails, collected metrics for monitoring and alerts are not automatically migrated. You must perform a manual migration. For more information, see Section 5.1.2.3, "Server Migration."

5.1.2.3 Server Migration

Oracle Service Bus leverages Oracle WebLogic Server's whole server migration functionality to enable transparent failover of managed servers from one system to another. For detailed information regarding Oracle WebLogic Server whole server migration, see the following topics in the Oracle WebLogic Server documentation set:

- "Failover and Replication in a Cluster" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*
- "Avoiding and Recovering from Server Failure" in *Oracle Fusion Middleware Managing Server Startup and Shutdown for Oracle WebLogic Server*.

5.1.2.4 Message Reporting Purger

The Message Reporting Purger for the JMS Message Reporting Provider is deployed in a single managed server in a cluster (see Appendix B, "Oracle Service Bus Deployment Resources").

If the managed server that hosts the Message Reporting Purger application fails, you must select a different managed server for the Message Reporting Purger and its associated queue (`wli.reporting.purge.queue`) to resume normal operation.

Any pending purging requests in the managed server that failed are not automatically migrated. You must perform a manual migration. Otherwise, target the Message Reporting Purger application and its queue to a different managed server and send the purging request again.

5.2 Oracle Service Bus Failure and Recovery

In addition to the high availability features of Oracle WebLogic Server, Oracle Service Bus has failure and recovery characteristics that are based on the implementation and configuration of your Oracle Service Bus solution. The following sections discuss specific Oracle Service Bus failure and recovery topics:

- Section 5.2.1, "Transparent Server Reconnection"
- Section 5.2.2, "EIS Instance Failover"

5.2.1 Transparent Server Reconnection

Oracle Service Bus provides transparent reconnection to external servers and services when they fail and restart. If Oracle Service Bus sends a message to a destination while the connection is unavailable, you may see one or more runtime error messages in the server console.

Transparent reconnection is provided for the following types of servers and services:

- SMTP
- JMS
- FTP
- DBMS
- Business services

Oracle Service Bus Console also provides monitoring features that enable you to view the status of services and to establish a system of SLAs and alerts to respond to service failures. For more information, see "Monitoring" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

5.2.2 EIS Instance Failover

Most business services in production environments will be configured to point to a number of EIS instances for load balancing purposes and high availability. If you expect that an EIS instance failure will have an extended duration or a business service points to a single, failed EIS instance, you can reconfigure the business service to point at an alternate, operational EIS instance. This change can be made dynamically.

For information about using the Oracle Service Bus Console to change an endpoint URI for a business service, see "Transport Configuration Page" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

5.3 High Availability for Poller-Based Transports

An additional JMS-based framework has been created to allow the poller-based File, FTP, and Email transports to recover from failure. These transports use the JMS

framework to ensure that the processing of a message is done at least once. However, if the processing is done, but the server crashes or the server is restarted before the transaction is complete, the same file may be processed again. The number of retires depends on the redelivery limit that is set for the poller transport for the domain.

New messages from the target (a directory in case of File and FTP transports and server account in case of Email transport) are copied to the download (stage) directory at the time of polling or pipeline processing.

> **Note:** For FTP transport, a file is renamed as `<name>.stage` in the remote directory. It is copied to the stage directory only at the time of pipeline processing,

For File and FTP transports, a JMS task is created corresponding to each new file in the target directory. For Email transport, an e-mail message is stored in a file in the download directory and a JMS task is created corresponding to each of these files.

These JMS task messages are enqueued to a JMS queue which is pre-configured for these transports when the Oracle Service Bus domain is created.

5.3.1 JMS Queues

The following poller-transport-specific JMS queues are configured for Oracle Service Bus domains:

Table 5–1 JMS Queues Configured for Oracle Service Bus Domains

Transport Name	JMS Queue Name
FTP	wlsb.internal.transport.task.queue.ftp
File	wlsb.internal.transport.task.queue.file
Email	wlsb.internal.transport.task.queue.email

A domain-wide message-driven bean (MDB) receives the JMS task. Once the MDB receives the message, it invokes the pipeline in an XA transaction. If the message processing fails in the pipeline due to an exception in the pipeline or server crash, the XA transaction also fails and the message is again enqueued to the JMS queue. This message is re-delivered to the MDB based on the redelivery limit parameter set with the queue. By default, the redelivery limit is `1` (the message is sent once and retried once). If the redelivery limit is exhausted without successfully delivering the message, the message is moved to the error directory. You can change this limit from Oracle WebLogic Server Console. For more information, see "JMS Topic: Configuration: Delivery Failure" in the *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Online Help*.

> **Note:** For a single Oracle Service Bus domain transport, the redelivery limit value is global across the domain. For example, within a domain, it is not possible to have an FTP proxy with a redelivery limit of `2` and another FTP proxy with a redelivery limit of `5`.

5.3.2 High Availability in Clusters

For clusters, the JMS queue associated with each of these poller based transport is a distributed queue (each Managed Server has a local JMS queue, which is a member of

the distributed queue). The JMS queue for a transport is domain-wide. The task message is enqueued to the distributed queue, which is passed on to the underlying local queues on the Managed Server. The MDB deployed on the Managed Server picks up the message and then invokes the pipeline in a transaction for actual message processing.

