Oracle® Virtual Assembly Builder

User's Guide

11*g* Release 1 (11.1.1.6)

E22514-03

September 2012

ORACLE®

Oracle Virtual Assembly Builder User's Guide, 11*g* Release 1 (11.1.1.6)

E22514-03

Contents

2 Architecture

3 Security

4 Appliance and Assembly Structure

5 Using Oracle Virtual Assembly Builder

A Command Line Reference

B Oracle Virtual Assembly Builder Introspection Plug-ins

C Common Properties for Oracle Virtual Assembly Builder Components

D Troubleshooting

E Third-Party Licensing

Preface

This book details conceptual, topology and configuration topics about Oracle Virtual Assembly Builder. This Preface includes the following topics:

Audience

The intended audience is system administrators who will use Oracle Virtual Assembly Builder for their organizations.

What's New in This Release

This release of Oracle Virtual Assembly Builder contains the following new features or product enhancements:

- The introduction of a standalone Oracle Virtual Assembly Builder Deployer, a Web application that maintains a repository of assembly archives created by Oracle Virtual Assembly Builder Studio. Oracle Virtual Assembly Builder Deployer provides operations for registering these assembly archives to virtualized systems such as Oracle VM and provides operations for orchestrating the deployment of the software system defined by the assembly archive.
  - Oracle Virtual Assembly Builder Deployer includes a Web service API for use in creating custom applications that leverage Oracle Exalogic. The Web service API provides operations for uploading assembly archives, registering the assembly archive virtualization system and managing assembly instances for the system defined in the assembly archive. See *Oracle Virtual Assembly Builder Developer's Guide*.
  - It also includes the `abctl` command line interface that is built on top of the same Web Service API and provides all the same operations.
  - Oracle Virtual Assembly Builder Deployer provides role based access control where all assembly operations are authenticated and authorized.
- Oracle Virtual Assembly Builder now supports deployment of assemblies to:
  - Oracle VM 3.0, and

  - Oracle Exalogic
- Support for assembly archives as open virtualization archives (OVA). An assembly archive contains one or more VM images, and a metadata file (OVF) describing how they are associated with one another.
- Multiple network interfaces support: Appliances can now have multiple network interfaces defined and input endpoints can be selectively bound to a specific network interface or any.
- Multiple virtual network (Vnet) support: You can now define multiple virtual networks (Vnets) for an assembly and have appliances' network interfaces bound to different Vnets.
- Shared storage support: You can configure appliance's file sets as shared or local. If supported by the underlying deployment platform, you can specify that file sets be shared with individual appliances within assemblies. Shared storage devices in the deployment platform can be populated with the contents of the shared file sets as part of assembly deployment.
- Extended files ownership support: Within an appliance, files ownership permissions (operating system user and group) can now be defined at each file set level.
- Zero-count appliances: Ability to initially deploy an appliance within an assembly with zero appliance instances which will cause no VM instance for that appliance to be created. In subsequent scaling operations you could add appliance instances to those appliances that are part of the assembly configuration but were initially "deployed" with a zero-instance count.
- Custom appliance properties: Ability to add custom properties to an appliance that can be edited along with appliance's pre-defined properties during assembly editing and as part of deployment plan. With this feature you can configure, and/or operate a custom product or component that gets deployed with an Oracle product in an appliance.
- Anti-affinity support: You can specify the requirement to place multiple instances of a particular appliance across different physical hosts in the deployment target.
- Oracle Virtual Assembly Builder now supports the introspection and deployment of the following additional products:
  - Oracle SOA: Examines an existing SOA WebLogic Server deployment, capturing all the configuration in the SOA domain as well as the container configuration and deployed composites in MDS.
  - Oracle Service Bus: Examines and captures an Oracle WebLogic Server domain where Oracle Service Bus is configured with or without Oracle SOA.
  - Oracle Tuxedo: Examines and captures a single or multiple-machine Oracle Tuxedo domain, and the Oracle Home Directory that it resides on (including add-ons).
  - Oracle Traffic Director: Examines and captures a collection of configurable elements (metadata) that determine the run-time behavior of an Oracle Traffic Director instance.
  - Oracle RAC Database: Examines Oracle Clusterware and RAC Database components and captures their metadata.
  - A Generic Appliance: Ability to create and deploy an opaque, standalone, and self-contained product or application as an appliance for which Oracle Virtual Assembly Builder does not have in-built support.

  - External Appliances: a virtual machine template created outside of Oracle Virtual Assembly Builder and compatible with deployment on Oracle VM 3.0 can be imported into a Oracle Virtual Assembly Builder catalog as an external appliance. Once imported, external appliances can be edited, added and deployed as part of any assembly.

- A revised, more streamlined Oracle Virtual Assembly Builder Studio graphical user interface with an IDE look and feel. This includes a Structure Pane displaying the contents of Assemblies, Appliances and Deployments, a Deployment navigator, a broken out Deployment Plan navigator, as well as drag and drop support for registration and deployment.
- Integration with Oracle Enterprise Manager 12c: Oracle Virtual Assembly Builder Studio supports browsing, upload and download of assembly archives to/from Oracle Enterprise Manager Software Library.
- Flexible installation options: Allows you to install and configure the functionality you want. You can choose to install and configure just the Oracle Virtual Assembly Builder Studio, or just the Deployer or both on the same machine or different machines.

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.

Related Documents

For more information, see the following documents in the documentation set:

- *Oracle Virtual Assembly Builder Installation Guide*
- *Oracle Virtual Assembly Builder Developer's Guide*
- *Oracle Virtual Assembly Builder Release Notes*

Conventions

The following text conventions are used in this document:

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

1 Introduction

This chapter provides an introduction to the major concepts and components of Oracle Virtual Assembly Builder, and contains the following sections:

1.1 Introduction to Oracle Virtual Assembly Builder

Increased operating costs, inefficient hardware utilization and rapidly expanding data centers have made virtualization the most compelling IT technology in years. Virtualization for desktop and server environments has evolved to finally deliver on its promise to lower operating costs by increasing the utilization of hardware and reducing the overall amount of hardware required.

