Oracle Insurance Product Definition for Health

Installation Guide

Release 2.13.1.0.0

E39537-01

February 2013

ORACLE

Oracle Insurance Product Definition for Health Installation Guide, Release 2.13.1.0.0

E39537-01

Contents

3 Release Installation

Preface

This document describes the installation information for the Oracle Insurance Product Definition for Health application.

Audience

This document is intended for database managers and system managers and others responsible for the installation of Oracle products.

Documentation Accessibility

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

Access to Oracle Support

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

Conventions

The following text conventions are used in this document:

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

1 Introduction

Disclaimer: This document is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle

1.1 Concepts

1.1.1 Application Modularity

The Oracle Health Insurance suite (OHI) is composed of several applications. In order to ensure full compatibility between these applications, they share a common base.

1.1.2 Database Users and Roles

In order to support the modularity, each application has a separate database schema.

1.1.3 Enabling Replication of Setup Data

During the lifecycle of the application, there is a regular need to transport seed data or setup data from one environment to another using the Configuration Migration tool. This data is identified by an ID and a numeric field. In order to prevent clashes between data created in one environment and existing data in the target environment, it is a prerequisite to ensure that a generated ID is unique across environments. The mechanism used for this purpose is to have the last digit of an ID indicate the source of the row. We recommend the following convention:

Source environments	Description	Discriminator Digit	Examples
OHI Factory	This is the environment at Oracle where the seed data is maintained. The seed data delivered by Oracle always has an ID ending on a zero.	0	17650 17660

Source environments	Description	Discriminator Digit	Examples
Setup	The environment in which you maintain your setup and configuration. This environment should not contain test data.	1	6341 6351
Production	The production environment.	2	3452 3462
Test	The environment in which you test the setup. Setup changes made in the Setup environment are transferred to this environment for testing.	3	165423 165433

Plan the environments and assign unique discriminator values for each environment.

1.2 Release directory structure

1.2.1 Overview of the directory structure

The distribution contains a number of directories that contain all the necessary information and sources to perform the install. The root directory is the directory where you decide to host the released files. It can be any location or name of your choosing and will be referenced throughout this document as <OHI_ROOT>.

For both the Database and the Application server installation, the required installation / configuration files can be made available by copying them to a location on the server or sharing them from their original location.

2

Initial Installation Requirements

This chapter lists specific instructions for installation of the Oracle software components that are required to run the OHI Product Definition application.

See the Certification information that is published on My Oracle Support for specific versions of operating systems and Oracle software that the OHI Product Definition application is certified to work with.

> **Note:** All OHI Product Definition releases contain the *complete* application. The only difference between installing OHI Product Definition for the *first time* and *upgrading* it to a new release, is thus the *pre-installation activities* (which only need to be executed when installing OHI Product Definition for the *first* time)

When the initial installation requirements are met, continue with the chapter *Release Installation.*

2.1 Required and optional software

2.1.1 Required Software

The following table lists the Oracle software that is required in order to execute OHI Product Definition:

Oracle Product	Version
Oracle Database Enterprise Edition	11g Release 2 (11.2.0.3)
Oracle Weblogic Server Enterprise Edition	11g Release 1 (10.3.4)
Oracle TopLink, Application Development Framework and TopLink Grid	11g Release 1 (11.1.1.4.0)
Oracle Coherence Enterprise Edition	3.7.1

> **Note:** For the Oracle Database and Oracle WebLogic server make sure to always apply the latest Critical Patch Updates.

The Oracle Database, Oracle ADF Runtime libraries and Oracle WebLogic server need to be installed. Oracle TopLink, TopLink Grid and Coherence are bundled with each release of OHI Product Definition and do not require a separate installation

2.1.2 Optional Software

This section covers the optional software that can be installed to be used with OHI Product Definition.

2.1.2.1 Oracle Database Options

The following table lists Oracle database options that can optionally be used with OHI Product Definition:

Oracle Product	Option	Version	Motivation
Oracle Database 11gR2 Enterprise Edition	Real Application Clusters	11g Release 2 (11.2.0.3)	Supports the deployment of a single database across a cluster of servers to support a high availability setup.
Oracle Database 11gR2 Enterprise Edition	Total Recall (Flashback Data Archive)	11g Release 2 (11.2.0.3)	For tracking changes to data over time. Although optional, **this is the recommended way to track historical changes to the OHI configuration tables which is not handled in the system in any other way**.

2.1.2.2 Oracle WebLogic Options

The following table lists Oracle WebLogic server additional software that can optionally be used with OHI Product Definition:

Oracle Product	Option	Version	Motivation
Oracle WebLogic Server Enterprise Edition	Oracle Repository Creation Utility (RCU)	11.1.1.4	The RCU is required for the configuration of WS-Security policies for OHI Product Definition web services (whether using Oracle WebLogic WS-Security policies or Oracle Web Services Manager WS-Security policies). If the OHI web services are not secured through WS-Security policies the RCU is not required. See the Security Guide for additional information. The RCU must be downloaded as patch set 11060956 from My Oracle Support.
Oracle WebLogic Server Enterprise Edition	Oracle Web Services Manager	11.1.1.4	Optional, provides more extensive WS-Security support and manageability features.

2.1.2.3 Software for Managing an OHI Product Definition Environment

Oracle recommends to manage and monitor OHI Product Definition environments using Oracle Enterprise Manager (OEM) Cloud Control. The following table lists recommended OEM options

Oracle Product	Option	Version	Motivation
Oracle Enterprise Manager (OEM) Cloud Control	12c Release 1 (12.1.0.1)		
Oracle Enterprise Manager (OEM) Cloud Control	WebLogic Server Management Pack EE	12c Release 1 (12.1.0.1)	To manage multiple WebLogic domains and monitor middleware availability and performance.
Oracle Enterprise Manager (OEM) Cloud Control	Oracle Diagnostic Pack for Database	12c Release 1 (12.1.0.1)	Provides real time and automatic performance diagnostics and simplifies the task of managing large sets of databases.
Oracle Enterprise Manager (OEM) Cloud Control	Oracle Tuning Pack for Database	12c Release 1 (12.1.0.1)	Provides real-time SQL monitoring and tuning advice.

2.2 Install and configure an OHI Database

2.2.1 Install Oracle Database software

First install Oracle Database software required for Oracle Insurance Product Definition for Health (OHI Product Definition); for specific certification details see OHI Product Definition Certification Guide.

2.2.1.1 Setup Real Application Clusters

Setup RAC when required.

2.2.1.2 Create OHI Product Definition database

Now create the OHI Product Definition database. For this activity the following requirements and restrictions apply.

2.2.1.2.1 Character Set

The character set of the database must be AL32UTF8.

2.2.1.2.2 Block Size

For OHI Product Definition, use an 8K block size.

2.2.1.2.3 Tablespaces

There is no specific requirement as to which tablespace exist. During the application installation and/or upgrade we use information specified in ohi_install.cfg. In there you can configure table tablespace (and optionally) an index tablespace.

All tablespaces must be created 11gR2 default style (locally managed, system/uniform managed extent allocation, Automatic Segment Space Management).

A default temporary tablespace TEMP (this name is mandatory) should be created.

Automatic undo must be used.

2.2.1.2.4 Parameters

- NLS_LENGTH_SEMANTICS = CHAR
- STATISTICS_LEVEL=TYPICAL
- _like_with_bind_as_equality=TRUE

Unless specified otherwise, keep all optimizer parameters (gv$sys_optimizer_env) default.

2.2.1.2.5 Required privileges

OHI Product Definition uses queues in the Oracle database. The owner of the queue objects, the Product Definition owner schema, requires execute privileges on the SYS.DBMS_AQIN package.

For installing OHI Product Definition database artifacts the SYSTEM account is used. In this process also database grants are given by the system database user. To be able to do that, SYSTEM user needs GRANT ANY OBJECT PRIVILEGE (without grant option).

2.2.1.2.6 Creating additional schemas in the database

Oracle recommends that the Oracle database instance that is used by OHI Product Definition is used solely for the purpose of running the OHI Product Definition system.

In the case that additional database schemas are created in the Oracle database instance, make sure that these are not prefixed with *OHI.*

2.2.1.2.7 Setup Total Recall (optional)

The Total Recall Option of Oracle Server (also known as Flashback Archiving) is used to log changes to setup tables. Configuring which table to archive and how is considered a responsibility of the database administrator

In order to use Flashback Archiving the following settings need to be made:

- The user that will be used to switch archiving on tables on and off should be granted the "FLASHBACK ARCHIVE ADMINISTER" privilege (grant FLASHBACK ARCHIVE ADMINISTER to <user>).
- This user should be granted "ALTER TABLE" rights on the tables that need to be archived (or stopped being archived).
- Create a separate tablespace for the Flashback Archive. Define this tablespace using "extent management local uniform size 40K". This 40K size (for 11gR2) has proven to be the optimal size for this tablespace.
- Create a Flashback Archive, for example:

```
CREATE FLASHBACK ARCHIVE [DEFAULT] fda1 TABLESPACE tbs1 QUOTA 10G RETENTION 5
YEAR;
```

2.3 Install and configure Oracle Fusion Middleware

OHI Product Definition runs on an Oracle Fusion Middleware Application Server. This may also be referred to as Oracle WebLogic Server. When running on more than one node, the application servers should be configured as a cluster.

This guide assumes experience with setting up Oracle WebLogic Server. For details regarding the installation process please consult the product documentation.

The Certification Guide specifies the required version of the Oracle WebLogic Server software that must be installed. It also describes how the software can be obtained and how the documentation can be accessed.

This chapter outlines the installation of the Oracle WebLogic Server software. Subsequently, the setup of a domain is explained for the following situations:

- A simple, non-clustered environment that is suitable for development and testing purposes. The description of this configuration also demonstrates how Oracle ADF runtime libraries are added to a WebLogic Server domain.
- An advanced, clustered setup that is typically used in production deployments and that is executed on multiple nodes.

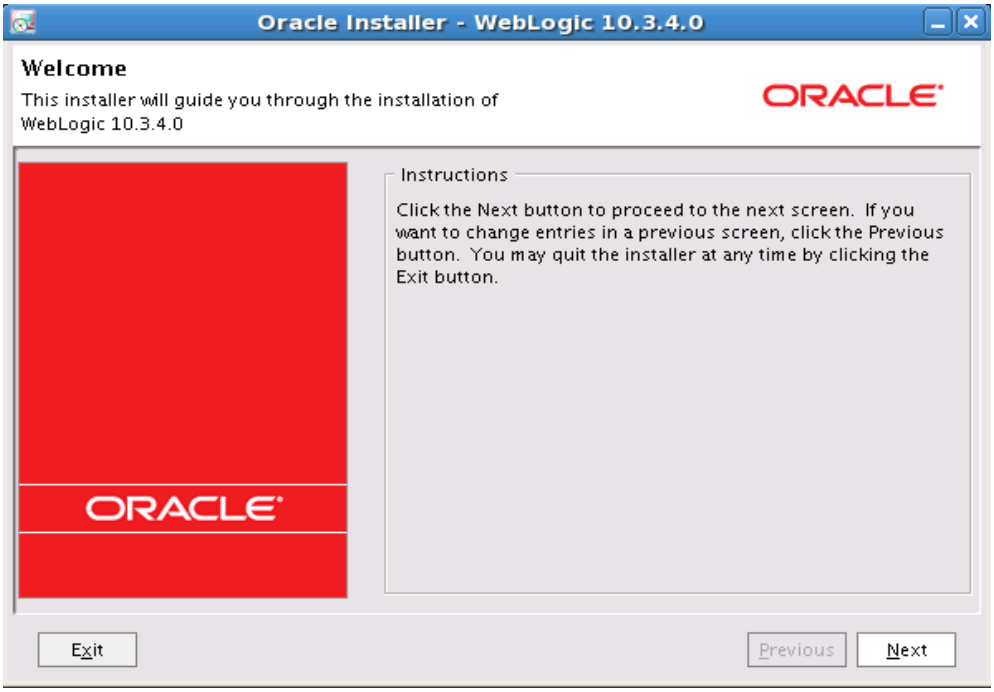
2.3.1 Installing Oracle Weblogic Server

The following steps describe how to install Oracle WebLogic Server. The Certification Guide lists the software that needs to be used as well as the download locations.

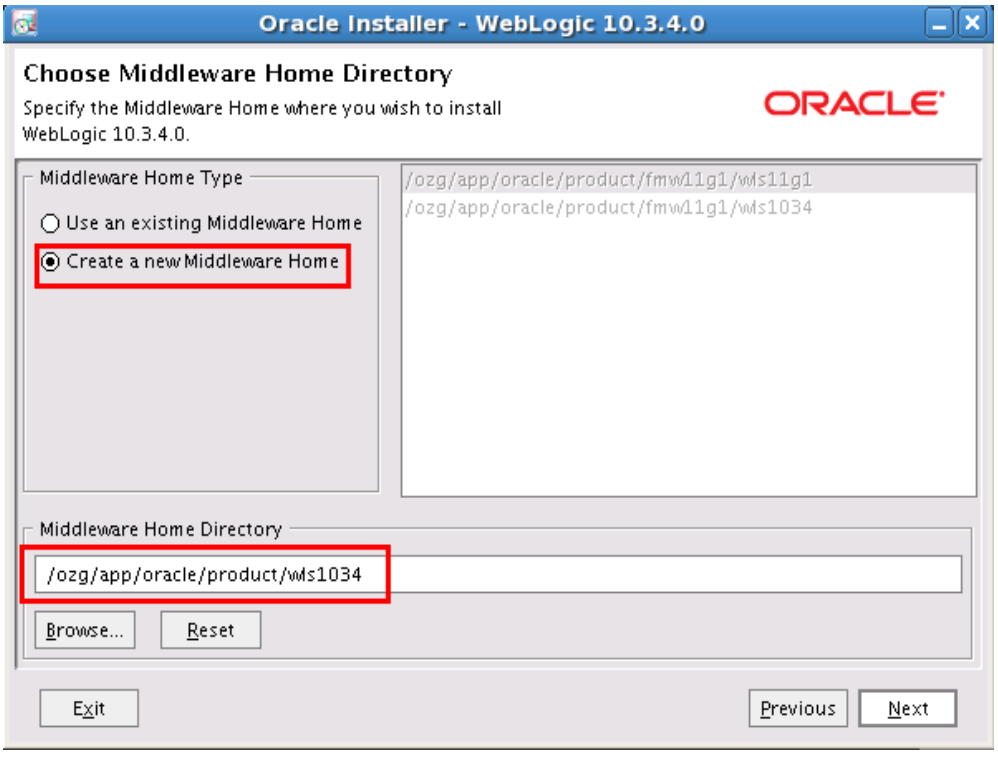
Step 1: Navigate to that folder and run the installer by entering the following command in command line: **java -jar wls1034_generic.jar**