Since the JMS queues are distributed queues in cluster domains, high availability is achieved by utilizing the Oracle WebLogic Server distributed queue functionality. Instead of all Managed Servers polling for messages, only one of the Managed Servers in a cluster is assigned the job of polling the messages. At the time of proxy service configuration, one of the Managed Servers is configured to poll for new files or e-mail messages.

The poller server polls for new messages and puts them to the uniform distributed queue associated with the respective poller transport. From this queue, the message is passed on the local queue on the Managed Server. The Managed Servers receive the messages through MDBs deployed on all the servers through these local queues.

Note: There is a uniform distributed queue with a local queue on all the Managed Servers for each of these poller based transports.

If the managed servers crashes after the distributed queue delivers the message to the local queue, you need to do manual migration. For more information, see Section 5.1.2.3, "Server Migration."

When a cluster is created, the uniform distributed queue is created with local queue members - on all the Managed Servers. However, when a new Managed Server is added to an existing cluster, these local queues are not automatically created. You have to manually create the local queues and make them a part of a uniform distributed queue.

To create a local queue:

1. Create a JMS Server and target it to the newly created Managed Server.
2. Create a local JMS queue, set the redelivery count, and target it to the new JMS server.
3. Add this local JMS queue as a member of the uniform distributed queue associated with the transport.

Note: The JNDI name of the distributed queue is `wlsb.internal.transport.task.queue.file` (for File transport), `wlsb.internal.transport.task.queue.ftp` (for FTP transport) and `wlsb.internal.transport.task.queue.email` (for Email transport).

5.3.2.1 Load Balancing

Since we use distributed JMS queues, messages are distributed to the Managed Servers based on the load balancing algorithm associated with the distributed queue. By default, the JMS framework uses round-robin load balancing. You can change the algorithm using the JMS module in Oracle WebLogic Server Console. For more information, see "Load Balancing for JMS" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*. If one of the Managed Servers fails, the remaining messages are processed by any of the remaining active Managed Servers.

Note: The poller server should always be running. If the poller server fails, the message processing will also stop.

6

Best Practices for Deploying Oracle Service Bus Resources

Oracle Service Bus has powerful capabilities to automate the deployment of resources to production systems and also provides complete support for various deployment use cases.

This document describes the best practices and procedures that may be used while deploying Oracle Service Bus. It is targeted towards deployers or administrators who plan out the deployment and topology rather than developers who write automated scripts or programs for deployment. For more information, see Appendix A, "Using the Deployment APIs." To write the scripts or programs, developers can refer to the *Oracle Fusion Middleware Java API Reference for Oracle Service Bus*.

This chapter contains the following sections:

6.1 Deployment Topologies

The following sections describe typical deployment topologies.

6.1.1 Development and QA Systems

There may be multiple development systems for a given production system. For example, each department for an enterprise ESB production system might have a separate project with a development system for that project's team. The development system typically has a QA system associated with it. The two systems are related one is to one.

6.1.2 Stage and Production Systems

Typically, there is a stage system associated with the production system. These two systems are related one is to one. The stage system mirrors the production system as

much as possible. Typically, the relationship of the development system (or QA system) to production systems (or stage systems) is many is to one.

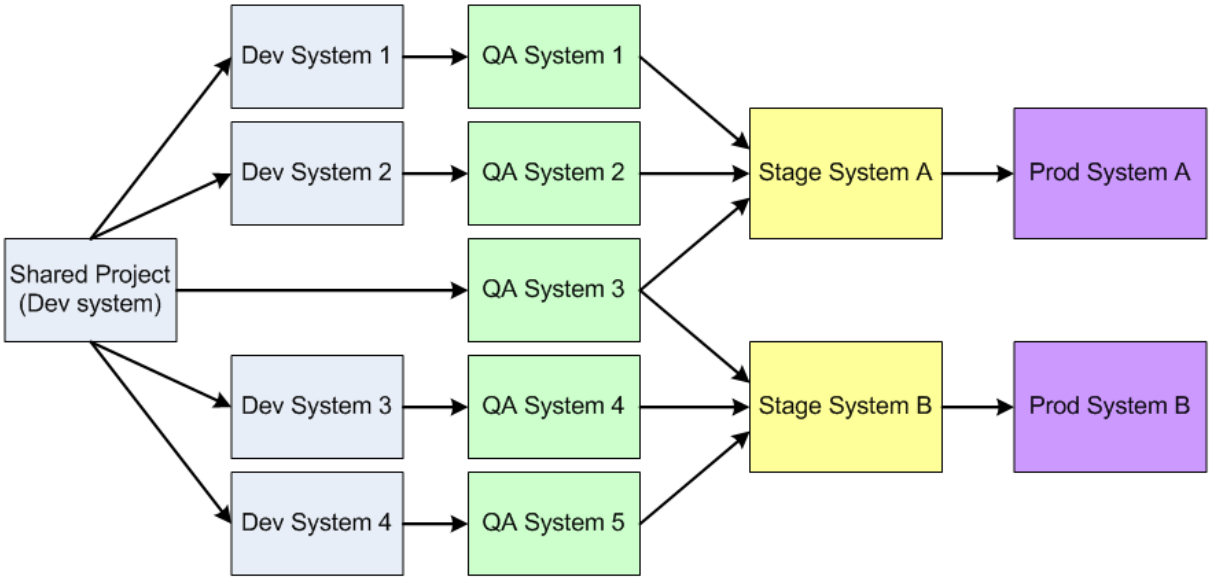
6.1.3 Shared Projects

In more complex deployment topologies (a development system for each department in a large organization) corporate-wide standards like schemas, transformation and shared services may be in a shared project. This scenario can be handled by having a separate development system for the shared project and providing a read-only copy of the shared project in each departmental development system. When any shared resource changes, all the departmental systems are updated by importing and exporting the required resources.