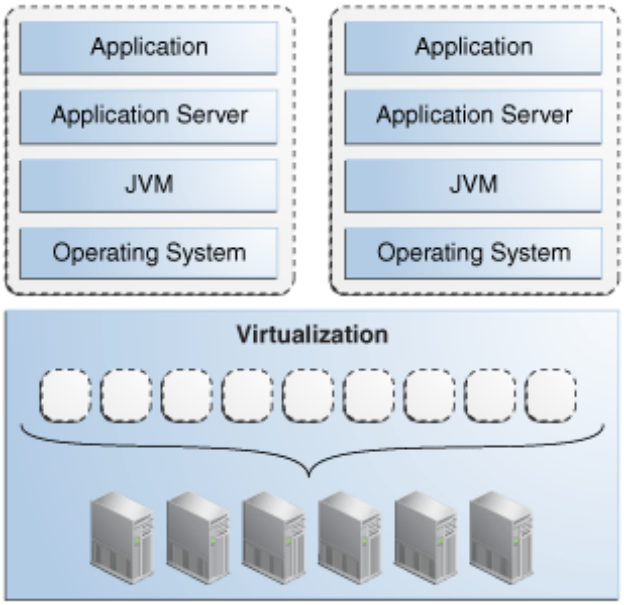
While virtualization has solved a multitude of problems, it is still difficult to deploy and manage complex applications made up of multiple tiers and components. Furthermore, virtualization is quickly becoming a commodity and the focus now shifts to directly virtualizing applications to reap the next level of benefits associated with virtualization.

1.1.1 What is Virtualization?

Virtualization is the process of abstracting hardware resources, such as CPU, memory, storage, and network interfaces, from the operating system and applications. The hardware runs virtualization software (for example, a hypervisor) that enables the installation of multiple operating systems, each capable of running simultaneously and independently, in its own secure physical environment.

The goal of virtualization is to make deployment of complete environments faster, easier, and more efficient. Virtualization's capabilities must be integrated to facilitate deployment and management of complete stacks. Virtualization must enable the entire stack to be easier to deploy, manage, and support.

Figure 1–1 Virtualization



1.1.2 Middleware Virtualization Challenges

The development and deployment of applications in your virtualized environment involves a sequence of operational stages including testing, staging, and production. The transition between these stages can be difficult as there are few facilities within existing virtualization infrastructure that guarantee consistency and correctness of the collection of software components. Implementing the physical to virtual (P2V) or virtual to virtual (V2V) transitions seems simple: create virtual images of the original deployments, then instantiate them in the target environment. Oracle VM can be used to implement such solutions.

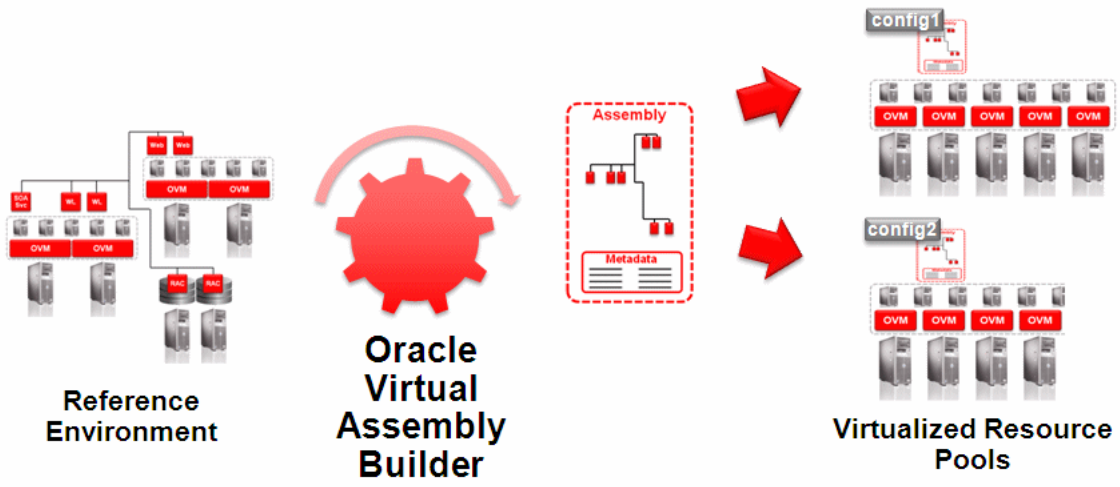
Handcrafting the virtualization solution has many pitfalls. Details of network connectivity may change in the deployment environment, but no automatic mechanism exists to perform or even to track these changes. Images may be specific to particular details of the deployment environment. The proliferation of images results in *sprawl*, creating maintenance overhead as each of the images must be patched at the operating system and application layers. These pitfalls create unanticipated costs.

1.1.3 What is Oracle Virtual Assembly Builder?

Oracle Virtual Assembly Builder is a tool for virtualizing installed Oracle components, modifying those components, and then deploying them into your own environment. Using Oracle Virtual Assembly Builder, you capture the configuration of existing software components in artifacts called software *appliances*. Appliances can then be grouped, and their relationships defined into artifacts called software *assemblies* which provide a blueprint describing a complete multi-tier application topology.

Oracle Virtual Assembly Builder allows the logical connections between appliances within an assembly to be reconfigured by a process known as *assembly editing*. When a desired assembly configuration has been achieved, you use Oracle Virtual Assembly Builder to prepare the assembly for deployment and then deploy it into your environment. The components and processes are described below.

Figure 1–2 Oracle Virtual Assembly Builder



1.1.4 Software Appliances

A software appliance (*appliance*) represents a single software component and its local execution environment.

1.1.5 Software Assemblies

A software assembly (*assembly*) is a collection of interrelated software appliances that are automatically configured to work together upon deployment. Assemblies are deployed onto a pool of hardware resources with minimal user input.