JAVA_HOME should be set before running the installer

Step 2: In the **Welcome** screen click on **Next** button

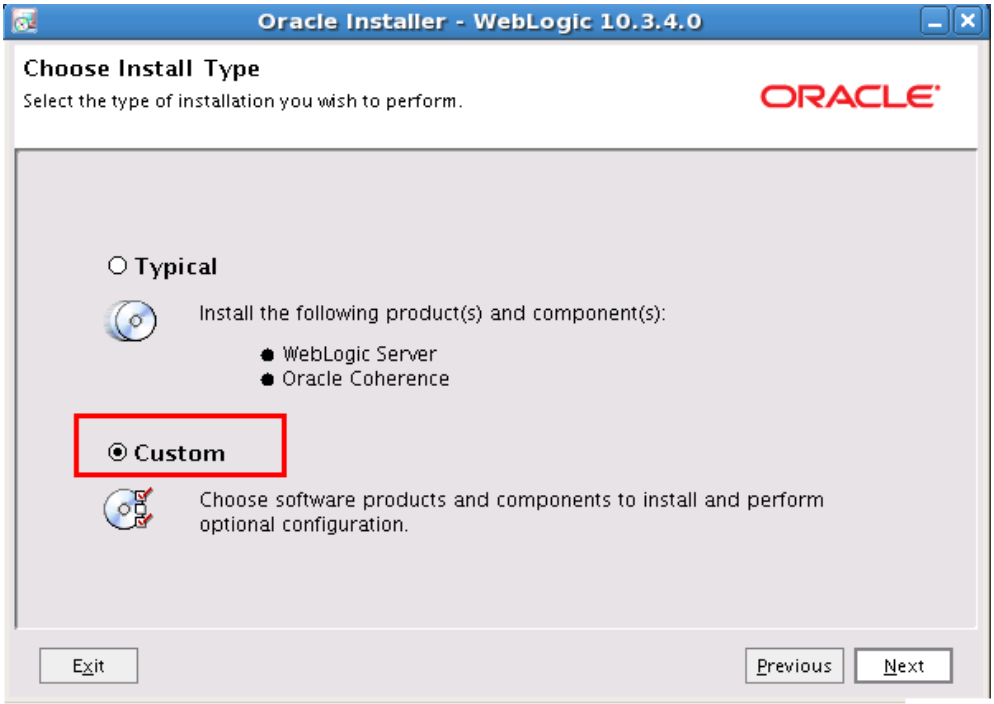


Step 3: In the **Choose Middleware Home Directory** page, select the option **Create a new Middleware Home** and enter the path in **Middleware Home Directory**. Click on **Next** button

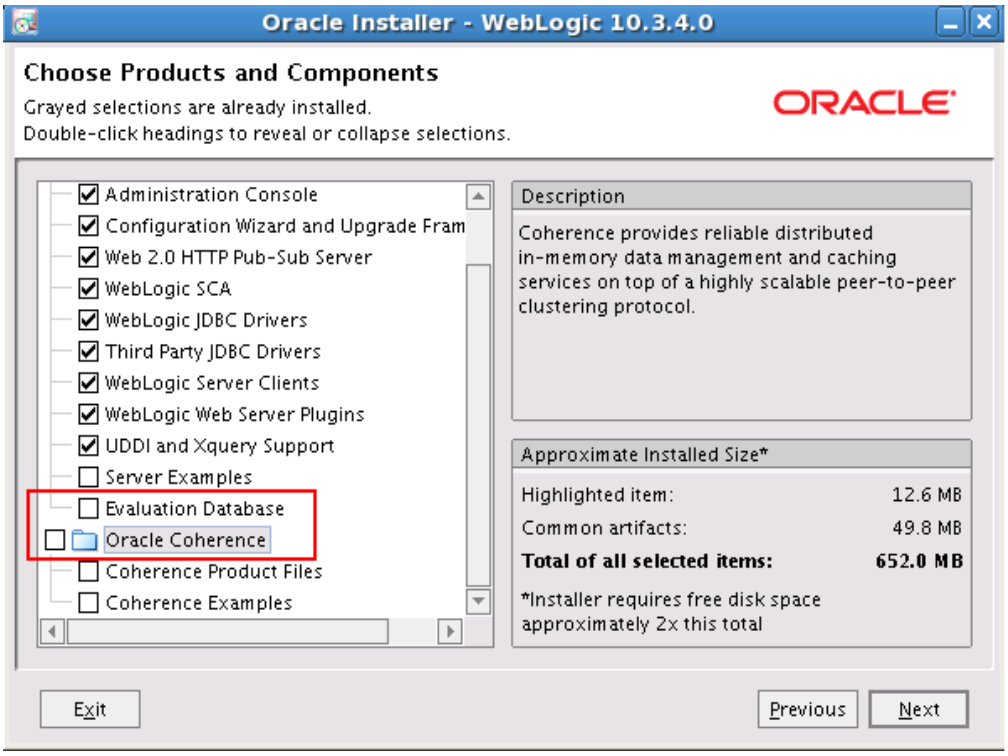


Step 4: In the **Register for Security Updates** page, enter your My Oracle Support Email address and Support Password (optionally, this can be skipped). Click on **Next** button

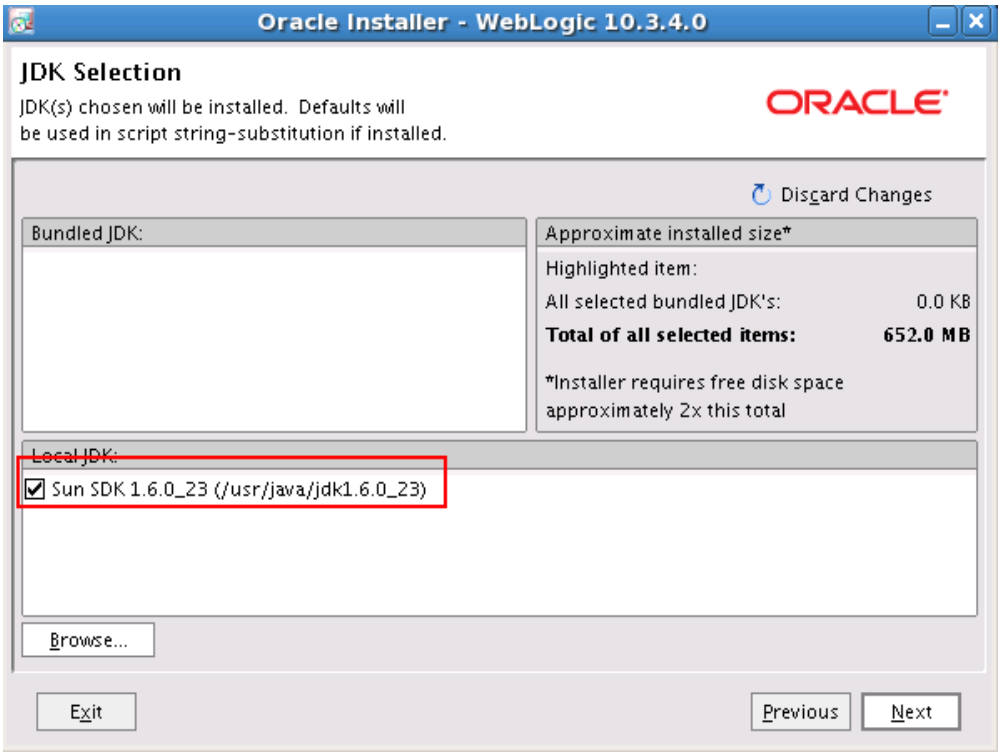
Step 5: In the **Choose Install Type** page, select the option **Custom.** Click on **Next** button



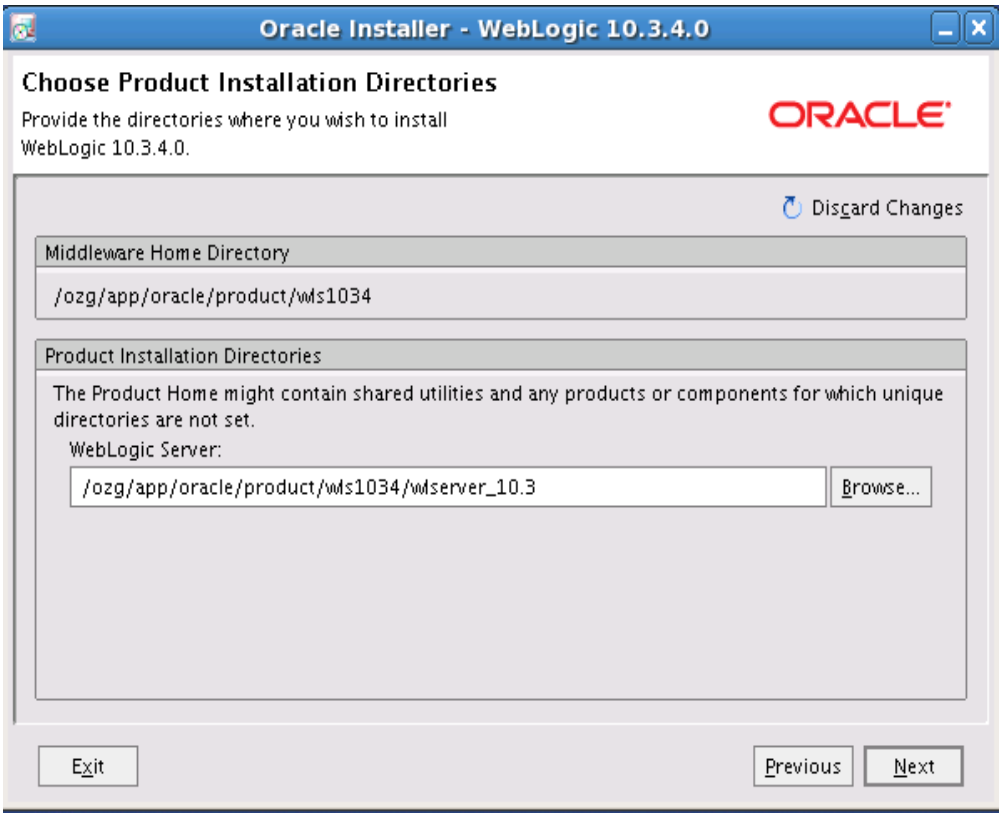
Step 6: In the **Choose Products and Components** page, **deselect** the options **Evaluation Database** and **Oracle Coherence**. Click on **Next** button



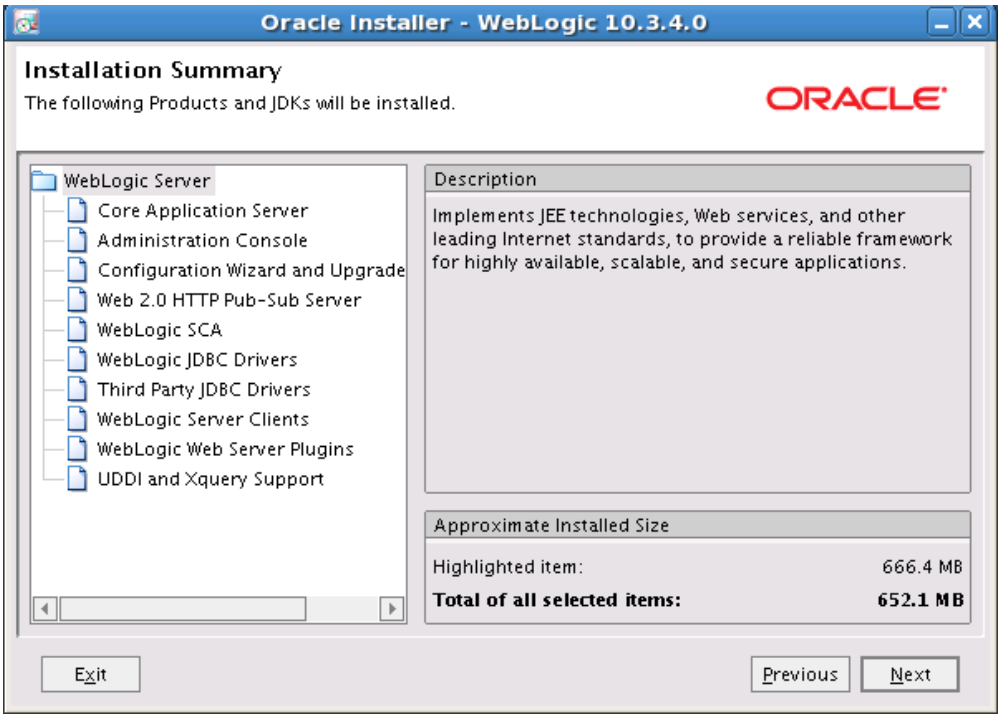
Step 7: The installer for 64-bit machines does not have bundled JDK. So, In **JDK Selection** page, click on **Browse** button to navigate to your JDK installation directory. Click on **Next** button



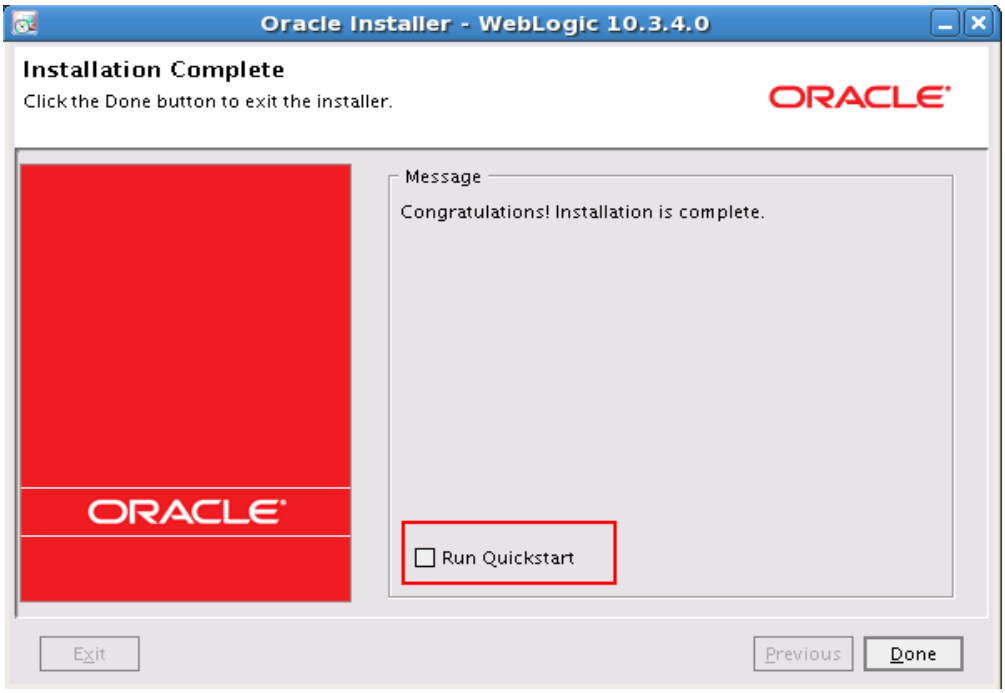
Step 8: In the **Choose Product Installation Directories** page accept the default setting and click on **Next** button



Step 9: In **Installation Summary** page, click on **Next** button



Step 10: When the installation is complete, un-check the check box **Run Quickstart** and click on **Done** button. Oracle WebLogic Server 10.3.4 is now installed.

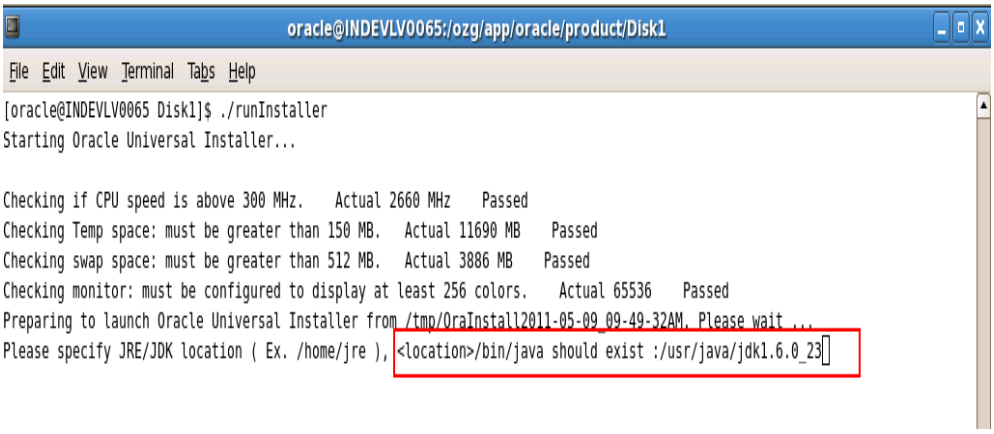


2.3.2 Installing Oracle Applications Development Runtime

The following steps describe how to install Oracle Application Development Runtime. The Certification Guide lists the software that needs to be used as well as the download locations.