When deploying resources into a stage system or a production system, the shared project along with all the impacted departmental projects should be deployed together to avoid any semantic errors that might cause the deployment to fail.

Sometimes, a shared project may be shared across multiple production systems. For example, when there are separate production systems for each department that are interconnected into a corporate ESB network, corporate-wide assets like schemas may be shared across all the production systems.

Figure 6–1 Complex Topology



6.2 Types of Deployment

The typical unit of deployment is called a project. There are two types of project deployment:

- **Complete** - The entire project is deployed
- **Incremental** - Only changes to the project since the last deployment or some explicitly selected subset of resources are deployed.

  Incremental deployment is additive - only new and updated resources are deployed on the target system. Note that any resource that is deleted in the project is not deleted from the target system. However, deleted project resources are also deleted from the target system during a complete project deployment.

Note: This behavior may be customized at import time. For example, alert destinations are defined in the production system by the production system operator, but these alerts do not exist in the development system. In such a scenario, you may not want to delete these alert destinations when you do a complete deployment of the project.

This document focuses on these basic types of deployment. Use cases involving shared projects are extensions of the basic use cases. In addition, this document also focuses on the use cases where export and import of resources is automated by scripts or programs.

6.3 Deployment Roles

Export, import and environmental customizations may be done by a deployer, operator or Administrator depending on the system and depending on the enterprise's policy.

Export from the development system is typically done by a deployer. The Administrator, operator, or deployer may be responsible for exporting and importing resources from the stage system to the production system.

Exporting and importing resources can be done using Oracle Service Bus Console or via a script or program (which can be written by a developer).

If an operator is responsible for exporting resources from a system, a pre-defined automated script or program can be executed to export either the complete project or specific resources in the project. Similarly, if an operator is responsible for importing resources into a system, a pre-defined automated script or program can be executed to do the import.

6.4 Customization Files

An Administrator uses customization files to make changes to environment values as well as to change references within resources. Customization files can include customizations for all the environment values found in the selected resources, including complex environment values types defined inside the EnvValueTypesclass. In addition, it includes a reference customization type for changing resource references inside resources with dependencies.

The customization schema (Customization.xsd) which describes the customization types is available at the following location in your Oracle Service Bus installation:

OSB_ORACLE_HOME\modules\com.bea.common.configfwk_*version*.jar

You can create sample customization files from Oracle Service Bus Console. The scope of a customization file can be a project or individual resources in a project. For more information, see "Creating Customization Files" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

The created sample customization file may be used as a starting point for making desired modifications by specifying the actual values for an environment during the export or import process.

6.4.1 Substituting Environment Values

As part of deployment, environment values in resources in source systems must be changed as part of the export process or the import process to reflect the values that are application in the target system.

Environment values are certain pre-defined fields in the configuration data whose values are very likely to change when you move your configuration from one domain to another (for example, from test to production). Environment values represent entities such as URLs, URIs, file and directory names, server names, e-mails, and such. Also, environment values can be found in Alert Destinations, proxy services, business services, SMTP Server and JNDI Provider resources, and UDDI Registry entries.

Certain environment values are complex XML objects that cannot be found and replaced using the Find and Replace option from Oracle Service Bus Console. However, you can still set these environment values directly by using the `ALSBConfigurationMBean` from a script. For detailed information about `ALSBConfigurationMBean`, see the *Oracle Fusion Middleware Java API Reference for Oracle Service Bus*. In addition to setting them through the API, you can set complex type environment values using customization files.

Customization files are XML files and you can open these files in any editor and substitute the required environment values. In addition, you can search for specific environment values (that are not complex XML types) in Oracle Service Bus Console or in a customization file and replace them with the new values. You can fine-tune the scope of the search by filtering these environment values based on variable type or project. For more information, see "Finding and Replacing Environment Values" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

Note: After you substitute the environment values, you must execute these customization files. For more information, see "Executing Customization Files" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

You can substitute environment values as part of the import or export process.

6.5 Exporting Resources

The exporter needs to know if incremental deployment is feasible. If so, a developer can write a script that can find all resources that have been changed or created after a specific time frame (for example, after the last deployment) and export them.

If the exporter uses Oracle Service Bus Console for exporting only selected resources (for incremental deployment), the exporter can look at the last modified timestamp of the resources and select the only those resources that need to be exported. For more information, see "Exporting Resources" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

6.5.1 Customizing Resources During Export

You can customize resources (including substituting environment values) while importing or exporting resources. When you export resources:

1. Start a session.
2. Customize the resources and execute the customization file.
3. Export the required resources.

4. Discard the session to ensure that the original values are retained at the source system. Since the exported file contains the updated values, these values are reflected at the target system.

6.6 Importing Resources

You can import resources from Oracle Service Bus Console or using a script. For information about importing resources using the console, see "Importing Resources" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

When a script is used, the script stages the import JAR. It then obtains a deployment plan from the system for the staged import JAR. The deployment plan should contain all resources in the project and if these resources should be created, deleted, updated or left untouched in the target system. The deployment plan is typically customized by the script to not overlay resources that are environment specific (like service account) or overlay or delete operator controlled resources.