While assemblies are simply a collection of appliances with defined interconnects, assemblies must provide a set of capabilities in order to be useful in a production environment, including:

- Allow for the composition of appliances as well as external systems
- Externalize configuration in the form of metadata that can easily be customized
- Optionally define the start order of appliances to reflect interdependencies
- Provide a management domain which integrates into existing management infrastructure allowing for metadata definition, deployment, oversight and diagnostics

In addition to being comprised of appliances, assemblies can also contain references to *external systems*. This is necessary to represent infrastructure such as databases, servers or security providers that cannot or should not be included in an assembly.

To summarize, the notion of being able to create pre-built assemblies for deployment is extremely powerful and has a number of advantages that drive down operational costs and complexity. These include:

- Ability to easily replicate assemblies in production, even allowing for variations of the assembly without adding complexity
- Reduced risk of configuration errors as assemblies are moved between development, test and production environments

- Replicated environments facilitate high-level standardization and consistency across application infrastructures, allowing for simple implementation of best practices.
- Accelerated deployment of new infrastructures and applications

1.1.6 The Role of Oracle Virtual Assembly Builder

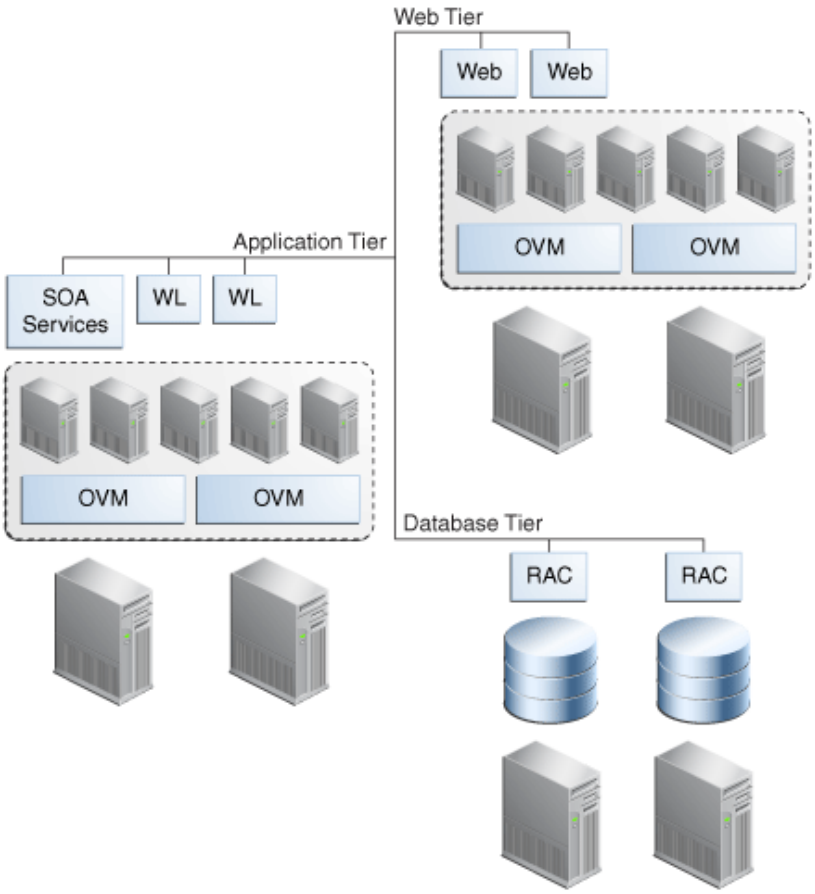
In order to realize these benefits, a simple means of composing assemblies of appliances is required. Specifically what is needed is tooling that allows for the composition of appliances as well as endpoint mapping of externalized systems and other larger non-virtual appliance-based systems such as databases and identity management servers.

Oracle Virtual Assembly Builder includes an intuitive visual environment, command line interface, and supporting infrastructure. Oracle Virtual Assembly Builder enables administrators to construct and deploy complete assemblies encompassing all of the components and systems that make up a potentially complex application structure or infrastructure.

Oracle Virtual Assembly Builder provides the following capabilities:

- Ability to browse a catalog of existing appliances and assemblies allowing for simple re-use of existing infrastructure
- Assembly editor that enables declarative composition of new assemblies based on existing appliances and external systems
- Ability to modify connections between appliances using drag-and-drop
- Property inspector that displays the editable properties of appliances and assemblies
- Ability to create templatized definitions of complete configurations, allowing for simple deployment
- Single-step deployment of virtualized multi-tier applications onto a pool of virtualized resources

Figure 1–3 Virtualized Multi-Tier Applications



1.1.7 Using Oracle Virtual Assembly Builder

Assembly creation and deployment is a straightforward, four-step process. First, in the *introspect* phase, the necessary metadata and configuration information is captured from an existing deployment for all components that make up the appliances within an assembly. During the *configure* phase, the relationships are established among the appliances and any external resources. The *prepare* phase creates the deployment artifacts necessary for the assembly that is relevant to the particular virtualization platform (that is, virtual images). Finally, the *deploy* phase deploys the assembly into your environment.

1.1.7.1 Introspect

In the introspect phase, you capture configuration metadata for individual software components, or collectively capture metadata for multiple distributed components. Target components may reside locally or remotely on multiple distributed systems that may be physical or virtual.

1.1.7.2 Configure

In the configure phase, you:

- Visually drag-and-drop components for creating complex assemblies using appliances maintained in a navigable catalog

- Establish relationships and connections between appliances using a wiring tool that automatically checks for protocol compatibility
- Create connections from appliances to external resources (such as database, security provider, messaging, and so on) not included within the assembly

1.1.7.3 Prepare

In the prepare phase, you:

- Create bootable virtual machine disk images with customized Oracle Enterprise Linux operating system distributions and configurable metadata allowing for deploy-time customization of the software component

1.1.7.4 Deploy

In the deploy phase, you:

- *Discover* targets available on virtualized environments by establishing authenticated connections directly with a virtual machine manager
- Create customized deployment configurations for assemblies that override base configuration properties for the appliances within the assembly
- Accommodate late-binding appliances automatically through deployment-specific customization
- Stage all appliance disk images and deploy entire assemblies onto targets in a single step
- Scale appliance instances after initial deployment of an assembly and automatically wire the newly deployed instances into the existing assembly

1.2 Understanding Oracle Virtual Assembly Builder

Oracle Virtual Assembly Builder captures the existing condition of a specific set of Oracle Fusion Middleware and Oracle Database software components from your environment, represents them as assemblies and appliances, and enables their deployment into your environment. Oracle Virtual Assembly Builder does not include the ability to administer the components and does not replace the administrative tools supplied with them.