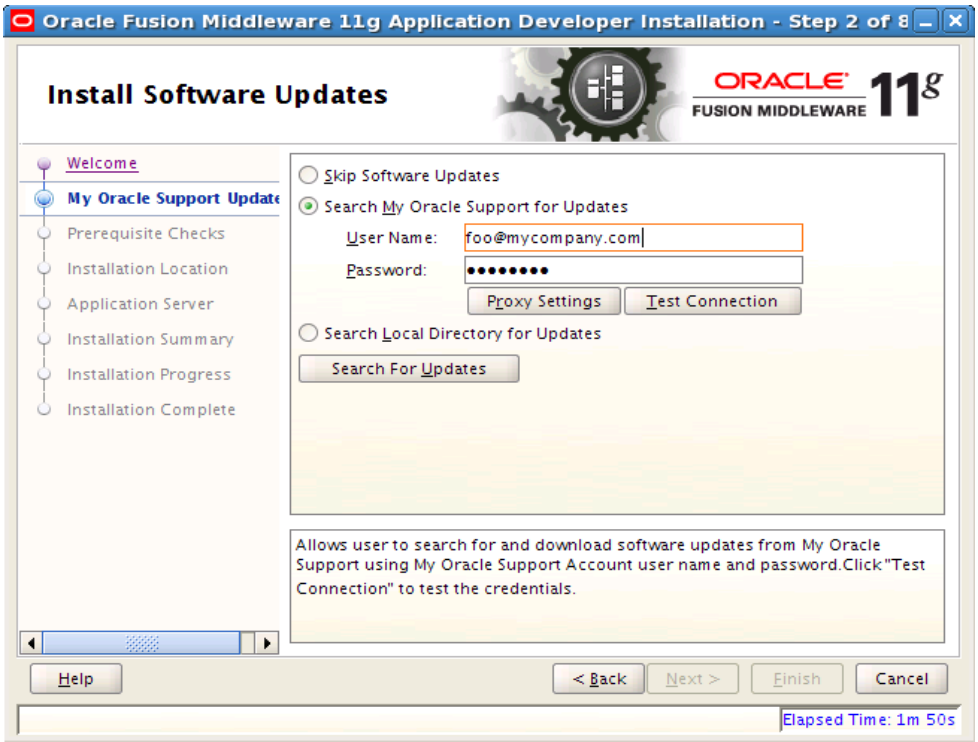
Step 1: Unzip the downloaded archive and navigate to the **Disk1** folder; run the installer by entering the following command in command line: **./runInstaller**

The installer will ask you to enter JDK installation directory

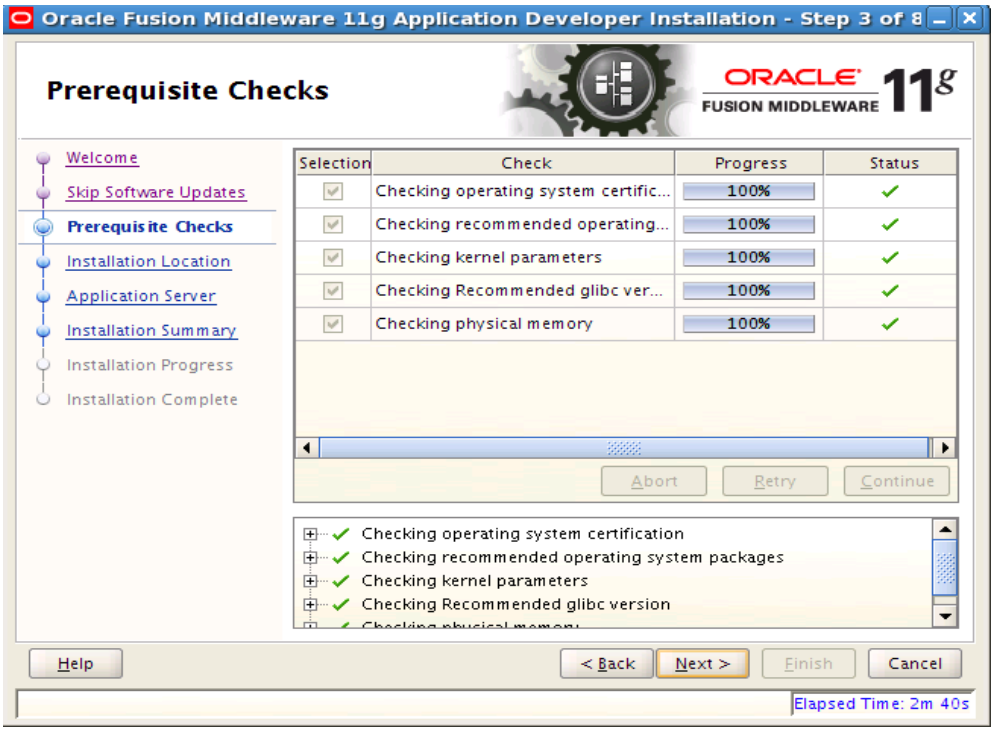


Step 2: In **Welcome** page, click on **Next** button

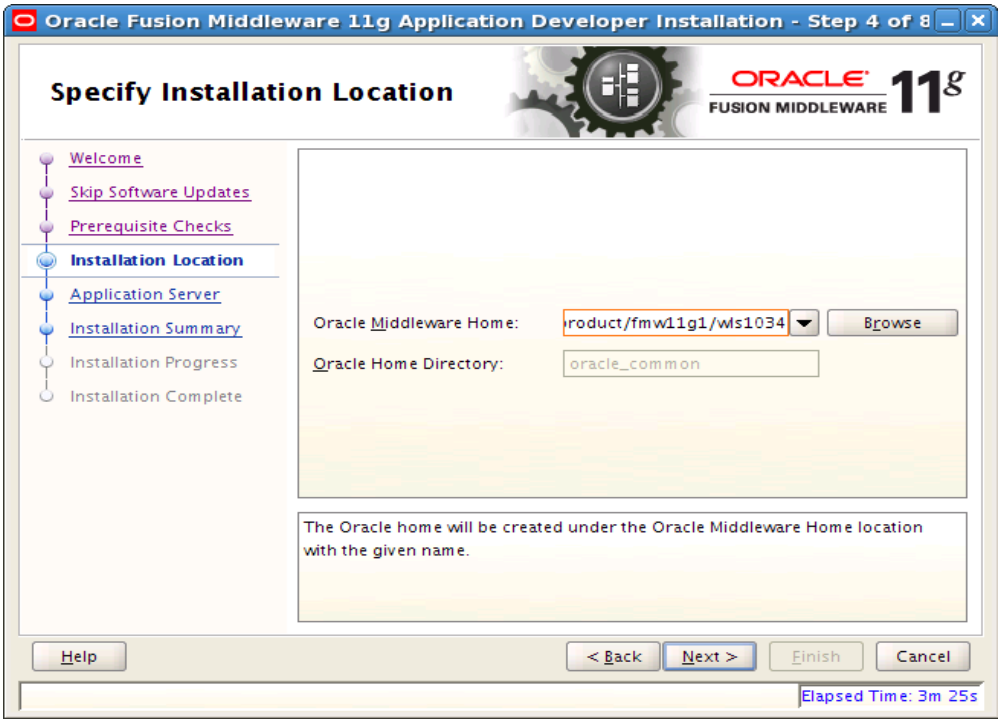
Step 3: In **Install Software Updates** page, enter your My Oracle Support User Name and Password (optionally, this can be skipped). Click on **Next** button



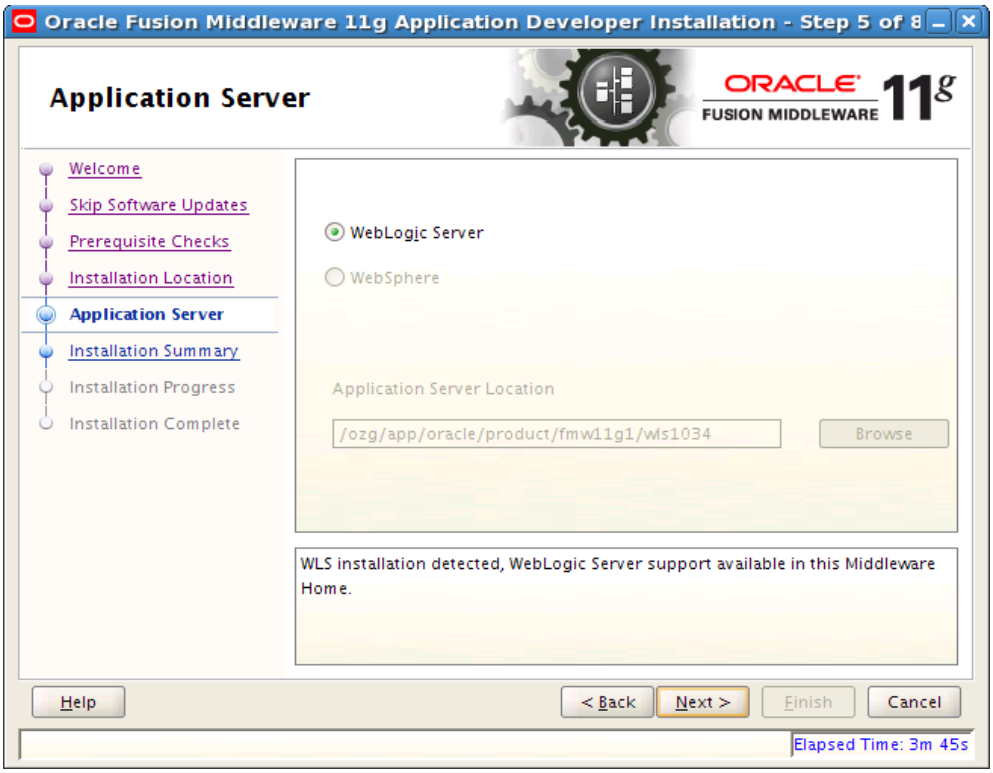
Step 4: Once **Prerequisite Checks** is completed, click on **Next** button



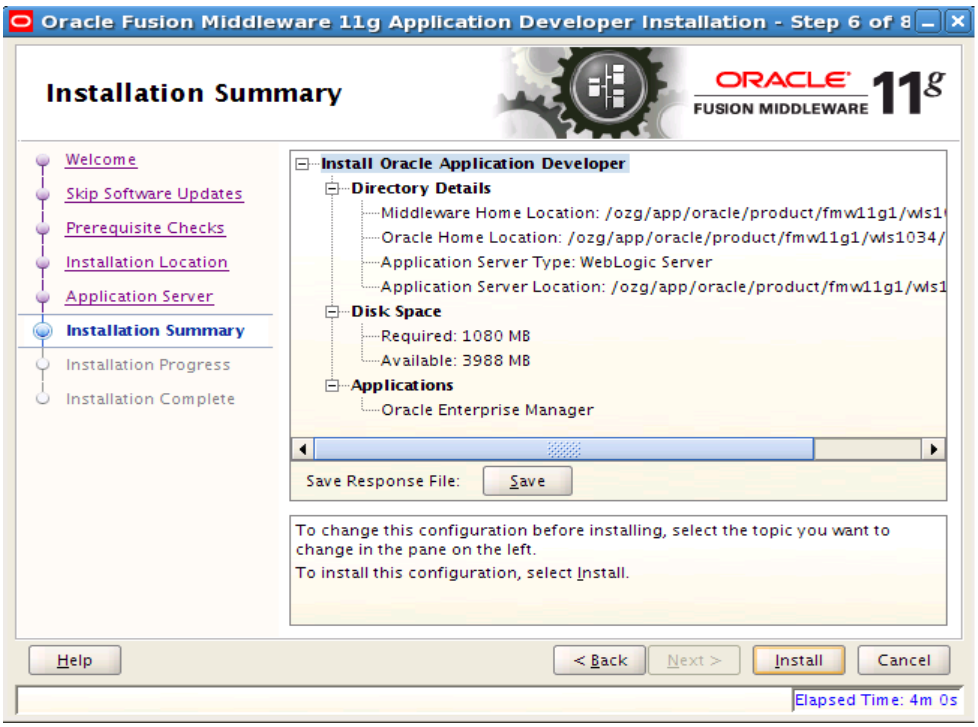
Step 5: In **Specify Installation Location** page, change the value of **Oracle Middleware Home** to suit your WLS installation directory and click on **Next** button



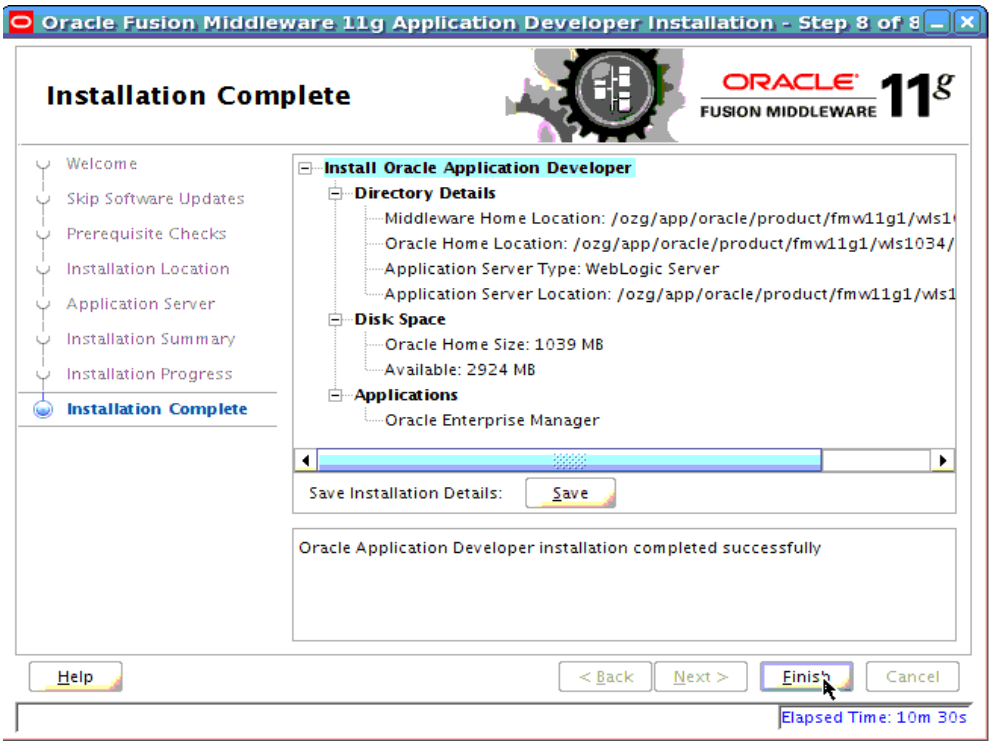
Step 6: In **Application Server** page, accept the default values and click on **Next** button



Step 7: In **Installation Summary** page, click on **Install** button



Step 8: Once the installation is complete, click on **Finish** button in **Installation Complete** page. Oracle Application Development Runtime 11.1.1.4.0 is now installed.



2.3.3 Configuring Oracle Fusion Middleware for running ADF Applications

After installing Oracle WebLogic Server and Oracle Application Development Runtime, perform the following steps:

- Create a domain in which the OHI Product Definition application will be configured and installed.
- Detailed instructions are available in the documentation library (http://download.oracle.com/docs/cd/E12839_01/web.1111/b31974/deployment_topics.htm#ADFFD1831).

Alternatively, follow these steps to create a domain.