6.6.1 Customizing Resources During Import

You can customize resources (including substituting environment values) while importing or exporting resources. When you import resources:

1. Start a session.
2. Import the required resources.
3. Customize the resources and execute the customization files.
4. Activate the session.

On import, you can preserve environment variables and ensure that only environment values of resources that are created on import are customized by the customization file.

6.6.2 Importing Environment-specific Resources

Resources like service accounts, service key providers, SMTP server, JNDI server, and UDDI server are fundamentally environment specific and under the control of administrators because they contain credentials. You need to define these resources in the target system and avoid moving such resources across environments

Even in a complete project deployment, such resources should not be updated in the target system. If there is a reference in the import file to an environment-specific resource that does not exist on the target system, the suggested best practice is to fail the import and redo the import after the administrator defines these resources.

An alternative strategy sometimes followed is to import these resources but enable the option to not overlay credentials during import (preserve credentials). After importing the very first time, the administrator can change the credentials manually to the right values.

Sometimes, there might be cases where the name of the same environment resource (like an SMTP server) is different across environments. This can be handled by the import script by mapping the reference to such a resource as part of the import process.

6.6.3 Importing Operational Values

There are two types of operational values in Oracle Service Bus, global Operational Settings and operational values and alerts for proxy and business services. Global Operational Settings is a resource located under the System project folder. The global Operational Settings resource can be exported like any other resource. On import, this resource can only be updated; it cannot be created or deleted.

On import, you can preserve the operational values in the target system by selecting the **Preserve Operational Values** option in Oracle Service Bus Console.

Note: Sometimes, operators define alert destinations for SLA alerts. Such resources are operator controlled and should not be overlaid or deleted during deployment.

6.6.4 Importing Access Control Policies

Typically the access control policies in each environment are different. In this case the user ensures the access policies are not lost during import by specifying the option to not overlay access control policies during import. However, sometimes the correct access control policies are set in the staging environment and apply to the production environment. In such cases, the access control policies are imported/exported along with the service.

6.6.5 Deploying Oracle WebLogic Server Artifacts

Ideally, deployment should fail if Oracle WebLogic Server artifacts required for completing the configuration are missing.

While some are checked during import, others are not checked (for example, request queue for a business service) and a runtime error occurs.

The best practice is to have the deployment script retrieve environment values for deployed services and alert destinations and check with Oracle WebLogic Server to see if the specified resource exists in Oracle WebLogic Server. If not the deployment fails and the administrator has to define these resources before the script is re-executed by the operator.

A more limited alternative is to import the artifacts into a session and look for semantic errors. Semantic errors occur if Oracle Service Bus checks for the existence of a Oracle WebLogic Server resource at design time.

6.7 Summary of Deployment Best Practices

This is a summary of recommended best practices for deployment of Oracle Service Bus resources.

- Avoid project renames. If projects need to be renamed, do so concurrently across development, QA, stage and production systems before the next deployment.
- Operations on environment specific resources are best filtered out at import time. Administrators may define these resources (referenced in the import file) in the target system before starting the import process. Alternatively, new environment-specific resources may be deployed and customized for the environment after the import.

- Operational resources controlled by the operator should not be impacted by the import. A naming convention or a dedicated folder could be used to identify such resources during import.
- When the customization file contains the customizations of all project resources in a single file, apply customizations only to resources that are imported. An alternative is to preserve environment values on import and only apply customizations to resources that are added during import.
- At export-time, the exporter needs to know if resources have been deleted, renamed, or moved; and if there have been project reorganizations of artifacts since the last export. If yes, complete deployment should be done. If no, the exporter can select only those resources that need to be exported and an incremental deployment can be done.

6.8 UDDI

There are special considerations to factor in when working with services deployed in UDDI registries. This section provides information about the proxy services and business services that are published to or imported from UDDI. The use of UDDI production registries complicates import into the production system and special care has to be taken.

6.8.1 UDDI Deployment Topologies

Based on your environment, you can have any of the following UDDI deployment topologies:

- Section 6.8.1.1, "Development-Only Registry"
- Section 6.8.1.2, "Production-Only Registry"
- Section 6.8.1.3, "Development and Production Registry"
- Section 6.8.1.4, "Registry Per Individual Domain"

6.8.1.1 Development-Only Registry

The simplest deployment of UDDI is having a single development (design) time registry where the resources are both published and discovered. This registry is used for governance using approval control. However, you can also have separate design time publish registry and discovery registry. After the design time publish registry is approved, it can be promoted to the discovery registry.

6.8.1.2 Production-Only Registry

In this production time registry topology, a single production UDDI registry contains all the production business services and proxy services, and their locations (URL). The production registry can be used to discover all production services. However, you can also have separate production time publish registry and discovery registry. After the production time publish registry is approved, it can be promoted to the production discovery registry.