Oracle Virtual Assembly Builder does not supply the virtual environment into which you deploy your Assemblies. You must establish the deployment environment using one of the target environments that Oracle Virtual Assembly Builder supports. For more information about supported deployment environments, see *Oracle Virtual Assembly Builder Installation Guide*.

1.2.1 Product Components

Oracle Virtual Assembly Builder consists of two major product components:

- Oracle Virtual Assembly Builder Studio provides you the capabilities to perform the first three phases of the assembly creation, the introspect phase, configure phase, and prepare phase. Oracle Virtual Assembly Builder Studio allows you to create and edit the assemblies, create assembly archives, and create templates and deployment plans which support deployment from Oracle Virtual Assembly Builder Deployer.
- Oracle Virtual Assembly Builder Deployer is a J2EE application that maintains a repository of assembly archives created by Oracle Virtual Assembly Builder

Studio. The Deployer provides operations for registering these assembly archives to virtualized systems such as Oracle VM and provides operations for orchestrating the deployment of the software system defined by the assembly archive.

The interface to Oracle Virtual Assembly Builder Deployer is a Web service which provides operations for uploading assembly archives, registering the assembly archive virtualization system and managing assembly instances for the system defined in the assembly archive. See *Oracle Virtual Assembly Builder Developer's Guide* for a description of the Web service.

1.2.2 Appliances and Assemblies

A minimal appliance consists of metadata (name and value pairs) describing the condition of the original component, together with a set of component-specific files that allow its configuration to be recreated at deployment time. As you use Oracle Virtual Assembly Builder to prepare an assembly for deployment into your environment, additional configuration information is created and stored along with the metadata.

The appliance metadata includes a description of each of the component's logical inputs and outputs. These inputs and outputs are collectively called *endpoints*. The HTTP input of an Oracle HTTP Server component is an example of an input endpoint. The `mod_wl_ohs` output of the same Oracle HTTP Server component is an example of an output endpoint.

The metadata describing endpoints includes protocols, port numbers, URLs, and so on. Oracle Virtual Assembly Builder captures enough information about each endpoint to allow the connection to be updated after the component is captured and before it is deployed. This capability allows Oracle Virtual Assembly Builder to ensure that the appliances will connect correctly within the deployment environment.

Appliances are grouped into *assemblies*. An assembly is a logical container for the appliances and the connections between them. You create assemblies using Oracle Virtual Assembly Builder and populate them with the appliances and the other assemblies (assemblies may contain other assemblies).

The process of capturing a software component from your environment as an Oracle Virtual Assembly Builder appliance begins with *introspection*.

1.2.3 Introspection

Introspection is an operation performed on a software component or a group of related components (to create an appliance or assembly). During introspection, Oracle Virtual Assembly Builder creates an xml description of the component and captures a component-specific set of configuration files. This information forms a snapshot of the component's configuration at the time of introspection. The introspection architecture is plug-in based and there is a plug-in for each supported component type. See *Appendix B, Oracle Virtual Assembly Builder Introspection Plug-ins* for more information about available plug-ins.

In most cases, the result of introspecting a component is an appliance. When you use Oracle Virtual Assembly Builder to introspect an Oracle WebLogic Server domain, however, the Introspector plug-in generates an assembly. The generated assembly contains an appliance representing the domain's Administration Server and other appliances representing each of the domain's Managed Servers.

Oracle Virtual Assembly Builder can introspect components on a local host or components located on remote, network-accessible hosts. Oracle Virtual Assembly

Builder uses the industry-standard SSH protocol to transport the introspection engine to the remote host and to return the introspection results.

Whether the introspection is local or remote, the results are stored locally in the catalog.

1.2.4 External Appliances

External appliances are virtual machine templates, created using non-Oracle Virtual Assembly Builder tools, that you import into your Oracle Virtual Assembly Builder installation. Once imported, the external appliance participates in all Oracle Virtual Assembly Builder operations, with certain limitations that result from the lack of introspection metadata. External appliances can be edited, added, and deployed as part of any assembly as you would any other appliance. Only Oracle virtual machines are eligible to be imported as external appliances at this time.

Use the abctl `importExternalTemplate` command to have an assembly template (created outside of Oracle Virtual Assembly Builder) captured in your catalog as an external appliance. For more information on using this command, see *Appendix A, Command Line Reference*.

1.2.5 Generic Appliances

Appliances constructed using the appliance type called "GenericProd". These type of appliances do not make use of product-specific logic to capture configuration or product location, instead a simple appliance is created and a set of user-supplied properties, paths, and scripts that make up the product are added to it in a generic manner. Also it does not make use of any product-specific logic to configure and start the product upon deployment, instead the set of scripts passed in at creation are executed at deployment to perform the necessary operations.

This allows user to create and deploy an opaque, standalone, and self-contained product or application as an appliance for which Oracle Virtual Assembly Builder does not have built-in support.

1.2.6 Catalog

Assemblies and appliances are represented on disk in an area called the *Catalog*. Assembly and appliance metadata is stored in nested directories within the metadata subfolder of the catalog root directory. Additional artifacts required for deployment are stored in other subdirectories defined by Oracle Virtual Assembly Builder. Since some of the on-disk artifacts may be very large, the catalog uses a sharing model for some artifacts of appliances and assemblies.

Only Oracle Virtual Assembly Builder-supplied tools should be used to operate on the catalog. Manually editing Oracle Virtual Assembly Builder metadata files is not supported.