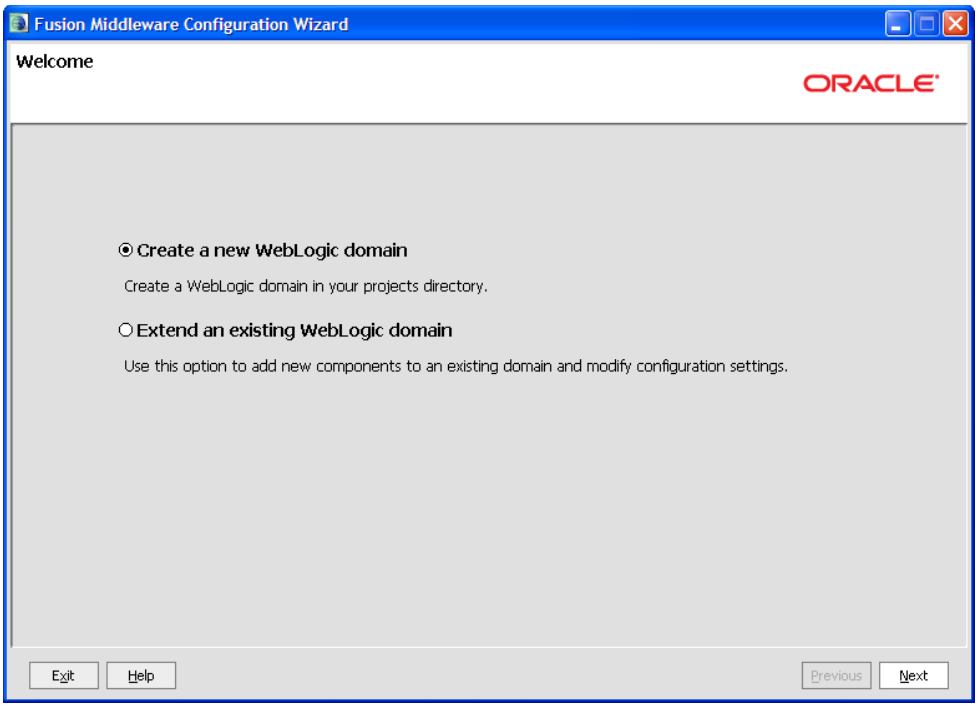
The following domain setup is suitable for development and testing purposes but should not be used in production situations.

Step 1: Go to **<MIDDLEWARE_HOME_DIRECTORY>/wlserver_10.3/common/bin** in command prompt . Here MIDDLEWARE_HOME_DIRECTORY is the path where you instaled WLS 10.3.4.

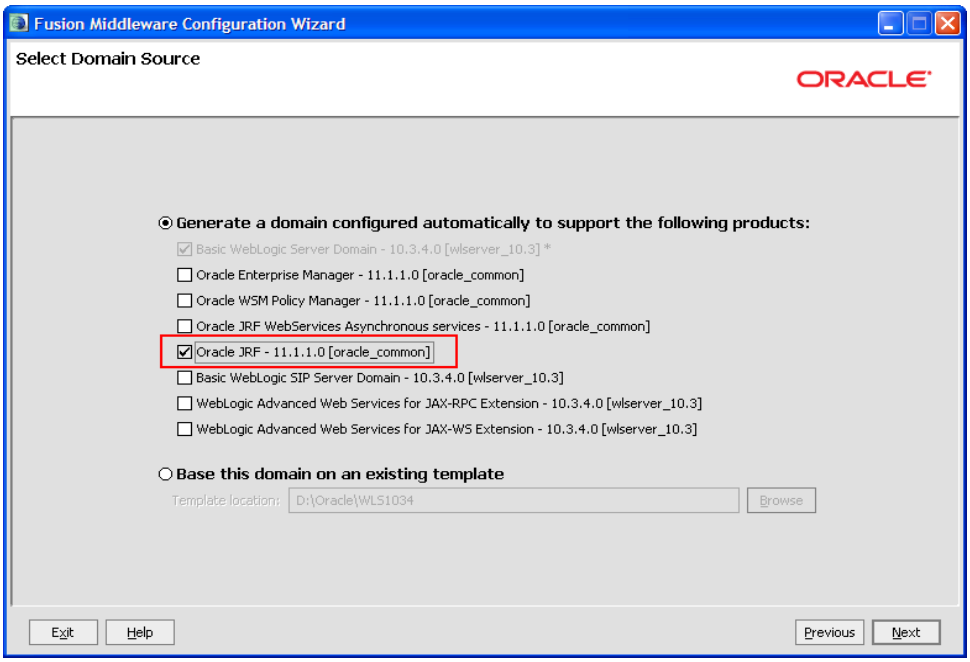
Step 2: Issue the following command: **./config.sh**

Step 3: **Fusion Middleware Configuration Wizard - Welcome** screen appears.

Step 4: In **Welcome** page, leave the default selection **Create a new WebLogic domain** and click on **Next** button

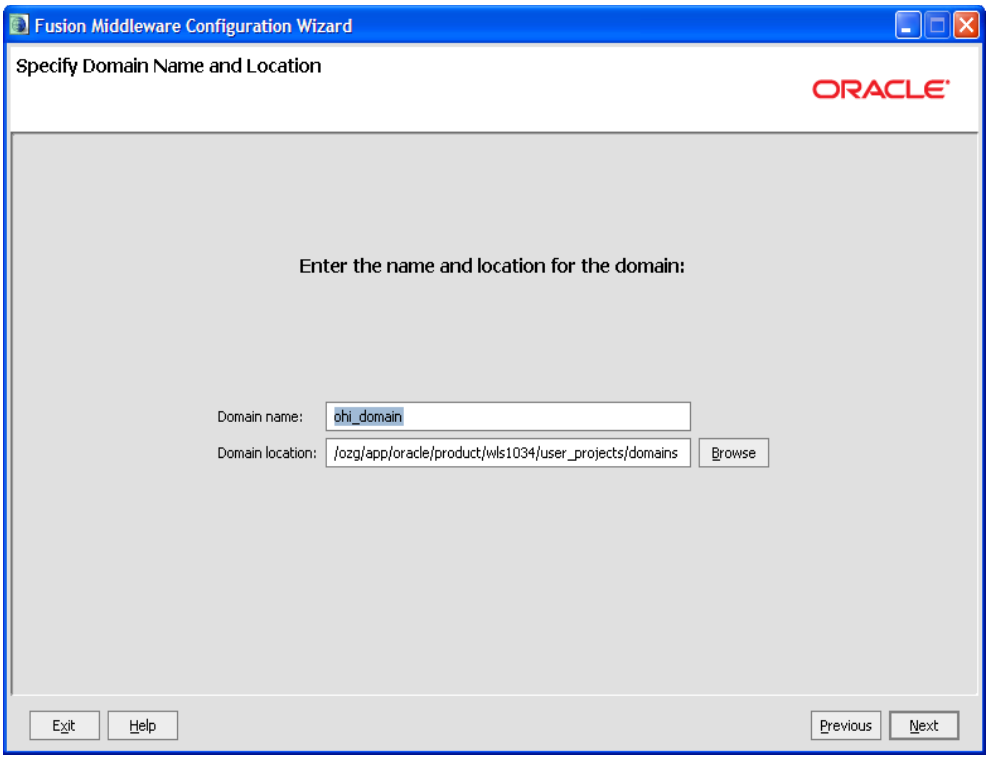


Step 5: In the **Select Domain Source** page, select the check box **Oracle JRF - 11.1.1.0 [oracle_common]** and click on **Next** button

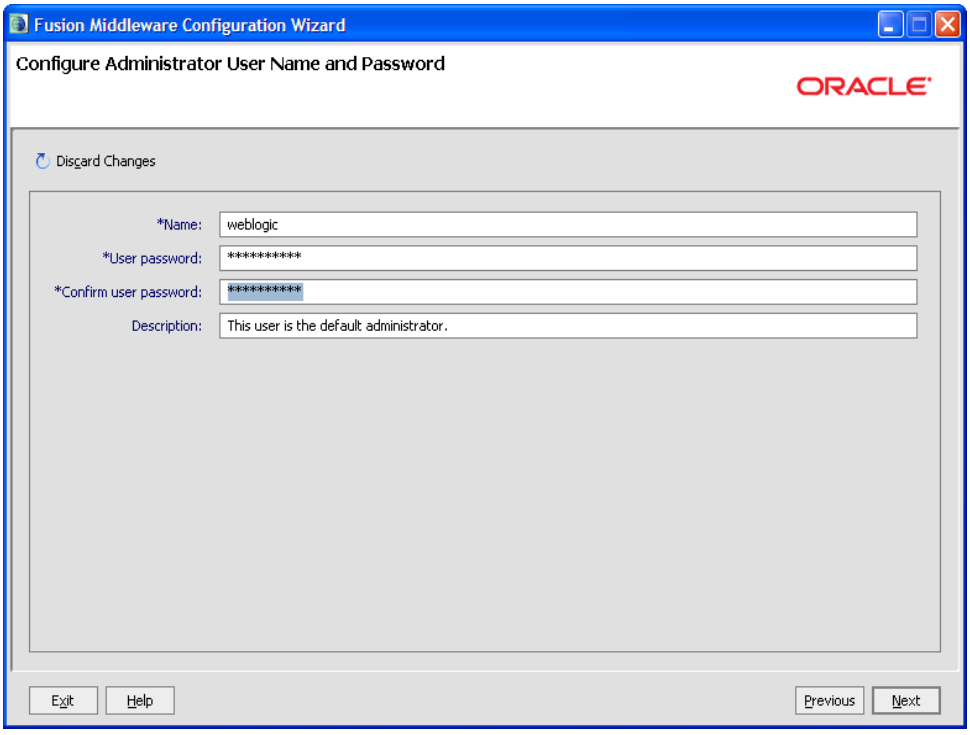


Step 6: In the **Specify Domain Name and Location** page, edit the values for **Domain name** and **Domain location** to suit your requirements or leave the default values and click on **Next** button.

For consistency, Oracle recommends the value "ohi_domain" as domain name.

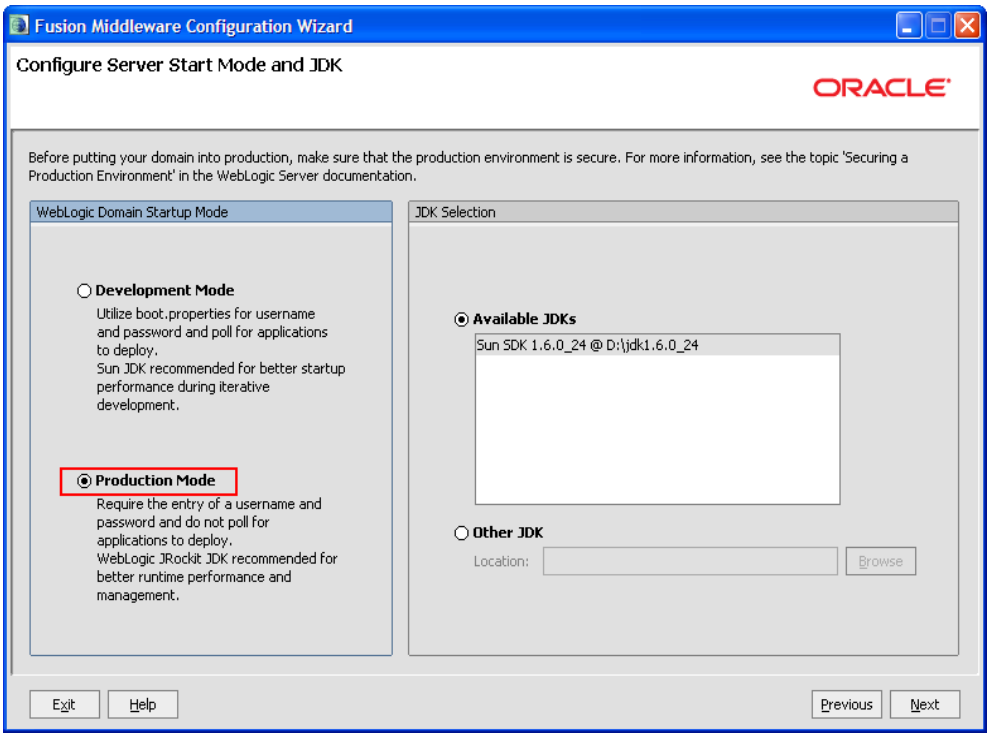


Step 7: In the **Configure Administrator User Name and Password** page, enter the values for **User password** and **Confirm user password** and click on **Next** button.

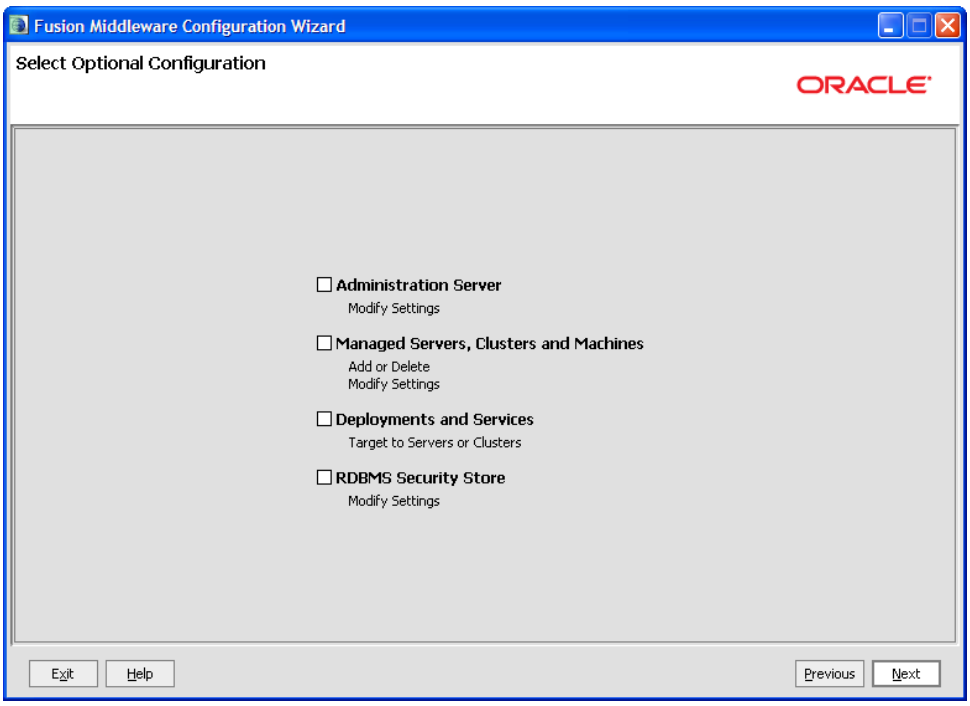


The password must be at least 8 alphanumeric characters with at least one number or special character.

Step 8: In the **Configure Server Start Mode and JDK** page, change the value for **WebLogic Domain Startup Mode** to **Production Mode** and click on **Next** button.