A key reason for having a production registry is dynamic endpoint management. You can change the business service location in the UDDI registry and have it automatically reflected in the Oracle Service Bus production system by enabling automatic synchronization of resources. For more information, see "Auto-Synchronization of Services With UDDI" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

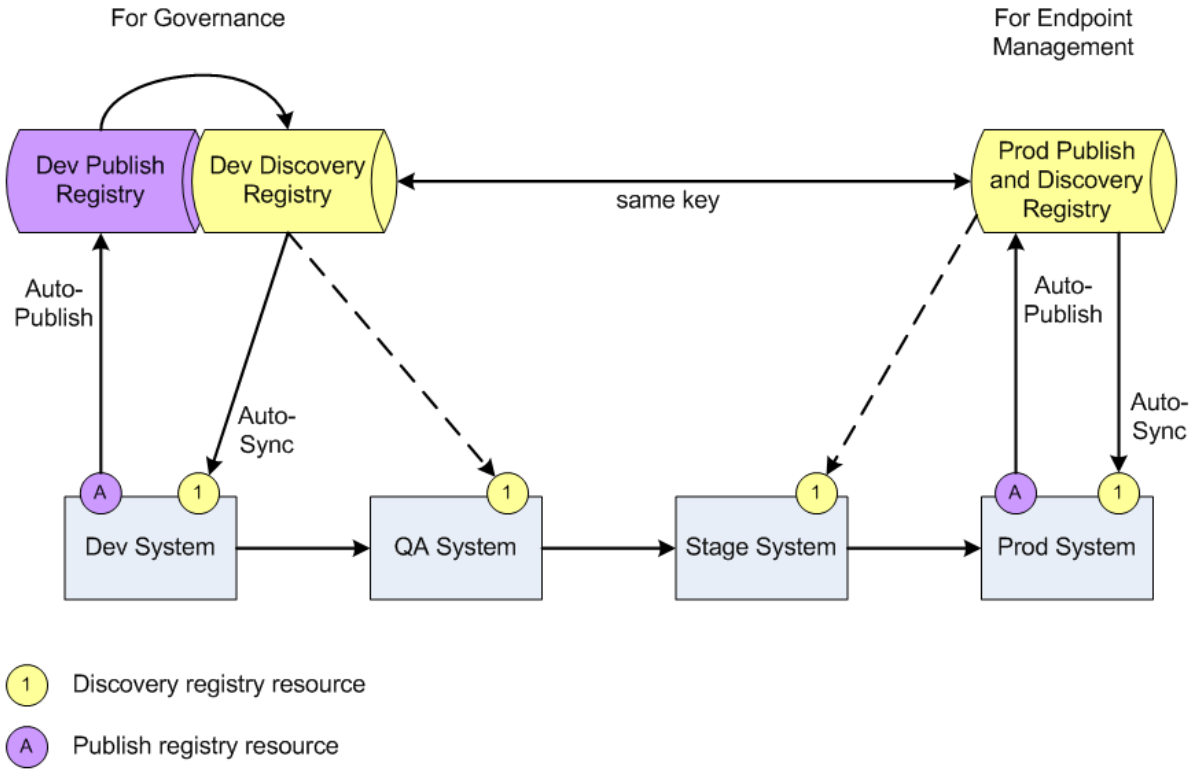
Typically, proxy services are automatically published to the publish registry. If the approval step in UDDI results in a reject of the service, the UDDI approver manually notifies the developer and the developer has to make the appropriate changes and re-export the services through the stage and production systems. For more information, see "Using Auto Publish" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

Production business services are imported from the production discovery registry. Automatic synchronization of the resources can be enabled to ensure that changes in the UDDI registry are automatically updated in Oracle Service Bus. However, when you manually import business services from the production registry into the development registry, you must change the environment values to conform to the values in the development system. The development business service is not automatically synchronized with the UDDI registry because the development environment values would most probably be different from the production registry.

6.8.1.3 Development and Production Registry

An alternative approach is to have a separate development and production UDDI registry. In this topology, the approval step can take place in the development registry. So, the extra development UDDI registry ensures that approvals are done earlier in the cycle.

Figure 6–2 Complex Topology - UDDI



6.8.1.4 Registry Per Individual Domain

Sometimes, the proxy service does a dynamic search of the UDDI registry with a POJO callout to select the service meeting the desired search criteria and retrieve the URL for dynamic routing. In this scenario, a dummy service is defined in Oracle Service Bus with a dummy URL and the URL is dynamically replaced with the actual value after

the lookup. In this scenario, a UDDI registry is needed for each environment. Also, the dummy business service in Oracle Service Bus is not linked to any UDDI service.

There is an impact on performance and availability in this scenario. So, the preferred approach is to enable automatic synchronization of a business service into Oracle Service Bus with a routing table XQuery resource that can be used for dynamic routing. For more information, see "Using Dynamic Routing" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

6.8.2 Summary of UDDI Deployment Best Practices

This is a summary of recommended best practices for working with services deployed in UDDI registries.

- Organize business services associated with a particular UDDI registry in a specific folder to make it easy to identify these resources during import.
- Use the same UDDI server resource name in all systems using that UDDI registry. When there is a separate development and production UDDI registry, use the same resource name for the development and production instances of the UDDI registry. This ensures that references to the server by services are automatically resolved during import.
- Preserve the same service key across the two registries when you use a development and production UDDI registry.

  Typically, the automatic publish option is enabled for production proxy services and the automatic synchronization option is enabled for business services. When the service is manually imported from the UDDI registry into the development system, the service still has information about the UDDI key of the source registry and a reference to the registry resource. The development service is imported into stage or QA systems in the normal way. The service in the stage and QA systems also has the UDDI key and registry reference to the source registry of the service. So, ignore any UDDI-related status messages in these systems that prompt you that the service is out of sync or that the service should to be deleted.

  The same service in development and production may not have the same keys when two registries are used. However, the recommended approach is to preserve the same keys.
- Create a new UDDI service with the new shape when the shape of a business service changes.