1.2.7 External Resources

When defining an assembly, it may be necessary to make reference to servers that lie outside it. Your IT environment may, for example, include database, identity management, or other servers that are shared by many unrelated virtual deployments. It may be undesirable or impossible to include these systems within any specific assembly. For this reason, Oracle Virtual Assembly Builder enables you to define external components representing server resources that exist in your environment and will not be deployed as appliances. Representing them as external resources ensures

that referencing appliance(s) within the assembly are correctly configured at deployment time, making it unnecessary to manually correct their network configuration after they are deployed to the virtual environment.

1.2.8 Capturing File Sets

The introspection process captures the condition of a component and generates a metadata description of the actual component installation. Introspection does not capture the executables, shared libraries or other binaries of the component. Instead, introspection generates file set definitions that specify one or more file system hierarchies that must be captured to reproduce the same component installation in the deployment environment. By default, after the introspection is complete, Oracle Virtual Assembly Builder automatically captures a copy of the actual installation described by the metadata. This step is known as *capturing file sets*.

By default, introspection and capturing file sets are done together whether you use Oracle Virtual Assembly Builder Studio or Oracle Virtual Assembly Builder command line interface. Optionally, you can choose to do these steps separately.

1.2.8.1 Assembly Archive

The assembly archives created by Oracle Virtual Assembly Builder Studio contain information about a software system composed of multiple, related software stacks which work together to form an application. This system is referred to as an assembly. The assembly archive contains metadata about the assembly and assembly templates that are used to instantiate an instance of the assembly in a virtualized environment.

1.2.8.2 File Sets and Shared File Sets

You can configure a file set as shared or local. If supported by the underlying infrastructure platform, you can specify that file sets be shared with individual appliances within assemblies.

1.2.8.3 Networks

The assembly archive defines a set of logical networks for the application it represents. For each appliance, the assembly archive also defines one or more network interfaces. Each network interface is associated with one of the archive's networks, allowing the assembly archive to fully represent the network connectivity requirements of the assembly.

The deployment plan specifies a network in the virtualization environment to be used for each logical network declared in the assembly archive. The model supports the configuration and binding to both public and private networks (where private is defined as existing between two appliances in an assembly and not surfaced as part of the public network for access to the deployed application topology.)

The Deployer creates and attaches one or more Vnets to the virtual machines it creates using the underlying interfaces of the virtualization system. Note that these Vnets are hypervisor-level Vnets, as opposed to virtual machine-level Vnets. If the virtualization system supports it, the Deployer may also dynamically create private networks to associate these Vnets with.

1.2.9 Assembly Templates

An assembly template is a set of virtual disk images that can be used to create and start new virtual machine instances. A template is created for each appliance in an assembly, consisting of a guest operating system, the appliance's file sets and

metadata, and supporting Oracle Virtual Assembly Builder infrastructure. Templates are made available to the virtualized environment by registering them to that environment, at which point virtual machine instances can be created based on the templates.

Oracle Virtual Assembly Builder supports Oracle Enterprise Linux as the virtual machine guest operating system.

1.2.10 Deployment Plans

Deployment plans are used to customize assemblies prior to deployment. You can create a deployment plan in which you customize default assembly and appliance properties, and provide deployment-specific information such as network configuration. In some cases you may be required to customize certain properties in order to proceed with a successful deployment (for example, static IP address, or password properties).

1.2.11 Understanding Deployer Concepts

This section describes Deployer concepts.

1.2.11.1 Targets

Different virtualization systems organize their resources in different ways and require different information for referencing and accessing them. In order to provide a common user experience across different systems, Oracle Virtual Assembly Builder Deployer defines the notion of a *target*. Targets are configured using administration interfaces defined later in this document and are used to reference a resource or pool of resources in the virtualized system. The configuration information provided for each target is specific to the virtualization system containing the target.

Oracle Virtual Assembly Builder Deployer supports Oracle VM and Oracle Exalogic. Oracle Exalogic comes pre-configured with a single target.

1.2.11.2 Assembly Instances

An assembly instance is a deployable instance of an assembly archive for a specific target virtual environment.

1.2.11.3 Appliance Instances

An appliance instance is an instance of an appliance running and/or created in the target virtual environment.

After you deploy an assembly instance, the target number of appliance instances for each appliance is started. The initial target for each appliance is specified in the deployment plan. You can dynamically specify a new target after an assembly instance has been deployed.

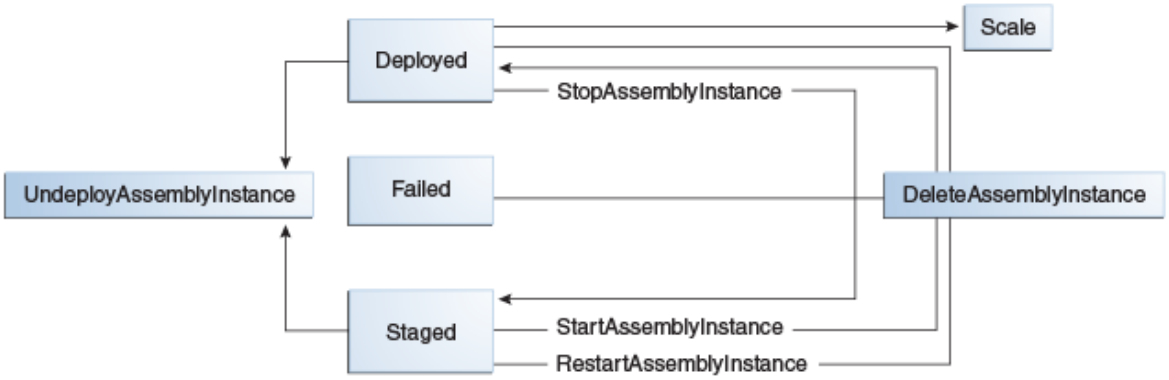
1.2.12 Deployment Life Cycle

An assembly instance is a deployable artifact. You would need to create an assembly instance by selecting an assembly, one of the its deployment plans and the target to which it must be deployed to. CreateAssemblyInstance can be used to create the assembly instance.

At deployment time, you choose the assembly instance to be deployed.