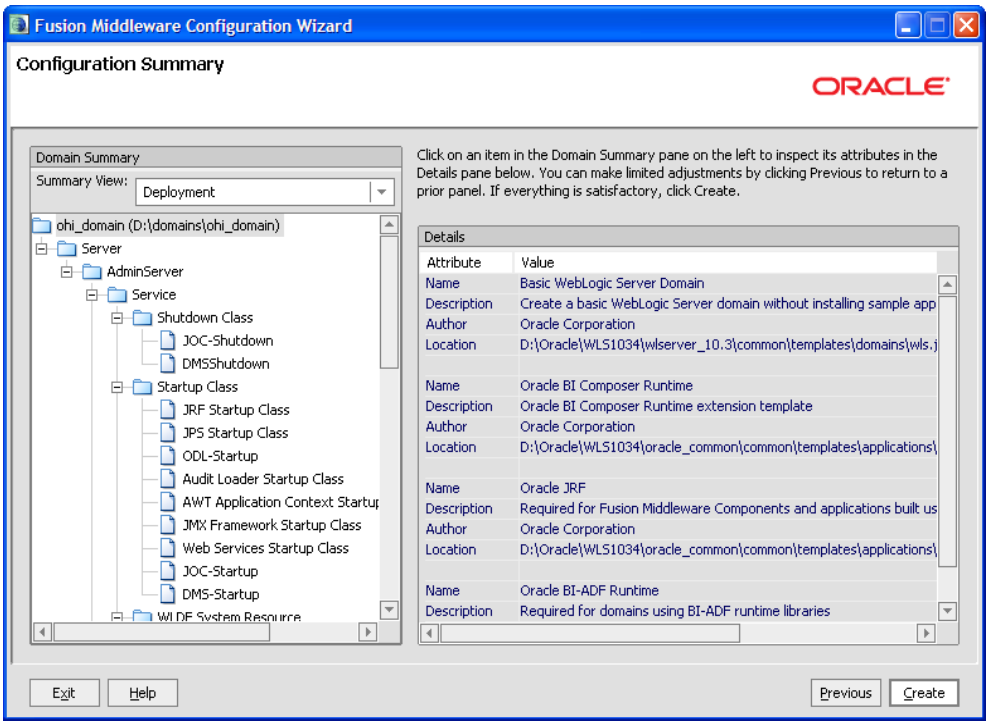


Step 9: In the **Select Optional Configuration** page, select the options that you want to configure. Else, leave with the default settings and click on **Next** button.

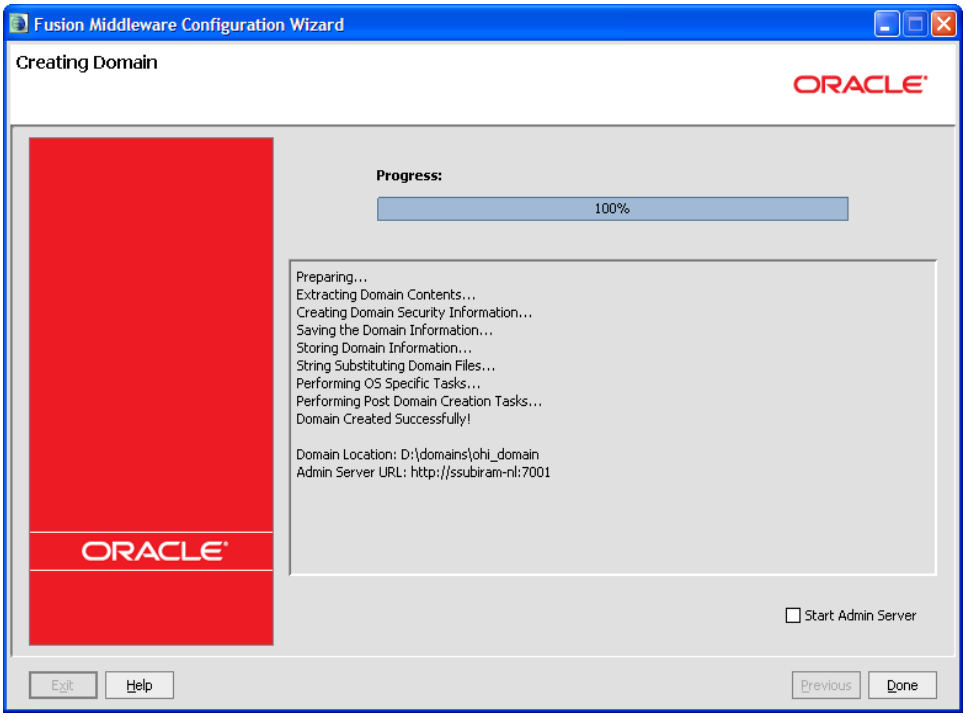


The default AdminServer listening port is 7001.

Step 10: In the **Configuration Summary** page, click on **Create** button.



Step 11: In the **Creating Domain** page, click on **Done** button once the domain is created.



2.3.4 Domain configuration for OHI Product Definition

This chapter contains directions for the following topics:

- Redirecting console log output
- Setting up OHI Product Definition properties files

- Coherence settings
- Setting OHI Product Definition Domain environment variables

2.3.4.1 Redirect JVM Output to a Log File

By default, the JVM output for a WebLogic server is written to the console. It is recommended to redirect the console output to file. Note that in development mode, the default size of a logfile before it is rotated is only 500Kb. Hence, it is also recommended to change the size of the log files before rollover to 10240 Kb and to specify the number of log files that will be retained. These configuration settings can be changed through the WebLogic Server Console.

2.3.4.2 Setting up OHI Product Definition Properties File

Create a directory that will hold OHI Product Definition properties and configuration files. This directory will be referenced as **<PROPERTIES_ROOT>** throughout this document.

Copy the following files that were delivered as part of the specific release from the **<OHI_ROOT>/properties** directory to the **<PROPERTIES_ROOT>**:

- logback.xml
- ohi-proddef.properties.template

Rename the copied ohi-proddef.properties.template to ohi-proddef.properties. A description of the properties files is available section 'Configure OHI Product Definition properties file' of this document.

Also copy file **<OHI_ROOT>**/util/security/ohi-security.config to the **<PROPERTIES_ROOT>**.

2.3.4.3 Coherence settings

OHI Product Definition uses Oracle Coherence. The IT infrastructure on which the system is installed determines the configuration for Oracle Coherence. This paragraph describes the following configuration options:

- Restrict a Coherence cluster to one machine
- Control multiple Coherence clusters that are spread across multiple machines
- Control multiple Coherence clusters that are executed on one machine
- Specific settings for running Coherence in a Production environment

Restrict a Coherence cluster to one machine

The **<PROPERTIES_ROOT>** directory contains a Coherence configuration file (single-server-tangosol-coherence-override.xml) that ensures that a Coherence cluster is restricted to a single machine.

These settings will constrain Coherence to run on a single machine. It will not prevent Coherence from clustering with other JVMs on the same machine that also run Coherence. Therefore, it is not suitable for setting up multiple Coherence clusters on a single machine.

Copy the following properties file that was delivered as part of the specific release from the **<OHI_ROOT>**\properties directory to the **<PROPERTIES_ROOT>**:

- single-server-tangosol-coherence-override.xml

In order to control which JVMs can join in a particular Coherence cluster, the Coherence Well Known Addresses (WKA) feature may be used. This can be used to:

- Control multiple Coherence clusters that are spread across multiple machines
- Control multiple Coherence clusters that are executed on one machine

A preconfigured tangosol-coherence-override.xml file for these situations cannot be provided as required host names or IP addresses must be used. The following sample files show the basic structure.

Schema validation is used to ensure that configuration files adhere to their respective schema definition. Configuration files that include a schema reference are automatically validated against the schema when the configuration file is loaded. A validation error causes an immediate failure and an error message is emitted that indicates which element caused the error. **Schema validation should always be used as a best practice.** To enable schema validation, make sure to always start a coherence-override file with the following root element definition:

```
<coherence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://xmlns.oracle.com/coherence/coherence-operational-config"

xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-operational-config
coherence-operational-config.xsd">

 ...

</coherence>

<coherence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://xmlns.oracle.com/coherence/coherence-operational-config"

xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-operational-config
coherence-operational-config.xsd">

 <configurable-cache-factory-config>

<class-name>com.oracle.healthinsurance.support.cache.factory.OhiDefaultConfigurabl
eCacheFactory</class-name>
 <init-params>
 <init-param>
 <param-type>java.lang.String</param-type>
 <param-value
system-property="tangosol.coherence.cacheconfig">/META-INF/coherence-cache-config.
xml</param-value>
 </init-param>
 </init-params>
 </configurable-cache-factory-config>

 ...

</coherence>
```

The use of the configurable-cache-factory-config element is mandatory for any Coherence configuration (override file) used with OHI Product Definition.

The following sample override file controls a Coherence cluster that runs on JVMs on several machines (host1, host2, ..., hostN):

```
<coherence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://xmlns.oracle.com/coherence/coherence-operational-config"

xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-operational-config
coherence-operational-config.xsd">

 <configurable-cache-factory-config>

<class-name>com.oracle.healthinsurance.support.cache.factory.OhiDefaultConfigurabl
eCacheFactory</class-name>
 <init-params>
 <init-param>
 <param-type>java.lang.String</param-type>
 <param-value
system-property="tangosol.coherence.cacheconfig">/META-INF/coherence-cache-config.
xml</param-value>
 </init-param>
 </init-params>
 </configurable-cache-factory-config>
 <cluster-config>
 <unicast-listener>
 <well-known-addresses>
 <socket-address id="1">
 <address system-property="tangosol.coherence.wka1">host1</address>
 <port system-property="tangosol.coherence.wka1.port">8088</port>
 </socket-address>
 <socket-address id="2">
 <address system-property="tangosol.coherence.wka2">host2</address>
 <port system-property="tangosol.coherence.wka2.port">8088</port>
 </socket-address>
 ...
 <socket-address id="N">
 <address system-property="tangosol.coherence.wkaN">hostN</address>
 <port system-property="tangosol.coherence.wkaN.port">8088</port>
 </socket-address>
 </well-known-addresses>
 </unicast-listener>
 </cluster-config>
</coherence>
```

Start the JVM on host1 with the following command-line parameters:

These options should be specified on one line, it was formatted differently in this guide for readability.

```
-Dtangosol.coherence.wka1=host1
-Dtangosol.coherence.wka1.port=8088
-Dtangosol.coherence.localport=8088
-Dtangosol.coherence.override=tangosol-coherence-override.xml
```

Start the JVM on host2 with the following command-line parameters:

```
-Dtangosol.coherence.wka2=host2
-Dtangosol.coherence.wka2.port=8088
-Dtangosol.coherence.localport=8088
-Dtangosol.coherence.override=tangosol-coherence-override.xml
```

The following sample override file controls a Coherence cluster that runs on multiple JVMs on the same machine (host1):

```
<coherence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://xmlns.oracle.com/coherence/coherence-operational-config"
```

```
xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-operational-config
coherence-operational-config.xsd">

 <configurable-cache-factory-config>

<class-name>com.oracle.healthinsurance.support.cache.factory.OhiDefaultConfigurabl
eCacheFactory</class-name>
 <init-params>
 <init-param>
 <param-type>java.lang.String</param-type>
 <param-value
system-property="tangosol.coherence.cacheconfig">/META-INF/coherence-cache-config.
xml</param-value>
 </init-param>
 </init-params>
 </configurable-cache-factory-config>
 <cluster-config>
 <unicast-listener>
 <well-known-addresses>
 <socket-address id="1">
 <address system-property="tangosol.coherence.wka1">host1</address>
 <port system-property="tangosol.coherence.wka1.port">8088</port>
 </socket-address>
 <socket-address id="2">
 <address system-property="tangosol.coherence.wka2">host1</address>
 <port system-property="tangosol.coherence.wka2.port">8089</port>
 </socket-address>
 ...
 <socket-address id="N">
 <address system-property="tangosol.coherence.wkaN">host1</address>
 <port system-property="tangosol.coherence.wkaN.port">8090</port>
 </socket-address>
 </well-known-addresses>
 </unicast-listener>
 </cluster-config>
</coherence>
```

Start the first JVM on host1 with the following command-line parameters:

These options should be specified on one line, it was formatted differently in this guide for readability.

```
-Dtangosol.coherence.wka1=host1
-Dtangosol.coherence.wka1.port=8088
-Dtangosol.coherence.localport=8088
-Dtangosol.coherence.override=tangosol-coherence-override.xml
```

Start the second JVM on host1 with the following command-line parameters:

```
-Dtangosol.coherence.wka2=host1
-Dtangosol.coherence.wka2.port=8089
-Dtangosol.coherence.localport=8089
-Dtangosol.coherence.override=tangosol-coherence-override.xml
```

For more information please check the Coherence documentation on Well Known Addresses.

Specific settings for running Coherence in a Production environment

By default, Oracle Coherence runs in Development mode. The production checklist in the Coherence documentation states that *it is recommended to use the development mode*

for all pre-production activities, such as development and testing. This is an important safety feature, because Coherence automatically prevents these nodes from joining a production cluster. The production mode must be explicitly specified when using Coherence in a production environment.

In the Production environment (and only in the Production environment), the system property *tangosol.coherence.mode* should be set to value *prod* in the script that is used to start Coherence nodes.

Next to that, Oracle advises to use system property *tangosol.coherence.cluster* to name the cluster. To join the cluster, all members must specify the same cluster name. Suggested naming convention:*OHI-<systemproperty.ohi.environment.identifier>*.

```
-Dtangosol.coherence.mode=prod
-Dtangosol.coherence.cluster=<cluster_name>
```

2.3.5 Set Environment Variables for OHI Product Definition

Environment variables for OHI Product Definition can be set in the startManagedWebLogic.sh script or the setDomainEnv.sh script. An alternative approach (offered as a best practice) is to create a separate shell script named **setOhiEnv.sh** in a directory (referred to hereafter as <SET_ENV_VAR_DIR>). Rationale:

- The setDomainEnv.sh file is generated by WLS and large (clutters the view).
- The setDomainEnv.sh file can be changed by WebLogic if the cluster configuration changes. A separate setOhiEnv.sh file shields from these changes.
- The settings in setDomainEnv.sh are also applied when stopWebLogic.sh is used. In that case, the memory settings for Managed Servers that execute the OHI Product Definition application can cause errors.

This approach does not work in conjunction with a Node Manager. For the Node Manager, the settings have to be in setDomainEnv.sh or set in the Server Start up properties (in Admin console).