6.8.3 Importing and Exporting Resources Between Multiple Systems

You can import or export UDDI related resources available in Oracle Service Bus from one system to another system. This section provides information you need to take into account in the various UDDI deployment topologies. In all topologies, all systems have the same UDDI registry resources configured. However, auto synchronization is not enabled in the stage and QA systems.

There are a few factors to consider when a production UDDI registry is used:

- UDDI contains only a subset of the information for a service. So, some part of the information related to the service comes to the production system through the normal import export process while other parts of the service information comes into the production system from synchronization with the UDDI registry. This information has to be carefully managed during import.

- Ideally, when a service comes into the development system from a design time UDDI registry, there has to be some way to identify an equivalent service in the production UDDI registry. This ensures that at import time, the information from that service can be merged into the equivalent service in the production UDDI registry.

6.8.3.1 Development-Only Registry

In this scenario, when you move one system to another system, you need to do the following:

1. Detach the business service from the UDDI registry.
2. Disable the auto publish option for the proxy service. You can do this using the MBean API or do this from Oracle Service Bus Console while configuring the proxy service.
3. If required, make any changes to the service.
4. Manually export the resource from the current system to a JAR and import it to the next system.

6.8.3.2 Production-Only and Development and Production Registry

When importing resources into the production system, there can be any one of the following scenarios:

- Importing a new service into a system, and the UDDI key of the service in the import JAR and the UDDI registry are the same.
- Importing a new service into a system, and the UDDI key of the service in the import JAR and the UDDI registry are the different.
- Importing an existing service into a system

Importing a new service into a production system, and the UDDI key of the service in the import JAR and the UDDI registry are the same

In this scenario, import the service into the system and do one of the following:

- Use the import script to explicitly force a UDDI re-synchronization of the service in the current session with an API call.
- Mark the imported service for background re-synchronization by Oracle Service Bus.

  The disadvantage of this option is that UDDI-related service fields may be incorrect until you complete the re-synchronization. However, if auto synchronization is enabled, this is a small interval.

Importing a new service into a production system, and the UDDI key of the service in the import JAR and the UDDI registry are the different

In this scenario, import the service into the system and do one of the following:

- In an import script, the user explicitly forces a UDDI resync of the service in the current session with an API call. The service is shown as "sync pending" after the session is activated.
- Mark the service for deletion because the matching UDDI service was not found in the UDDI registry. Customize the resource appropriately and then execute the customization file. Later, use Oracle Service Bus Console to detach the service from the UDDI registry.

The disadvantage of this approach is that besides UDDI, the production environment values must also be present in the customization file.

After detaching the service from the registry, you can download the UDDI-related information from the UDDI registry and then enable automatic synchronization, if required.

Importing an existing service into a production system

In this scenario, do one of the following:

- Overlay the production UDDI service during import when a UDDI registry (or identical UDDI registries - in terms of service key) is used. You can then force a UDDI re-synchronization of the service in the current session with an API call.
- Use the import script to skip importing this service to the production system. This is the only option if the UDDI production keys are different.

  However, when you skip importing the service, non-UDDI sourced data in the service may be wrong. So, you must directly update such information in the production service definition instead of importing the latest service.

A

Using the Deployment APIs

You can use the Oracle Service Bus MBeans in Java programs and WLST scripts to automate promotion of Oracle Service Bus configurations from development environments through testing, staging, and finally to production environments. The Oracle Service Bus MBeans you can use to programmatically perform deployment operations include:

- **SessionManagementMBean**: used to create, activate and discard a session, or to return the name of an existing session
- **ALSBConfigurationMBean**: used to import and export Oracle Service Bus configurations, update environment-specific information (endpoint URIs, etc.), query Oracle Service Bus configurations and resources.

The latter MBeans are interfaces in the com.bea.wli.sb.management.configuration package.

Numerous customization options can be applied during deployment. An extended list of environment variables allows you to preserve or tailor settings when moving from one environment to another.

This section contains the following topics:

> **Tip:** Oracle Service Bus APIs are documented in *Oracle Fusion Middleware Java API Reference for Oracle Service Bus*.

A.1 Managing Sessions Using Programs and Scripts

Oracle Service Bus sessions allow different users to update discrete parts of configuration data without interfering with each other. A session is essentially a named sandbox, in which your changes are abstracted from other users, as well as from the core data (the data on which Oracle Service Bus runs), until the changes are activated. In order to modify resources and Oracle Service Bus configurations, you must create a session and perform changes in that session. The changes are only reflected in the core data when you activate the session. You can create multiple sessions as long as no two sessions have the same name. A session can only be activated using the instance of the `SessionManagementMBean` that works on that session data.

Each MBean type, except for `SessionManagementMBean`, has one instance per session. When a session is created, a new set of MBean instances (one for each MBean Type) is created automatically. One instance of each MBean Type operates on the core data that is saved to the Oracle Service Bus data cache. MBean instances created for a

session are destroyed when the session is discarded or activated. MBean instances that operate on core data, however, are never destroyed. MBean instances that work on core data do not support update operations.

A.1.1 Creating, Activating, Discarding, and Locating Sessions

Oracle Service Bus sessions are created using the Oracle Service Bus Console. The methods in the `SessionManagementMBean` interface directly parallel the interactive features provided in the Oracle Service Bus Console, and require execution in the same order as their GUI counterparts. The following table lists the methods available in the `SessionManagementMBean` interface and the tasks they perform.