Deployment of an assembly instance will transition through various phases (Figure 1–4). The phases include: *Staged*, *Deployed*, and *Failed*. Each state allows a subset of operations. For example, when an assembly instance is deployed, you may start and stop the appliance instances, or you may increase or decrease the number of appliance instances associated with that deployed assembly instance. Oracle Virtual Assembly Builder does not monitor the health of the deployed application; it will only inform you of whether or not an assembly is deployed or staged, as well as the success or failure of a deployment-related operation.

Figure 1–4 Deployment Life Cycle



Here is a summary of the assembly instance phases:

- *Deployed* When the assembly instance is deployed and the operation has successfully completed, it reaches the deployed state. The operations that can be performed on a *Deployed* assembly instance are:
  - *StopAssemblyInstance* This operation will shut down all the running appliance instances for the assembly instance. The assembly instance is transitioned to the *Staged* phase after this operation is completed. It leaves the appliance instances in the virtualized environment so that they can be restarted later.
  - *UndeployAssemblyInstance* This operation will stop all the running appliance instances and remove them from the environment. After this operation is completed, the assembly instance will kept in the system so that it can be deployed again.
  - *RestartAssemblyInstance* This operation will restart all the running appliance instances of the assembly instance. The assembly instance will transition to the *Staged* and then transition back to *Deployed*.
  - *RedeployAssemblyInstance* This operation will redeploy the assembly instance. As part of this operation all appliance instances will be stopped and removed from the target environment. New appliance instances will be created and started.
  - *Scale* Scales the scaling group within an assembly instance. Scaling can be performed to scale up or down a scaling group with the assembly instance. The number of appliance instances that can be running for a scaling group must lie between its configured minimum and maximum instance limits. The Deployment continues to remain in the *Deployed* state.
- *Failed* When there is a failure in a deploy or undeploy operation, the assembly instances reaches this phase. A deployment operation may fail for a variety of

reasons, such as insufficient resources. The operations that can be performed on a failed deployment are:

  - *DeleteAssemblyInstance* This operation will do the necessary cleanup (such as stopping and removing the appliance instances). After this operation is completed, the assembly instance no longer exists.

- *Staged* The staged phase is reached by stopping an assembly instance. In this phase all the appliance instances have been shut down. The operations that can be performed from this phase are:

  - *StartAssemblyInstance* This operation will start up all the appliance instances that have been shut down. After this operation is completed, the assembly instance is returned to the *Deployed* state.

  - *UndeployAssemblyInstance* This operation will remove all the appliance instances that have been shut down from the virtualized environment. After this operation is completed, the assembly instance will be kept around so that it can be deployed again.

2 Architecture

This chapter describes the architecture of Oracle Virtual Assembly Builder, and contains the following sections:

- Section 2.1, "Major Components"
- Section 2.2, "Setup Scenarios"
- Section 2.3, "Deployment Platforms"

2.1 Major Components

This section describes the major components of Oracle Virtual Assembly Builder.

2.1.1 Oracle Virtual Assembly Builder Studio

Oracle Virtual Assembly Builder Studio is the component that allows users to create assemblies, and has two interfaces, the Oracle Virtual Assembly Builder Studio GUI and the `abctl` command-line interface.

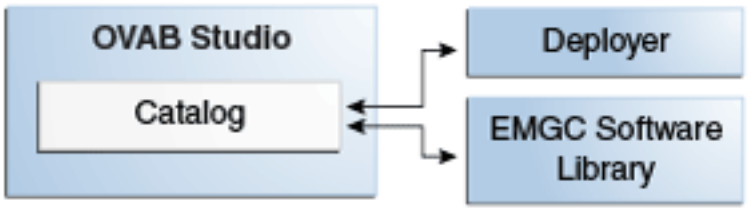
Oracle Virtual Assembly Builder Studio and `abctl` interact with Oracle Virtual Assembly Builder Deployer as a Web client as shown in Figure 2–2. Oracle Virtual Assembly Builder Studio provides a Deployer user interface whereby deployments can be initiated from Oracle Virtual Assembly Builder Studio and leverage the Oracle Virtual Assembly Builder Deployer.

2.1.1.1 High-Level Catalog Overview

The Oracle Virtual Assembly Builder Studio catalog contains the metadata definitions of appliances and assemblies that are the result of introspection. The captured file sets, virtual machine templates and deployment plans for those appliances and assemblies are also stored in the catalog.

Figure 2–1 shows Oracle Virtual Assembly Builder Studio and its relationship to both its catalog as well as to the Deployer and Enterprise Grid Control Software Library.

Figure 2–1 Oracle Virtual Assembly Builder Studio and Deployer



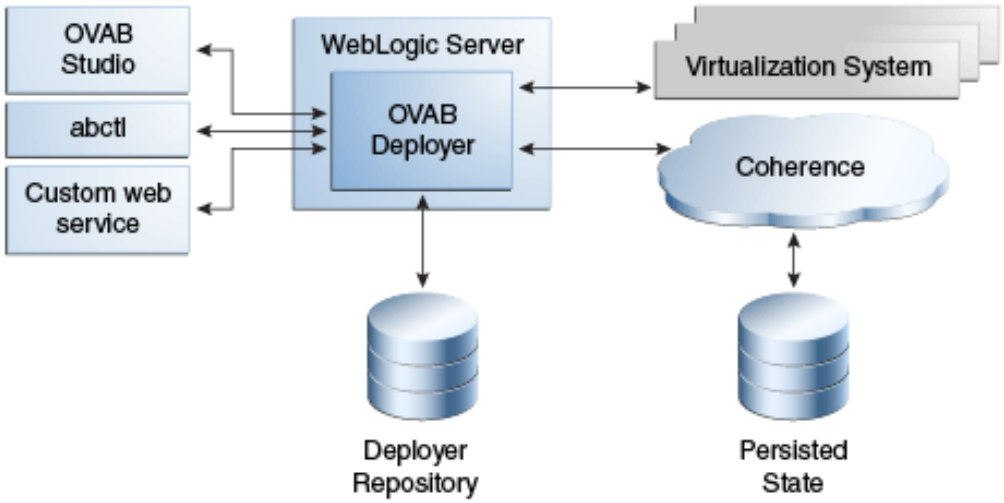
2.1.2 Deployer

The Oracle Virtual Assembly Builder Deployer is a J2EE application that maintains *assembly archives* created by Oracle Virtual Assembly Builder Studio, provides operations for registering assembly archives to virtualized systems such as Oracle VM and provides operations for orchestrating the deployment of the software system defined by the assembly archive.