Make sure that <SET_ENV_VAR_DIR> is a shared directory for all the managed servers in the cluster. The following is a sample setOhiEnv.sh script:

```
# Memory Args
USER_MEM_ARGS="-Xms4096m -Xmx4096m -XX:PermSize=768m -XX:MaxPermSize=768m
-XX:NewSize=1280m -XX:MaxNewSize=1280m"
USER_MEM_ARGS="$USER_MEM_ARGS -XX:ReservedCodeCacheSize=128m"
USER_MEM_ARGS="$USER_MEM_ARGS -XX:+UseConcMarkSweepGC"
USER_MEM_ARGS="$USER_MEM_ARGS -XX:+UseParNewGC"
USER_MEM_ARGS="$USER_MEM_ARGS -XX:+ExplicitGCInvokesConcurrent"
USER_MEM_ARGS="$USER_MEM_ARGS -XX:+CMSClassUnloadingEnabled"
USER_MEM_ARGS="$USER_MEM_ARGS -XX:+UseCMSCompactAtFullCollection"
export USER_MEM_ARGS

# Java Options
JAVA_OPTIONS="-Dohi.mds.country=US"
JAVA_OPTIONS="${JAVA_OPTIONS}
-Dohi.properties.file=/properties/ohi-proddef.properties"
JAVA_OPTIONS="${JAVA_OPTIONS}
-Dtangosol.coherence.override=/properties/tangosol-coherence-override.xml"
JAVA_OPTIONS="${JAVA_OPTIONS}
-Djava.security.auth.login.config=/properties/ohi-security.config"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.cluster=<cluster_name>"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.mode=prod"
```

```
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.localport=<coherence_port>"
JAVA_OPTIONS="${JAVA_OPTIONS}
-Dcom.sun.org.apache.xml.internal.dtm.DTMManager=com.sun.org.apache.xml.internal.d
tm.ref.DTMManagerDefault"
JAVA_OPTIONS="${JAVA_OPTIONS}
-Djavax.xml.datatype.DatatypeFactory=com.sun.org.apache.xerces.internal.jaxp.datat
ype.DatatypeFactoryImpl"
export JAVA_OPTIONS

# Optional settings for JMX management
JAVA_OPTIONS="${JAVA_OPTIONS} -Dcom.sun.management.jmxremote.authenticate=false"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dcom.sun.management.jmxremote.ssl=false"
JAVA_OPTIONS="${JAVA_OPTIONS}
-Djavax.management.builder.initial=weblogic.management.jmx.mbeanserver.WLSMBeanSer
verBuilder"
export JAVA_OPTIONS

# Optional settings to enable monitoring Coherence through JMX
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.management=all"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.management.remote=true"
export JAVA_OPTIONS

# Settings specific to Managed Servers -- repeated for each Managed Server 1 to N
if [ "$1" = "managed_server1" ] ; then
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.member=<coherence_member_name>"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.wka1=<hostname or IP address of
managed server 1>"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.wka1.port=<coherence_port>"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dlogback.configurationFile=<properties
directory>/logback_node_1.xml"
# Optional JMX Setting
JAVA_OPTIONS="${JAVA_OPTIONS} -Dcom.sun.management.jmxremote.port=<unique_port>"
export JAVA_OPTIONS
fi

if [ "$1" = "managed_server2" ] ; then
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.member=<coherence_member_name>"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.wka2=<hostname or IP address of
managed server 2>"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.wka2.port=<coherence_port>"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dlogback.configurationFile=<properties
directory>/logback_node_2.xml"
# Optional JMX Setting
JAVA_OPTIONS="${JAVA_OPTIONS} -Dcom.sun.management.jmxremote.port=<unique_port>"
export JAVA_OPTIONS
fi

if [ "$1" = "managed_serverN" ] ; then
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.member=<coherence_member_name>"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.wkaN=<hostname or IP address of
managed server N>"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dtangosol.coherence.wkaN.port=<coherence_port>"
JAVA_OPTIONS="${JAVA_OPTIONS} -Dlogback.configurationFile=<properties
directory>/logback_node_N.xml"
# Optional JMX Setting
JAVA_OPTIONS="${JAVA_OPTIONS} -Dcom.sun.management.jmxremote.port=<unique_port>"
export JAVA_OPTIONS
fi
```

Go to **<MIDDLEWARE_HOME_DIRECTORY>/user_projects/domains/<DOMAIN_NAME>/bin** where **<DOMAIN_NAME>** is the name of the domain that was given in Step 6 of the previous section. Edit the file startManagedWebLogic.sh in that directory and add the following line (highlighted below) at the beginning as shown in this sample:

```
#!/bin/sh

# WARNING: This file is created by the Configuration Wizard.
# Any changes to this script may be lost when adding extensions to this
configuration.

# --- Start Functions ---

usage()
{
 echo "Need to set SERVER_NAME and ADMIN_URL environment variables or specify"
 echo "them in command line:"
 echo "Usage: $1 SERVER_NAME {ADMIN_URL}"
 echo "for example:"
 echo "$1 managedserver1 http://bal03028:7005"
}

# --- End Functions ---

# ******************************************************************************
# This script is used to start a managed WebLogic Server for the domain in
# the current working directory. This script can either read in the SERVER_NAME
and
# ADMIN_URL as positional parameters or will read them from environment variables
that are
# set before calling this script. If SERVER_NAME is not sent as a parameter or
exists with a value
# as an environment variable the script will EXIT. If the ADMIN_URL value cannot
be determined
# by reading a parameter or from the environment a default value will be used.
#
# For additional information, refer to "Managing Server Startup and Shutdown for
Oracle WebLogic Server"
# (http://download.oracle.com/docs/cd/E17904_01/web.1111/e13708/overview.htm)
# ******************************************************************************
. <SET_ENV_VAR_DIR>/setOhiEnv.sh

DOMAIN_NAME="<domain_name>"

ADMIN_URL="http://<machine>:<port>"
```

USER_MEM_ARGS Explanation:

- -Xms2048m -Xmx2048m -- this represents the heap size allocated for the JVM. Xms and Xmx should always be the same number.
-
  - Determining what these sizes should be in production environments requires a full JVM sizing exercise. More on JVM sizing for production is available at 'OHI-Product Definition JVM Sizing'.
- -XX:PermSize=512m -XX:MaxPermSize=512m -- this sets the size for the permanent generation of the JVM's heap.

- -XX:+UseConcMarkSweepGC -XX:+UseParNewGC -XX:+ExplicitGCInvokesConcurrent -XX:+CMSClassUnloadingEnabled -XX:+UseCMSCompactAtFullCollection -- these are the garbage collector settings recommended for use with the OHI-Product Definition application. More information on JVM options and garbage collector settings at Java HotSpot VM Options.

JAVA_OPTIONS Explanation:

- tangosol.coherence.mode: use this property for production environments only.
- tangosol.coherence.cluster: the same name needs to be specified by all members in order to join a specific cluster.
- tangosol.coherence.member: the member-name element contains the name of the member itself. This name makes it possible to easily differentiate among members, such as when multiple members run on the same machine. If a name is not specified, the node will fail to start (IllegalArgumentException). Suggested naming convention: OHI-<systemproperty.ohi.environment.identifier>-<machinename_or_ip-address>-<unique-identifier>. Note that the value of this system property should not exceed 32 characters.

2.4 Setting up a Weblogic Cluster for running OHI Product Definition on multiple nodes

A WebLogic Server cluster consists of multiple WebLogic Server Managed server instances running simultaneously and working together to provide increased scalability and reliability. A cluster appears to clients to be a single WebLogic Server instance. The server instances that constitute a cluster can run on the same machine, or be located on different machines.

A cluster's capacity can be increased by adding additional Managed server instances to the cluster on an existing machine, or by adding machines to the cluster to host the incremental Managed server instances. Each server instance in a cluster must run the same version of WebLogic Server.

Typically, the administration for the WebLogic Server instance is done through an Administration Server or Admin Server. The Managed Servers do not require the Administration Server to be up and running.

Prerequisites

Make sure that the following prerequisites are met before configuring a WebLogic cluster:

- Experience in setting up a WebLogic Server cluster is required!
- The WebLogic software needs to be installed on all the machines that will be part of the cluster (that will run WebLogic server instances). Make sure that the same version of the WebLogic software is installed on all nodes.

The OHI Product Definition release bundle contains scripts that may be used to automate the creation of a WebLogic Cluster.

Using these requires experience in setting up a WebLogic Server cluster. The scripts are located in **<OHI_ROOT>\util\wlst.** Note that the scripts are provided "as is".

Before using the scripts, stage these to the environment in which they will be applied and make sure that the scripts can be executed.

If the cluster setup is for a distributed environment, make sure to stage the files on a shared disk so that all machines can access these.

Before executing the steps to create a WebLogic Cluster, the following must be done in preparation:

- Change the **setEnv.sh** script to match the settings of the environment in which the scripts will be applied, e.g. set the correct Middleware Home (MW_HOME) and reference a Java Home.
- Populate the **wlst\properties\createOHIDomain.properties** file with the values for the desired setup.

The OHI Domain creation script supports the following Domain Topologies:

1. Admin Server only
2. Admin Server + single Managed Server (single host)
3. Admin Server + single Managed Server (distributed)
4. Admin Server + multiple Managed Servers (single host)
5. Admin Server + multiple Managed Servers (distributed)
6. Admin Server + multiple Clustered Managed Servers (single host)
7. Admin Server + multiple Clustered Managed Servers (distributed)

Sample configuration files are provided in **<OHI_ ROOT>\util\wlst\properties\samples** for all Domain Topologies mentioned.

Steps for setting up a Weblogic Cluster

Perform the following steps for setting up a WebLogic Cluster:

- Set up a Node Manager on all hosts in the cluster.
- Create a WebLogic domain for OHI Product Definition.
- Generate node manager boot & startup properties.
- Register the Domain with the Node Manager.
- Optional: Create WebLogic Domain Template for secondary hosts.
- Set up a load balancer to distribute requests to different managed servers in the cluster.

Before putting the domain into production, make sure that the environment is secure. See the specific WebLogic documentation with respect to "Securing a Production Environment".

Starting and stopping WebLogic Server is covered in the Operations Guide.

Setup a Node Manager for all nodes in the cluster

This step must be performed on all hosts (primary and secondary) that will be part of the WebLogic Server domain.

Node Manager is a WebLogic Server utility that controls start, shut down, and restart of Administration Server and Managed Server instances from a remote location. A Node Manager process is not associated with a specific WebLogic domain but with a machine. The same Node Manager process can be used to control server instances in any WebLogic Server domain, as long as the server instances reside on the same machine as the Node Manager process. Node Manager must run on each computer

that hosts WebLogic Server instances -whether Administration Server or Managed Server- that need to be controlled with Node Manager.

Before a domain is created set up a Node Manager.The Node Manager will run as "init.d" service. Use the **<OHI_ROOT>\util\wlst\registerNodeManagerService.sh** script (as ***root***) to create the nodemgrservice file and to set the correct property values in the nodemanager.properties file:

- StopScriptEnabled=true
- CrashRecoveryEnabled=true
- StartScriptEnabled=true

All scripts are driven from properties for which the values are specified in the **<OHI_ROOT>\util\wlst\properties\createOHIDomain.properties** file.

Create a Weblogic domain for OHI Product Definition in the cluster

This step must be performed on the primary host only.

Use the WebLogic Configuration Wizard to create a domain for OHI Product Definition. Alternatively, use the ***<OHI_ROOT>\util\wlst*** script (as ***oracle*** user).

Oracle suggested values for configuration of the WebLogic Cluster are listed in the following table:

Parameter	Suggested Value
Domain Name	ohi_domain
Administration Server Name	ohi_admin_server
Managed Server Name(s)	ohi_managed_serverX (where X is an integer value that starts with 1)
Cluster Name	ohi_cluster

Note that these values can be set in the **<OHI_ROOT>\util\wlst\properties\createOHIDomain.properties** file.

Required setting: make sure that the Server Start Mode for the domain is set to *Production Mode*.

Generate node manager boot & startup properties

Make sure that the server is up and running.

The easiest way to do is by using the ***<OHI_ROOT>\util\wlst*** script (as ***oracle*** user). Make sure that the **<OHI_ROOT>\util\wlst\properties\createOHIDomain.properties** file has all required values.

The generateNMPropsOHIDomain.sh script needs to be executed from the root directory of the WebLogic domain that was created.

Verify that the boot.properties and startup.properties files were created correctly for all server instances and in the proper location ($DOMAIN_HOME/servers/[SERVER_NAME]/data/nodemanager.

Regsiter the Domain with the Node Manager

Make sure that the server is up and running.

Enroll the domain (i.e. register the domain with the node manger service) by running **<*OHI_ROOT*>*util**wlst*\enrollOHIDomain.sh** (as **oracle** user). Verify that the enroll operation was successful, by checking the script output for "Successfully enrolled…".

The enrollOHIDomain.sh script needs to be executed from the root directory of the WebLogic domain that was created.

Optional: Create WebLogic Domain Template for secondary hosts

This step is only required if Managed Servers are defined that run on other hosts than the Admin Server.

Execute the "pack" command to create a WebLogic Domain Template for all secondary host machines. Alternatively, use the **<*OHI_ROOT*>*util**wlst*** script to do that. The script requires the fully qualified root directory of the WebLogic domain that was created as an input parameter. Transfer the generated WebLogic Domain Template to all secondary host machines. The template can now be removed from the primary host.

On any secondary host machine, use the "unpack" command to create the WLS Domain Directory. Alternatively, , use the **<*OHI_ROOT*>*util**wlst*** script to do that. The script requires two arguments:

- a reference to the generated WebLogic Domain Template
- the fully qualified root directory of the WebLogic domain that was created as an input parameter

Set up a Load Balancer

A load balancer is needed to distribute incoming requests to the participating nodes in the cluster. Details about configuration of load balancers can be found in Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server .

OHI Product Definition requires that HTTP session "stick" to the same node; that needs to be supported by the load balancer. OHI Product Definition maintains the session by sending a cookie to the client. The name of the cookie is OHIPRODDEFSESSION.

OHI Product Definition does not support HTTP Session state replication.

Final steps

The domain is almost ready to deploy the application. Perform these final steps before deployment:

- Set the ohi-proddef.properties, logback.xml and ohi-security.config files.
- Configure the Coherence cluster.

Before putting a domain into production, make sure that the environment is secure. See the specific WebLogic documentation with respect to "Securing a Production Environment".

2.5 Initial configuration for OHI Product Definition in Oracle Fusion Middleware

2.5.1 Logging configuration

OHI Product Definition makes use of Logback library for generating log output. That log output is controlled by logback.xml file that is referenced in the WebLogic Server configuration. Through the configuration file, the logging level can be controlled as well as the output channels (referred to as 'appenders') for log messages. An example of an output channel for logging is a file.

Predefined logging configurations

OHI Product Definition comes bundled with a number of predefined logback configurations:

- logback.xml: a default logging config file.
- production-logback.xml: for maximum performance, will reveal errors.
- trace-logback.xml: provides trace-level output (most detailed).

By default, logback.xml is used. To use one of the others, use the *-Dlogback.configurationFile* Java option in the setDomainenv script:

```
-Dlogback.configurationFile=production-logback.xml
or
-Dlogback.configurationFile=trace-logback.xml
```

The OHI Product Definition Operations Guide describes log files and how to control log output.

2.5.1.1 Logging Configuration For Web Services

To enable logging web services request and response, enable debug logging on DefaultServerSOAPHandler class. Add the following entries in logback.xml:

```
<logger
name="com.oracle.healthinsurance.support.ws.handlers.MessagePayloadLoggingHandler"
level="debug" />
```

2.5.2 Setup required defaults

The application requires default settings for a number of objects. Before default settings can be applied, users must be provisioned in order to access the system. Make sure the following prerequisites are met:

- Set up users in an LDAP Directory Server as outlined below.
- Provision the users in OHI Product Definition. For this purpose, a Provisioning service is provided

2.5.2.1 Set up a directory for File Exchange

In a number of scenarios OHI Product Definition processes files, for example for the File Import integration points. It is recommended to set up a shared directory structure that can be accessed by any machine that executes the system.

For example:

- For inbound files: /<MOUNT_POINT>/ohi-proddef/transfer/in

- For outbound or response files: /<MOUNT_POINT>/ohi-proddef/transfer/out

2.5.2.2 Authentication and User Provisioning

Before users can access the OHI Product Definition application, the following prerequisites must be met:

- Users need to authenticate themselves by entering a valid combination of username and password credentials. All pages (other than the login page) are only available to authenticated (and properly authorized) users.
- A user must be provisioned to access the OHI Product Definition application. The main purpose of OHI Product Definition user accounts is authorization: the administration of (role-based) access rights for users is handled in the OHI Product Definition application.

The following paragraphs provide details on authentication and provisioning.