Table A–1 Session Management Methods

To...	Use...
Activate a session	`activateSession(String session, String description)`
Create a new session with a user-specified name.	`createSession(String session)`
Delete the session without activating changes	`discardSession(String session)`
Return true if a session with the given session name exists	`sessionExists(String session)`

For reference material on the `SessionManagementMBean` interface and Java usage examples, as well as sample code describing how to use MBeans from a Java client and in a script, see the SessionManagementMBean Interface in the `com.bea.wli.sb.management.configuration` package in the *Oracle Fusion Middleware Java API Reference for Oracle Service Bus*.

A.1.1.1 Examples

The SessionManagementMBean Interface in the `com.bea.wli.sb.management.configuration` package in the *Oracle Fusion Middleware Java API Reference for Oracle Service Bus* includes example code illustrating how to create a session, how to obtain SessionManagementMBean for creating a session, and ALSBConfigurationMBean for operating on the session that is created and on core data, and so on.

A.2 Managing Configuration Tasks Using Programs and Scripts

The Oracle Service Bus `ALSBConfigurationMBean` allows you to programmatically query, export and import resources, obtain validation errors, get and set environment values, and in general manage resource configuration in an Oracle Service Bus domain. Oracle Service Bus configurations are packaged as simple JARs containing Oracle Service Bus resources such as proxy services, WSDLs, and business services. These resources can span multiple projects, or contain only partial configuration information. For example, you can export only a subset of a project, a whole project, or subsets of resources from many projects.

The following sections describe how to use the `ALSBConfigurationMBean` to perform these deployment activities from a Java client:

- Section A.2.1, "Importing, Exporting, and Querying Configurations"
- Section A.2.2, "Updating Environment-Specific Information"

For reference material on the `ALSBConfigurationMBean` interface, see the ALSBConfigurationMBean Interface in the `com.bea.wli.sb.management.configuration` package in the *Oracle Fusion Middleware Java API Reference for Oracle Service Bus*.

A.2.1 Importing, Exporting, and Querying Configurations

Oracle Service Bus configurations are created using the Oracle Service Bus Console, and are stored through export in `.jar` files. After a configuration `.jar` file has been exported, you can promote the configuration by importing it into a different Oracle Service Bus domain and changing the environment-specific values in the configuration to match those of the new environment.

The methods in the ALSBConfigurationMBean Interface allow you to manage resources in an Oracle Service Bus domain, including tasks such as:

- Query, export, and import resources (includes importing resources from a zip file, and exporting resources at the project level)
- Get and set environment values
- Clone a project, folder, or resource with a new identity
- Modify the existing references from all the resources in the given list to a new set of references
- Customize multiple properties at once
- Obtain validation errors

See the ALSBConfigurationMBean in the *Oracle Fusion Middleware Java API Reference for Oracle Service Bus* for the comprehensive method summary for the `ALSBConfigurationMBean` methods.

A.2.2 Updating Environment-Specific Information

The methods in ALSBConfigurationMBean and the Customization Class allow you to update environment specific information. This includes.

- Updating the value of endpoints in proxy and business service configurations
- Updating the directory elements in File, E-mail, and FTP transport configurations
- Directly setting environment value(s)
- Searching for environment-specific values specified in a query
- Find environment values specified in a query, and replace all occurrences of the environment value pattern with the given parameter.

See ALSBConfigurationMBean and Customization Class in the *Oracle Fusion Middleware Java API Reference for Oracle Service Bus* for a comprehensive summary of methods. Import customizations are supported by the **ALSBImportPlan**.

You must update your security configuration and all other environment-specific settings interactively using the Oracle Service Bus Console. For information on configuring environment-specific settings, see Section 2.4, "Step 4. Deploy an Oracle Service Bus Configuration."

A.2.2.1 Examples

The ALSBConfigurationMBean Interface in the `com.bea.wli.sb.management.configuration` package in the *Oracle Fusion*

Middleware Java API Reference for Oracle Service Bus includes example code illustrating how to import and export Oracle Service Bus configurations, how to change environment values, how to query resources, and so on.

A.2.3 Related Topics

"Oracle Service Bus APIs" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Service Bus*.

B

Oracle Service Bus Deployment Resources

This section describes Oracle Service Bus deployment resources that are added to your domain when you extend your domain with the Oracle Service Bus domain extension template.

The Oracle Service Bus Console and the UDDI manager run on the administration server. Therefore, you must run an administration server to manage Oracle Service Bus and to facilitate publishing and importing to and from a UDDI registry.

B.1 Oracle Service Bus Domain Extension Template

Using the Oracle Fusion Middleware Configuration Wizard or WLST, you can easily extend a base Oracle WebLogic Server domain to create an Oracle Service Bus domain. You accomplish this by adding the resources and services provided in the Oracle Service Bus extension template to a base Oracle WebLogic Server domain.

> **Note:** Using the Oracle Fusion Middleware Configuration Wizard in graphical mode, you can easily create a new Oracle Service Bus domain by checking the Oracle Service Bus check box in the Select Domain Source window. The result is the same as creating a base Oracle WebLogic Server domain first and then extending that domain with the Oracle Service Bus extension template. For more information about the templates required to create an Oracle Service Bus domain, see "Relationships Between Templates" in the *Oracle Fusion Middleware Domain Template Reference*.