The assembly archives created by Oracle Virtual Assembly Builder Studio contain information about a software system comprised of multiple, related software stacks which work together to form an application. This system is referred to as an assembly. The assembly archive contains metadata about the assembly and virtual machine templates that are used to instantiate an instance of the assembly in a virtualized environment.

The Deployer runs in an Oracle WebLogic Server servlet container on the Oracle WebLogic Server Administration Server. Figure 2–2 shows the top-level components of the Oracle Virtual Assembly Builder Deployer.

Figure 2–2 Deployer Architecture



2.1.2.1 Web Application

The interface to the Oracle Virtual Assembly Builder Deployer is a Web service which provides operations for uploading assembly archives, registering assembly archives to the virtualization system and managing deployment instances for the system defined in the assembly archive.

Operations against the Web service are made by posting an HTTP request to the Oracle Virtual Assembly Builder Deployer's context path. The request includes a request parameter that defines the action followed by zero or more request parameters that define arguments for the operation. The response is usually an XML document related to the operation performed, except in cases where artifacts such as assembly archives, deployment plans or metadata files are being downloaded from the Deployer to the client. See *Oracle Virtual Assembly Builder Developer's Guide* for sample requests and responses.

Some operations may define an asynchronous action.

2.1.2.2 Deployments navigator

The *Deployments* navigator of the Deployer is a feature of Oracle Virtual Assembly Builder Studio which is a specific Web client. It contains the assembly archives and any deployment plans uploaded to the Deployer through the Web client. You create assembly archives and deployment plans using Oracle Virtual Assembly Builder Studio, however, the repository maintained by the Deployer is separate from the Oracle Virtual Assembly Builder Studio catalog. The runtime state for the Deployer is persisted on disk so that the Deployer instance can be recovered in the case of a failure (the Deployer process exits).

For Oracle VM, the Deployer interacts with one or more virtualization systems and orchestrates the deployment of assembly archives into these systems. For Oracle Exalogic, the Deployer has one preconfigured target and orchestrates the deployment of assembly archives into that system.

2.1.2.2.1 Assembly Archive Versioning The Deployer repository allows you to keep multiple versions of an assembly archive in the repository at the same time. When you upload an assembly archive, the Deployer assigns it a version number. If another assembly archive of the same name is uploaded, then it will get the next version.

If you do not specify the version number in an operation (such as `registerAssemblyArchive`, or `describeAssemblyArchives`) the latest version will be used.

2.2 Setup Scenarios

This section describes the different ways that you can set up Oracle Virtual Assembly Builder.

2.2.1 Deployer-only Installation Scenario

You can set up Oracle Virtual Assembly Builder Deployer by itself, without Oracle Virtual Assembly Builder Studio. This installation scenario provides you the ability to configure deployment targets, upload assembly archives to Deployer, create assembly instances, deploy/undeploy/start/stop assembly instances and scale appliance instances.

This configuration contains the following interfaces:

- You can use the Web service API to perform PaaS operations.
- You can use abctl to interact with the Web service API for either PaaS or Administrative operations.
- You can use the Web service API to perform administrative operations on the deployer.

2.2.2 Studio-only Installation Scenario

You can set up Oracle Virtual Assembly Builder Studio by itself, without Oracle Virtual Assembly Builder Deployer. This installation scenario provides you the ability to create appliances and assemblies, create appliance templates and assembly archives and create deployment plans.

This configuration contains the following interfaces:

- The Oracle Virtual Assembly Builder Studio graphical user interface, and
- `abctl` CLI (all functionality for creating and deploying assemblies)

Note: This scenario does not apply to Oracle Exalogic, without additional installation.

2.2.3 Deployer Co-located with Studio

You can set up Oracle Virtual Assembly Builder Deployer co-located with Oracle Virtual Assembly Builder Studio on the same machine, but in a different process. This configuration contains the previously described interfaces for Oracle Virtual Assembly Builder Deployer and Oracle Virtual Assembly Builder Studio, and provides you the ability to perform all the operations described in the "Deployer-only Installation Scenario", plus those operations in the "Studio-only Installation Scenario".

2.2.4 Remote Deployer with Studio

You can set up Oracle Virtual Assembly Builder Studio to interact remotely with Oracle Virtual Assembly Builder Deployer. Oracle Virtual Assembly Builder Studio can interact remotely with both the Oracle VM Deployer and the Deployer running inside Oracle Exalogic. You can use both the graphic user interface and the `abctl` command-line interface of Oracle Virtual Assembly Builder Studio to interface with either Deployer.

This configuration contains the previously described interfaces for Oracle Virtual Assembly Builder Deployer and Oracle Virtual Assembly Builder Studio, and provides you the ability to perform all the operations described in the "Deployer-only Installation Scenario", plus those operations in the "Studio-only Installation Scenario".

2.3 Deployment Platforms

You can deploy Oracle Virtual Assembly Builder assembly archives to the Oracle VM or Oracle Exalogic platforms.

2.3.1 Deploy to Oracle VM x86

You can deploy to Oracle VM 3.0.3 or higher.

2.3.2 Deploy to Oracle Exalogic

Oracle Exalogic is a target platform for Oracle Virtual Assembly Builder. Oracle Virtual Assembly Builder Deployer is configured and provisioned to be running as part of the Oracle Exalogic runtime.