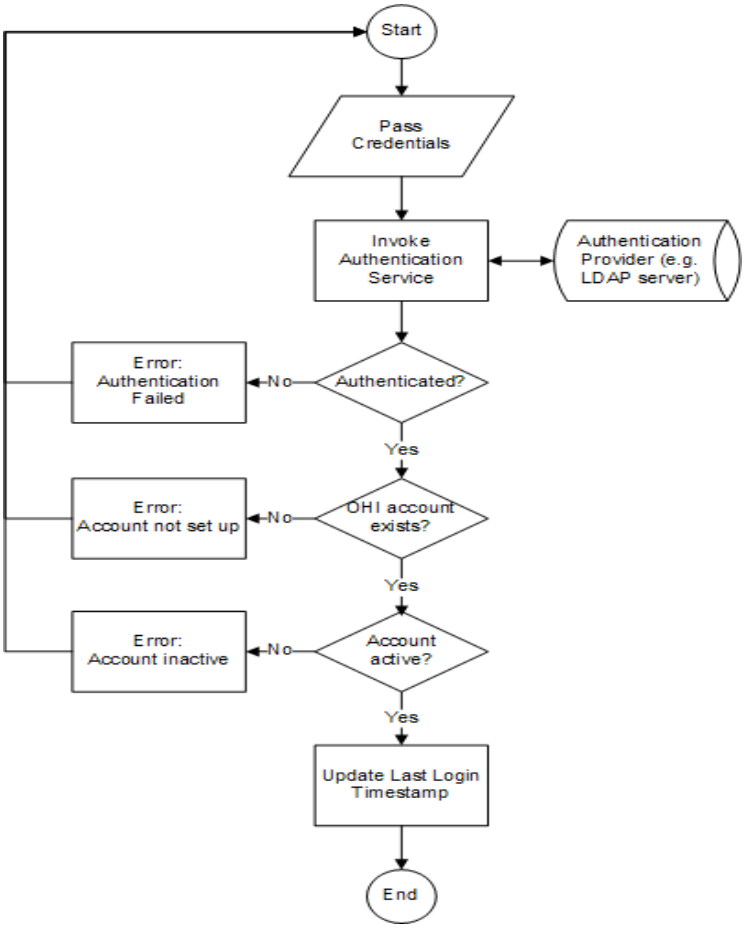
2.5.2.3 Authentication

Although user accounts are stored in the application, user passwords are not. As a result, the application relies on external services for authentication. It provides support for LDAP based authentication (LDAP version 3).

The application supports LDAP authentication by binding to the LDAP server using the user-supplied credentials. This way, no LDAP-specific account info needs to be stored in OHI Product Definition.

Users in the LDAP server are expected to be defined using the industry standard *inetOrgPerson* object class (which is derived from the *organizationalPerson* object class). Typically, in that class, the properties *uid* and *userpassword* are used to store the credentials used for logging in.

The following picture shows the flow of the authentication process:



Credentials are passed by the user via the OHI Product Definition Login page.

In the authentication process, the user account data that is stored in OHI Product Definition is accessed, for example for logging the last time the user successfully logged in to the system. Before someone can authenticate and subsequently access OHI Product Definition, an account has to be set up. For that purpose, OHI Product Definition offers a user provisioning service which is documented in the Integration Guide.

2.5.2.4 Internal System User

During installation, an acount for the Internal System User is created in the OHI_USERS table with the following characteristics:

- ID=10
- IND_ACTIVE=Y
- DISPLAY_NAME='Internal System User'
- LOGIN_NAME=null

This user cannot be used to log in to the application via the UI pages, because the LOGIN_NAME is null. The Internal System User is used for the internal processing. For example, records created or updated by an Integration Point, will have CREATED_BY and/or LAST_UPDATED_BY = 10 (the id of the internal system user).

2.5.2.5 Seeded access roles

As said in the previous section, the seeded Internal System User cannot be used to log in to the application to use the UI pages. So after installation, new users should be created with appropriate roles.

There is a bootstrap issue here: new roles should be defined first in the OHI Product Definition application using the Setup access role page. To be able to access the setup access role page, a user should exists with a role that gives access to this page.

To solve the bootstrap issue, role SETUP_ACCESS_ROLE is seeded during installation as follows:

Access Role Attribute	Value
Code	SETUP_ACCESS_ROLE
Name	Setup Access Role
Description	System role that gives access to setup access role page only.
Active	Y
Enabled	Y
Ohi specific?	Y

Table 2–1 Table 2-1: Access Restriction Grants for SETUP_ACCESS_ROLE

Access Restriction Grant Attribute	Value
Access Restriction	AccessRoles
Create?	Y
Retrieve?	Y
Update?	Y
Delete?	Y
OHI specific?	Y

So the role SETUP_ACCESS_ROLE gives access to the setup access role page only.

After installation, the following steps needs to be taken to setup a new user with the SETUP_ACCESS_ROLE granted:

1. Create a new access role SETUP_ACCESS_ROLE in the external identity store.
2. Create a new user in the external identity store and grant the SETUP_ACCESS_ROLE to that user.
3. Provision the user with the SETUP_ACCESS_ROLE granted to the OHI Product Definition application.

For explanation of these steps, see 'Function Authorization'.

To facilitate testing, role ALL_FUNCTIONS_ACCESS_ROLE is seeded also. This role gives access to all pages of the application. This role is not intended to be used in production environments, so this role is disabled by default.

Access Role Attribute	Value
Code	ALL_FUNCTIONS_ACCESS_ROLE

Access Role Attribute	Value
Name	All Functions Access Role
Description	System role that gives access to all pages (disabled by default)
Active	Y
Enabled	N
Ohi specific?	Y

Table 2–2 Table 2-2: Access Restriction Grants for ALL_FUNCTIONS_ACCESS_ROLE

Access Restriction Grant Attribute	Value
Access Restriction	All access restrictions of type 'Function'
Create?	Y
Retrieve?	Y
Update?	Y
Delete?	Y
OHI specific?	Y

After installation, the following steps needs to be taken to setup a new user with the ALL_FUNCTIONS_ACCESS_ROLE granted:

1. Create a new access role ALL_FUNCTIONS_ACCESS_ROLE in the external identity store.
2. Create a new user in the external identity store and grant the ALL_FUNCTIONS_ACCESS_ROLE to that user.
3. Provision the user with the ALL_FUNCTIONS_ACCESS_ROLE granted to the OHI Product Definition application.
4. Enable to access role ALL_FUNCTIONS_ACCESS_ROLE.

3

Release Installation

In this chapter, the generic process for Installing an OHI Product Definition release is described. Release specific instructions are documented in the Release Notes for that specific release.

3.1 Install database objects

3.1.1 Change Installation Configuration

1. In **<OHI_ROOT>\util\install**, make a copy of **ohi_install.cfg.template** and name it **ohi_install.cfg**.
2. Edit **ohi_install.cfg** to contain your specific database connection data and other configuration settings. The settings are explained in the file itself.

By default, the schema passwords will be similar to the schema user names. The ohi_install.cfg files allows the specification of different passwords. Alternatively, specify empty string passwords to have the option of entering the passwords at the command prompt. In the latter case, the passwords will not end up in a configuration file.

The tablespaces mentioned in the ohi_install.cfg should exist, prior to running the installation.

Default schema passwords should not be used.

Oracle recommends that schema passwords are entered at the command prompt. Never store passwords in configuration files.

3.1.1.1 Configure Instance Discriminator

In accordance with the concepts explained in the paragraph Enabling Replication of Setup Data, the correct environment or instance must be configured during a new installation. The data that is entered for ohiInstances in the ohi_install.cfg file is stored in the database when a fresh install is performed. Make sure to assign unique discriminator values for each environment.

3.1.2 Run Installer

1. Open a command window and browse to **<OHI_ROOT>\util\install**.
2. Depending on the O/S you should run ohi_install.bat (Windows) or ohi_install.sh (Linux). This will assume that the config file ohi_install.cfg is present in the same directory where ohi_install.bat is present and uses the default configuration from

ohi_install.cfg. To specify a different location of ohi_install.cfg or to specify a different environment from ohi_install.cfg, follow the next step.

3. Specify the command line options to specify the location of ohi_install.cfg file and environment to use from ohi_install.cfg.
   - Eg: **ohi_install.sh -c /home/oracle/someLocation/ohi_install.cfg -e dev**
4. The command line arguments are explained below:

Option		Arguments	Description
Short	Long		
-c	--cfg	config file path	The location of the configuration file. Default is ohi_install.cfg
-e	--env	environment names	The name that specifies which of the environment settings from the config file to use

3.1.2.1 Install Seed Data

Part of the database objects installation is the installation off Seed Data.

Types of Seed Data

3.1.2.1.1 Generic Seed Data

Seed data is maintained by Oracle. Customers should not change this data except the updatable columns. It is delivered as part of a release and may be updated by software upgrades.

3.1.2.1.2 Localization Seed Data

This category covers specific data that is required by localizations. The data is maintained by Oracle. Customers should not change this data. It is delivered as part of a release and may be updated by software upgrades.

3.1.2.1.3 Sample Data

Sample data is provided by Oracle to give you a headstart during configuration. This is data configured by Implementation Consultants. It is *not* modified during future upgrades. For more information, see Appendix B - Seed Data

3.1.2.1.4 Restrictions on using Seed Data

Because Seed Data is maintained by Oracle, it may be modified or even deleted as part of an upgrade. Customers should therefore exercise caution when using seed data in their configuration by abiding these rules.

1. Do not remove (delete) Seed Data rows. A patch may re-insert the row.
2. Do not update columns, other than those indicated as updateble below.
3. Do not make references to rows that may be deleted by Oracle (see table below).

Violations of the rules above (especially rule 3) may lead to failures during the installation of upgrades.

The table below lists the Seed Data tables.

- **Data**: The table or logical entity

- **Updateable columns**: The customer may update the values in these columns. They will not be overwritten by upgrade scripts. Other columns should not be updated by the customer.
- **Physical Delete**: Upgrade scripts may delete this data. The customer should not create references to this data.

Data	Updateable Columns	Physical Delete	Remarks
Access Restrictions		Yes	Also deletes Access Restriction Grants referring to this row
Access Restriction Grants		Yes	
Access Roles	ind_enabled	No	Two roles are seeded
Boilerplate Texts		Yes	
Country Regions		No	
Data set definitions		Yes	
Dynamic Field Usages		No	
Dynamic Logic		No	
Fields (+ dynamic logic)		No	
Flex Codes		No	
Flex Code Sets (+ details)		No	
Flex Code Systems		No	
Languages	ind_default ind_ installed	No	
Messages	ind_suppress_log_in_ ui ind_suppress_log_ in_ext ind_mark external_code	Yes	
Single Flex Code Definitions (+ usage)		No	
Task Types		Yes	Customer is not allowed to change anything in base table
Task Type Attributes	value_char value_ number value_ datetime value_clob	Yes	
Users		No	One User will be seeded (system user)

3.1.2.2 Enable Total Recall (optional)

When Total Recall Option is activated, you should decide if one or more of the new tables should be added to a Flashback Data Archive.

Syntax to enable history tracking for a table is:

```
ALTER TABLE <tablename> FLASHBACK ARCHIVE [<Flashback Data Archive name>];
```

Note that the FDA name is required only when adding the table to a non-default FDA.

To disable history tracking for a table use:

```
ALTER TABLE <tablename> NO FLASHBACK ARCHIVE;
```

3.2 Install Application

This section lists the steps that are required to install the OHI application on the Oracle Fusion Middleware WebLogic Server (WLS).

3.2.1 Creating WebLogic Work Managers

By default, WebLogic Server uses the *default* work manager to handle thread management and perform self-tuning. This *default* Work Manager is used by an application when no other Work Managers are specified in the application's deployment descriptors. For more information, check the WLS documentation: http://docs.oracle.com/cd/E17904_01/web.1111/e13701/self_tuned.htm.

However, it is recommended to use the following application-specific work managers:

- a work manager to control UI requests, named "wm/ui-work-manager"
- a work manager to control Web Services requests, named "wm/ws-work-manager"
- and a work manager to control task processing (like File Import batch processing), named "wm/core-work-manager"

This allows more fine-grained control and work load monitoring of the system.

These work managers are configured by the WLST scripts as part of creating a new domain. For an existing domain the work managers need to be configured manually. If these work managers are not configured, the system issues warnings at startup like the following example and will use the WLS default work manager instead:**<Warning> <WorkManager> <BEA-002919> <Unable to find a WorkManager with name wm/core-work-manager. Dispatch policy wm/core-work-manager will map to the default WorkManager for the application.**

Create global work managers through the Administration Console using the steps that are listed in the following table in the given order. Make sure to associate the work managers with the managed servers.

Step	Configuration	Value
1	Minimum Threads Constraint	
	Name	core-work-manager-min-threads-constraint
	Count	16
2	Maximum Threads Constraint	
	Name	core-work-manager-max-threads-constraint
	Count	16
3	Fair Share Request Class	

Step	Configuration	Value
	Name	core-work-manager-fair-share-req-class
	fair Share	40
4	Work Manager	
	Name	wm/core-work-manager
	Request Class	core-work-manager-fair-share-req-class
	Minimum Threads Constraint	core-work-manager-min-threads-constraint
	Maximum Threads Constraint	core-work-manager-max-threads-constraint
	Capacity Constrain	None Configured
	Ignore Stuck Threads	Checked
5	Minimum Threads Constraint	
	Name	ws-work-manager-min-threads-constraint
	Count	10
6	Maximum Threads Constraint	
	Name	ws-work-manager-max-threads-constraint
	Count	100
7	Fair Share Request Class	
	Name	ws-work-manager-fair-share-req-class
	Fair Share	50
8	Work Manager	
	Name	wm/ws-work-manager
	Request Class	ws-work-manager-fair-share-req-class
	Minimum Threads Constraint	ws-work-manager-min-threads-constraint
	Maximum Threads Constraint	ws-work-manager-max-threads-constraint
	Capacity Constraint	None Configure
	Ignore Stuck Threads	Checked
9	Minimum Threads Constraint	
	Name	ui-work-manager-min-threads-constraint
	Count	5
10	Maximum Threads Constraint	
	Name	ui-work-manager-max-threads-constraint
	Count	100

Step	Configuration	Value
11	Fair Share Request Class	
	Name	ui-work-manager-fair-share-req-class
	Fair Share	50
12	Work Manager	
	Name	wm/ui-work-manager
	Request Class	ui-work-manager-fair-share-req-class
	Minimum Threads Constraint	ui-work-manager-min-threads-constraint
	Maximum Threads Constraint	ui-work-manager-max-threads-constraint
	Capacity Constraint	None Configured
	Ignore Stuck Threads	Checked

After configuring the work managers, managed servers need to be restarted.

The work manager configuration, like the minimum and maximum threads constraints and the fair share request factors, can be modified at any time in WLS Admin Console as is required, for example to increase the number of threads used for task processing.

This largely depends on hardware capacity, system configuration and the load characteristics and typically requires a thorough understanding of the system's performance.

3.2.2 Configuring OID Authentication Provider

The application uses a WebLogic Authentication Provider to connect to Oracle Internet Directory (OID) or to a third party LDAP server. This section describes the configuration of an OID or third party LDAP Authentication Provider.

Alternatively, for creating a new WebLogic domain for OHI Product Definition use the WLST scripts for setting up the Authentication Provider.

Step 1: Login to WLS admin console and click on **Security Realms** link.

Step 2: Click on **myrealm** link.

In WLS Production-mode use the **Lock & Edit** button before clicking on the **New** button.

Step 3: Click on **Providers** tab.

Step 4: Click on **New** button.

Step 5: Change **Name** and **Type** to **OHIProdDefAuthenticationProvider** and **OracleInternetDirectoryAuthenticator** (or to **LDAPAuthenticator** in case a third party LDAP server is used) respectively in **Create a new Authentication Provider page**. Click on **OK** button.

Step 6: Click on **OHIProdDefAuthenticationProvider** link.

Step 7: Change the **Control Flag** to **SUFFICIENT** and click on **Save** button.

Step 8: Click on **Provider Specific** tab.

Step 9: Enter/change the values for various fields as shown below and select the option **Propagate Cause For Login Exception**. Click on **Save** button.