B.1.1 Generated Domain Output

The following table defines the default directory structure and files generated after applying the Oracle Service Bus extension template to a base Oracle WebLogic Server domain. Unless otherwise specified, by default, the Oracle Fusion Middleware Configuration Wizard creates the domain in the *MW_HOME*\user_projects\domains\base_domain directory. If you modify the default configuration settings, the output directory structure may be different from the structure described here.

Table B–1 Your Domain After Applying the Oracle Service Bus Extension Template

Directory	Files	Description
user_projects\domains\your_domain\	fileRealm.properties	File containing ACLs, users, and groups that can be used for the default security realm when Compatibility security is used.
user_projects\domains\your_domain\	derby.ini	File containing initialization information for an Apache Derby JDBC database.
user_projects\domains\your_domain\	startWebLogic.cmd startWebLogic.sh	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
user_projects\domains\your_domain\	URLs.dat	File containing the URL for the JDBC database.
user_projects\domains\your_domain\	alsbdebug.xml configfwkdebug.xml	Files containing debug parameters for the domain. The default setting for all the parameters is false.
autodeploy\	readme.txt	File providing information about the directory, which initially serves as a placeholder for automatic deployments.
bin\	setDomainEnv.cmd setDomainEnv.sh	Scripts used to set up the development environment on Windows and UNIX systems, respectively.
bin\	startManagedWebLogic.cmd startManagedWebLogic.sh	Scripts used to start a Managed Server on Windows and UNIX systems, respectively.
bin\	startDerbyConsole.cmd startDerbyConsole.sh	Scripts used to start the Apache Derby console on Windows and UNIX systems, respectively.
bin\	startWebLogic.cmd startWebLogic.sh	Scripts used to start the Administration Server on Windows and UNIX systems, respectively.
bin\	stopManagedWebLogic.cmd stopManagedWebLogic.sh	Scripts used to stop a Managed Server on Windows and UNIX systems, respectively.
bin\	stopWebLogic.cmd stopWebLogic.sh	Scripts used to stop the Administration Server on Windows and UNIX systems, respectively.
config\	config.xml	File containing the configuration information used by the Administration Server.
config\deployments\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for staging an application when the application's staging mode is "staged."
config\diagnostics\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing the system modules associated with instrumentation in the WebLogic Diagnostic Framework (WLDF).

Table B–1 (Cont.) Your Domain After Applying the Oracle Service Bus Extension Template

Directory	Files	Description
config\jdbc\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JDBC modules that can be configured directly from JMX (as opposed to JSR-88).
config\jdbc\	wlsbjmsrpDataSource-jdbc.xml	Global non-XA JDBC data source module for the Oracle Service Bus domain.
config\jms\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing global JMS modules that can be configured directly from JMX (as opposed to JSR-88).
config\jms\	wseejmsmodule-jms.xml	Configuration file containing JMS resources for Web Services Reliable Messaging (WS-RM).
config\jms\	xbusResources-jms.xml	Global JMS module for the Oracle Service Bus domain.
config\lib\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing JAR files that are added to the system classpath of the server when the server's Java virtual machine starts.
config\nodemanager\	nm_password.properties	File containing Node Manager password property values.
config\security\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules for the security framework. The directory contains one security provider configuration extension for each type of security provider in the domain's current realm.
config\startup\	readme.txt	File providing information about the directory, which initially serves as a placeholder, and is later used for storing system modules that contain startup plans. Startup plans are used to generate shell scripts that can be used as part of server startup.
console-ext\	readme.txt	File providing information about the directory, which initially serves as a placeholder for custom extensions to the Oracle WebLogic Server Administration Console.
init-info\	domain-info.xml	File used to identify domain creation and extension information. Such information includes the identity of the components in the domain, the location of the JDK and applications directory used by the domain, and the templates used to create and extend the domain.

Table B–1 (Cont.) Your Domain After Applying the Oracle Service Bus Extension Template

Directory	Files	Description
init-info\	security.xml	File used for creating user groups and roles that establish identity and access to domain resources.
init-info\	startscript.xml	File used to create the `*.cmd` and `*.sh` files that are placed into the domain's root and `bin` directories.
init-info\	tokenValue.properties	File that contains the actual values to substitute for the tokens specified in the start scripts.
lib\	readme.txt	File providing information about the directory, which initially serves as a placeholder for the domain's libraries. The JAR files in this directory are added dynamically to the end of the server classpath at server startup.
rmfilestore\	none	Directory serving as a disk-based file store to store persistent messages and durable subscribers.
security\	DefaultAuthenticatorInit.ldift DefaultAuthorizerInit.ldift DefaultRoleMapperInit.ldift XACMLAuthorizerInit.ldift XACMLRoleMapperInit.ldift	Files used for bootstrapping tasks, including authentication (user and group), authorization, and role mapping. These files contain LDAP-specific information. **Note**: WebLogic domains created with this release use the XACML providers by default. These XACML security providers are compatible with policies and roles created using the WebLogic Authorization provider (DefaultAuthorizer) and WebLogic Role Mapping provider (DefaultRoleMapper).
security\	SerializedSystemIni.dat	File containing encrypted security information.
servers\AdminServer\security\	boot.properties	File containing server startup properties, including the user name and password required to boot the server (in encrypted format). It is generated only when you select development startup mode. This file enables you to bypass the prompt for user name and password during a server's startup cycle.