Field	Value
Host	LDAP hostname or IP address
Port	LDAP Port or SSL Port if the LDAP is SSL enabled. E.g.: 3060. In case LDAPS is used, make sure to check the SSLEnabled flag as well.
Principal	LDAP admin principal: E.g.: cn=orcladmin
Credential	LDAP admin password
Confirm Credential	LDAP admin password
User Base DN	User Base distinguished name. E.g.: ou=Users,dc=healthinsurance,dc=oracle,dc=com
All Users Filter	E.g.: (&(uid=*)(objectclass=person))
User From Name Filter	E.g.: (&(uid=%u)(objectclass=person))
User Name Attribute	E.g.: uid
Group Base DN	If there are no groups in the LDAP, leave this field empty.

There are a few more properties (or fields in the page) which are not mentioned in the table above. Change the values of those fields to suit your LDAP settings.

Step 10: Click on **myrealm** link and then **DefaultAuthenticator** link. Change the **Control Flag** to **SUFFICIENT** and click on **Save** button.

Step 11: Restart the WebLogic Server.

Optionally, verify that the authentication provider is configured successfully (after the WebLogic Server is restarted) by following the steps mentioned below:

Step 1: Login to WLS Admin Console and click on **Security Realms**

Step 2: Click on **myrealm**

Step 3: Click on **Users and Groups** tab

Step 4: You should be able to see the list of users from **OHIProdDefAuthenticationProvider** (in addition to the default users from **DefaultAuthenticator**).

3.2.3 Set up JDBC Data Sources

The application connects to the Oracle database through a Data Source that need to be specified in the WLS Server.

For security reasons, the database connections used by the application connect to database schemas that do not own database objects. These schemas are only granted the required privileges to use the objects.

The following sections describe setting up data sources for connecting to:

- an Oracle database that is running on a single machine
- a RAC-enabled Oracle database that is running on multiple machines

3.2.3.1 Data Source for connecting to an Oracle database that is running on a single machine

The following table lists the Data Source that must be configured in WLS before installing the application for use with an Oracle database that is executed on a single machine (not clustered):

Data Source Parameters	Non-clustered database	Explanation
Data Source Name	ohi-application-datasource	Logical name
JNDI Name	jdbc/proddefUserOhiApplicationDS	Used by the application to resolve the Data Source
Database Type	Oracle	
Database Driver	Oracle's Driver (Thin) for Instance connections; Versions:9.0.1,9.2.0,10,11 or Oracle's Driver (Thin) for Service connections; Versions:9.0.1,9.2.0,10,11	
Database Name	SID or service name of the database If the name of the Oracle driver that was selected contains the words "for Instance connections" enter the SID. If the name of the Oracle driver contains the words "for Service connections" enter the service name.	
Host Name	Name or IP address of the machine where the database is running	
Port	Port on which the database is running	
Database User Name	ohi_proddef_user	Fixed value, do not change
Password & Confirm Password	Password of "ohi_proddef_user"	The schema password as selected during the installation

The data sources can be created by either

1. using the **<OHI_ROOT>\util\wlst\createOHIDomain.sh** script (i.e. the data sources are created at the time the domain is created) or
2. creating them through WLS Admin Server console (see sample below).

3.2.3.2 Data Source for connecting to an Oracle RAC database that is running on multiple machines

To support Oracle RAC features within Oracle WebLogic Server, Oracle recommends using Oracle WebLogic Server **GridLink Data Source**. A single GridLink data source provides connectivity between WebLogic Server and an Oracle Database service targeted to an Oracle RAC cluster. It uses the Oracle Notification Service (ONS) to adaptively respond to state changes in an Oracle RAC instance. An Oracle Database

service represents a workload with common attributes that enables administrators to manage the workload as a single entity.

To configure this, the following steps need to be performed. For more details about GridLink Data Source configration, see the Oracle WebLogic Server documentation: http://download.oracle.com/docs/cd/E17904_01/web.1111/e13737/gridlink_datasources.htm.

3.2.3.2.1 Configuring GridLink Data Source

Step 1: Login to WLS admin console and click the **Services/Data Sources** link.

Step 2: Click on **New** button and select the option **GridLink Data Source**

In WLS Production-mode use the **Lock & Edit** button before clicking on the **New** button.

Step 3: Change the value of **Name** to **ohi-application-datasource** and enter **jdbc/proddefUserOhiApplicationDS** in **JNDI Name**. Click the **Next** button.

Step 4: In **Transaction Options** page, accept the default settings (Supports Global Transactions and One-Phase Commit) and click the **Next** button.

Step 5: If SCAN (Single Client Access Name) is used for the Oracle RAC database, select the option **Enter complete JDBC URL**. Else, select the option **Enter individual listener information**.

Step 6: In **Connection Properties** page either

- enter the values of various fields as outlined in the table below if option **Enter complete JDBC URL** is selected:

Parameters	Value	Explanation
Complete JDBC URL	jdbc:oracle:thin:@{scan-listener-host}:{scan-listener-port}/{service-name}	JDBC URL using SCAN
Database User Name	ohi_proddef_user	Fixed value, do not change
Password & Confirm Password	Password of "ohi_proddef_user"	The schema password as selected during the installation

- or enter the values of various fields as outlined in the table below if option **Enter individual listener information** is selected:

Parameters	Value	Explanation
Service Name	Oracle RAC service name	
Host and Port	hostname1:port hostname2:port	Individual RAC node details. The format is <HOSTNAME>:<PORT>
Database User Name	ohi_proddef_user	Fixed value, do not change
Password & Confirm Password	Password of "ohi_proddef_user"	The schema password as selected during the installation

Step 7: In **Test GridLink Database Connection** page, click on **Test All Listeners** to see if the connection is successful. Once the test connection succeeds, click on **Next** button.

Step 8: Enter the details of ONS client configuration as outlined in the table below and click the **Next** button.

Parameters	Value	Explanation
Fan Enabled	Check-box selected	Enables the data source to subscribe to and process Oracle FAN events. This attribute is only applicable for RAC configurations that publish FAN notification events using the ONS protocol.
ONS Nodes	Eg: hostname1:6200,hostname2:6200	A comma-separated list of ONS daemon listen addresses and ports to connect to for receiving ONS-based FAN events.
ONS Wallet File	Location of ONS Wallet File (including the file name)	The location of the Oracle wallet file in which the SSL certificates are stored. Only required when the ONS client is configured to communicate with ONS daemons using SSL.
ONS Wallet Password & Confirm ONS Wallet Password	The wallet password	The wallet password attribute that is included as part of the ONS client configuration string. This attribute is only required when ONS is configured to use the SSL protocol.

Step 9: Click on **Test All ONS Nodes** to see if the connection is successful. Once the connection test succeeds, click the **Next** button.

Step 10: Select the Target(s) in the next page and click the **Finish** button.

Make sure to specify the managed server as target for the GridLink Data Source and change the connection pool settings by executing the following steps:

1. Select the newly created GridLink Data Source
2. Click on the tab **Connection Pool**
3. Expand the **Advanced** node at the bottom of the page to display all properties and set the following:

Property	Value
Initial Capacity	0
Test Connections On Reserve	Checked
Test Frequency	300
Connection Creation Retry Frequency	30
Seconds to Trust an Idle Pool Connection	10

Set the following driver property:

Prperty	Value
oracle.net.CONNECT_TIMEOUT	10000

3.2.4 Installing the UI Customization Library through WLS Admin Server Console

To enable the creation of site-level UI Customizations, without having to change the OHI Application itself, an initially empty library called custom.oracle.healthinsurance needs to be installed before the OHI application can be installed.

Step 1: Login to the Admin Server console (for example: http://machine.domain:port/console).

Step 2: Click the "**Deployment**" link and then click on the "**Install**" button. If the Install button is disabled, click the Lock & Edit button first (in the upper left section of the page).

Step 3: Select the path where the library **custom.oracle.healthinsurance.war** file is located (**<OHI_ROOT>\lib**) and click the "**Next**" button.

Step 4: Select the option "**Install this deployment as a library**" and click on "**Next**" button.

Step 5: Ensure that the General - Name is set to **custom.oracle.healthinsurance**. This is the name the OHI Application refers to when loading the library, so this is the name under which it must have been installed. The OHI Application will automatically load the highest version of all installed libraries with this deployment name. Then click on the **"Next"** button.

Step 6: Click on "**Finish**" button. You should see a success message. The library is now installed. Note: if you had to click Lock & Edit in step 1, you now have to click Activate Changes

Installing the Jersey Library through WLS Admin Server Console

Please follow the above steps to install the jersey libararies through weblogic console. The libraries to be installed are packaged along with weblogic and can be found at location: <WLHOME>/Middleware/wlserver_10.3/common/deployable-libraries.

The name of the libraries are following:

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

3.2.5 Installing The OHI Product Definition Application Through WLS Admin Server Console

The OHI applications are delivered in a so called Java Enterprise Archive (EAR) which will be installed through the WLS Admin Server Console. In order to do that, perform the following steps.

Step 1: Login to the Admin Server console (for example: http://machine.domain:port/console).

Step 2: Click the "**Deployment**" link and then click on the "**Install**" button.

Step 3: Select the path where the EAR file is located and click the "**Next**" button.

Step 4: Select the option "**Install this deployment as an application**" and click on "**Next**".

Step 5: Click on "**Finish**" button. The OHI Application is now installed.

Step 6: If you are deploying the application to cluster, in "**Select deployment targets**" page, select the Clusters target.

3.2.6 Changing the Context-Root for UI or Web Services

To change the default context-root of OHI Product Definition web application (which is "/proddef") or web services deploy the application with a customized deployment plan.

Perform the following steps for a new deployment of the application:

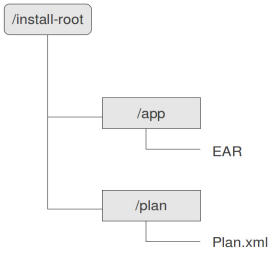
Step 1: Edit the value of the variable **UI_CONTEXT_ROOT** in **<OHI_ROOT>/application/plan/Plan.xml** to suit your requirements.

Step 2: The following example shows how the context-root for a web service can be changed using the deployment plan (the example does so for the context-root of the File Import web service):

```
...

<variable>
     <name>FILEIMPORT_CONTEXT_ROOT</name>
     <!-- Here claims is the new context root which will overwrite the default
context root -->
     <value>new-ohi-common-ws-fileimport</value>
</variable>

...

<module-override>
     <!-- Copy the name of the EAR (including .ear file extension) from <OHI_
ROOT>/application/app/ dir -->
     <module-name>NAME_OF_THE_EAR</module-name>
     <module-type>ear</module-type>
     <module-descriptor external="false">
         <root-element>application</root-element>
         <uri>META-INF/application.xml</uri>
         <variable-assignment>
             <name>FILEIMPORT_CONTEXT_ROOT</name>
             <!-- Here ohi-common-ws-fileimport refers to the default context
root.
                  See the appendix of the Installation Guide for the default
context roots of web services -->
<xpath>/application/module/web/[context-root="ohi-common-ws-fileimport"]/context-r
oot</xpath>
             <operation>replace</operation>
         </variable-assignment>
     </module-descriptor>
</module-override>

--
```

The module names for the web services are listed in an appendix.

Step 3: The EAR and Plan.xml (deployment plan) are packaged under a directory named "**application**" in the release bundle (See the directory structure below). It is recommended to copy the "**application**" directory to a location (this directory will be referred as ***<INSTALL-ROOT>*** hereafter) and optionally rename the directory (for example, rename to OHIProdDef).



Step 4: To install the application using Administration Console, select the directory ***<INSTALL-ROOT>*** instead of selecting the EAR file. By default, the Administration Console will use a deployment plan named Plan.xml, if one is available in the \plan subdirectory.

3.2.6.1 Changing OHI Product Definition Session Timeout

OHI Product Definition application does not ship with default session timeout. Instead, it leverages WebLogic Server's default session timeout - which is **3600 seconds** (1 Hour). It is possible to change this default session timeout value through WebLogic Server's Admin Console. Follow the steps below to change the default session timeout.

Follow the section **Changing the Context-Root for UI or Web Services** to deploy OHI Product Definition application in order to change the session timeout through WebLogic Server Admin Console.

Step 1: Login to WebLogic Server Admin Console

Step 2: Click on **Deployments** link and expand OHI Product Definition application tree

Step 3: Click on the name of the UI application (by default it is **proddef** unless the context-root is changed as mentioned in the section **"Changing the Context-Root for UI or Web Services"** in **Modules** section.

Step 4: Make sure that the name of the module is **ohi-proddef-ui.war**

Step 5: Click on **Configuration** tab and change the default **Session Timeout (in seconds)** from 3600 seconds (1 Hour) to suit your needs and click on **Save** button.

You may get into trouble if the load balancer session timeout is shorter than WebLogic session timeout. So, it is important to set load balancer session timeout in align with WebLogic session timeout

Step 6: Click on **Deployments** link and select OHI Product Definition application. Click on **Update** button

Step 7: Select the first option in **Update Application Assistant** and click on **Finish** button.

Step 8: After activating changes, restart WebLogic Server.

3.3 Validate Installation

Validate the installation by performing the following steps:

Note: if users have not been provisioned yet, the application cannot be accessed.

1. Point a web browser to the home page of the application and verify that a login screen is displayed. The URL for the home page is http://machine.domain:port/proddef.
2. Using a web browser, verify that the Web Service WSDL's are available. The URL's for the accessing the Web Service WSDL's are listed elsewhere in this guide.

3.4 Configuring OHI Product Definition properties file

A changed version of the ohi-proddef.properties file may be delivered in a new OHI Product Definition release.

The following tables describe the properties that are maintained in this file.

Category	Parameter	Value	Explanation
File Import	ohi.ws.fileimport.filesrootdirectory	.	Directory paths used for File Import will be prepended with the given root directory. This is for security reasons, it ensures that files are stored in a specific area only
Dynamic Logic	ohi.dynamiclogic.classes.directory	.	Path to directory in which the system generates Dynamic Logic classes

OHI Product Definition User Interface related properties

The following table lists other user interface related properties:

Category	Parameter	Value	Explanation
User Interface	ohi.environment.identifier	Samples: "User Acceptance Test", "Development".	Text string that is displayed on the home page of the system that helps the user to identify the environment
User Interface	ohi.ui.maxrowstoretrieve	Suggested default is 200.	Maximum number of rows retrieved to show in a UI table. Note that memory usage and page load times are impacted by this value.

Processing Related Properties

The following table lists Product Definition processing related properties:

Category	Parameter	Value	Explanation
Product Definition Processing	ohi.processing.fillthreshhold	**OPTIONAL** Positive integer value. Default 10	Suggested value is a multiple of the number of CPU cores available to the managed server. Determines the number of tasks that will be submitted for processing at any given time.
Product Definition Processing	ohi.processing.filldepth	**OPTIONAL** Positive integer value. Default 30	Suggested value is 1 less than number of CPU cores available to the managed server. Determines when the system will look for more tasks to be submitted for processing.
Product Definition Processing	ohi.processing.maxIncompleteAttempts	**OPTIONAL** Positive integer value. default is 100000	Number of times a task can resolve as "incomplete" before it's marked as in error.
Product Definition Processing	ohi.processing.maxErorAttempts	**OPTIONAL** Positive integer value. Default is 3	Number of times a task can resolve as "errored" before it stops a task flow.
Product Definition Processing	ohi.processing.attemptLogLevel	**OPTIONAL** Integer Value >= 0. Default is 0	A value of greater than 0 means data for failed task processing attempts will be retained.
Product Definition Processing	ohi.processing.retryimmediate	**OPTIONAL** Boolean [true,false]. Default is false	Determines if a failed task is retried immediately, or re-queued for another attempt after a delay.
Product Definition Processing	ohi.processing.defaultdelay	**OPTIONAL** Positive Integer. Default is 3	Default delay in seconds used when a failed task is re-queued for another attempt. Is overridden if a delay is set on the task type.