3 Security

This chapter describes the security of Oracle Virtual Assembly Builder, and contains the following sections:

- Section 3.1, "Resources"
- Section 3.2, "Security Model Employed by Deployer"
- Section 3.3, "Roles and Groups"

3.1 Resources

The resources describe in Table 3–1 are protected:

Table 3–1 Resources

Resource	How protected
Target	For Oracle VM, a target is created by the Cloud Admin. For Oracle Exalogic, a single target is implicitly configured at install time and no new targets may be created after installation. For Oracle VM, the Cloud Admin grants permission to Application Admins to use a target. For Oracle Exalogic, any Cloud Admin may use a configured target, however, they must supply their own credential information to the virtualization system. The reason for this difference is that in Oracle VM, the Cloud Administrator provides shared credentials for Oracle VM Manager. For Oracle Exalogic, the single implicitly configured target only includes the target URL. Only a Cloud Admin may view the configuration information of a target.
Credentials (passwords and keys)	Encrypted
Assembly archive in the Deployer	Protected by the "owner" concept, see Section 3.2.2, "Assembly Archive Authorization".
Assembly instances and deployment plans in the Deployer	Protected by the "owner" concept, see Section 3.2.2, "Assembly Archive Authorization".

3.2 Security Model Employed by Deployer

Oracle Virtual Assembly Builder Deployer is constructed as an application running in the Oracle WebLogic Server container and leverages the security infrastructure provided by Oracle WebLogic Server. Oracle WebLogic Server is configured with the

embedded LDAP authenticator by default but you can reconfigure Oracle WebLogic Server to point to an external corporate LDAP. The Deployer depends upon Oracle WebLogic Server users being put into one or both of two groups: Cloud Admins or Application Admins.

For information on creating the Cloud Admin and Application Admin groups and roles, see Section 3.3, "Roles and Groups".

Having a user in those groups allows them to be mapped to having the roles needed by the Deployer: Cloud Admin or Application Admin. The Deployer's access control requires that a user be in one or both of those groups. When you set up a connection to the Deployer using `abctl` you must specify a username and password for an Oracle Weblogic Server user that has been added to one or both of those groups.

When a user attempts a Deployer operation using either Oracle Virtual Assembly Builder Studio or `abctl`, they are authenticated using the connection information and then once authenticated their request goes to the Deployer's Servlet running in the WLS Servlet Container. The servlet checks that they are in one of those roles and then performs additional checks (for example, whether the user is allowed to access the specified target, or whether the user is allowed to access the specified assembly archive).

Oracle Virtual Assembly Builder Deployer uses Oracle WebLogic Server capabilities for access checking, however, Oracle Virtual Assembly Builder Deployer does not have visibility on the LDAP information for managing those identities. Due to this circumstance, it is possible for you to delete a user out of Oracle WebLogic Server while Oracle Virtual Assembly Builder configuration referencing that username is still in place.

Caution: Once an Oracle Virtual Assembly Builder request has moved beyond the authentication and access checking described above, it will continue to completion even if the user information is removed from the WebLogic Server authentication store.

3.2.1 Target Authorization

For Oracle VM, only the Cloud Admin can perform functions such as creating a target, or viewing the configuration information of a target. For Oracle VM, the Cloud Admin grants permission to Application Admins to use a target.

For Oracle Exalogic, any Cloud Admin may use a configured target, however, they must supply their own credential information to the virtualization system. In Oracle VM, the Cloud Administrator provides shared credentials for Oracle VM Manager.

Caution: If a user is removed from the Oracle WebLogic Server authentication store, you must remove any cached information about that user from the Deployer by removing that user from any targets (`describeUserTargets` and `removeTargetUsers`) to which they were previously added.

Removing them from the targets in the Deployer removes any cached information for that user from the Deployer. This prevents a situation where a user has cached information, the user is removed from Oracle WebLogic Server, a different user of the same name is added into Oracle WebLogic Server and the new user inherits all the previous user's cached information in the Deployer.

3.2.2 Assembly Archive Authorization

When you attempt to access an assembly archive, the Deployer performs a check. The following users are granted access:

- A user with the OVAB_ADMIN role.
- The owner of the assembly archive. The first user to upload the assembly archive becomes the owner.

3.2.2.1 Authoring Additional Users with the addAssemblyUsers Command

You can authorize additional users, adding them to the access list, using the `addAssemblyUsers` command. Only a user with the OVAB_ADMIN role or the assembly archive owner can perform this operation.

Only the Application Admin who is the owner of the assembly archive can manage it by adding/removing/describing assembly users, deleting the assembly or updating the assembly archive. The users in the assembly access list can use the assembly by viewing, downloading, registering and creating assembly instances (assuming they also have access to the target).

3.2.2.2 Assembly Instances

The user who creates an assembly instance is its owner. Besides the Cloud Admins, only the assembly instance owner can manage and use an assembly instance.

3.2.2.3 Assembly Resources

The `uploadAssemblyResources` command is controlled by a security policy. A resources file may or may not contain scripts. If the resource file does not contain scripts, a user on the assembly access list can run the command. If the resource file does contain scripts, only the Cloud Admin user is allowed to run the command, to prevent a malicious attack.

When including scripts in the resources files, the lifecycle names that are supported are: `pre-deploy`, `post-deploy`, `deployer-pre-app-config`, `deployer-post-app-config`, `deployer-pre-vm-start`, `deployer-post-vm-start`, `deployer-pre-vm-stop`, `deployer-post-vm-stop`, `pre-undeploy`, `post-undeploy`. You can create corresponding script folder names.

3.2.3 Enabling the Deployer's Authentication and Authorization Model

Perform these steps to enable the Deployer's authentication and authorization model:

1. A system administrator defines users in LDAP and assigns a `Cloud Admin` or `Application Admin` role to those users. These roles control what things a given user can do. This step is done through the Oracle WebLogic Server administrative console.

   You perform the remainder of the operations against the Web service; the Web service calls require Oracle Virtual Assembly Builder user credentials. The Deployer Web service operations that need to be called are as follows:

2. `createTarget` - This operation, which can only be performed by the Cloud Admin, defines the connection information, and, depending on the backend type, user credentials for the backend. For Oracle VM, credentials are supplied here, but for Oracle Exalogic, individual users provide their own credentials in the `AddTargetUser` step.

3. `addTargetUser` - Depending on the backend type, this may be a Cloud Admin call or Application Admin call, due to differences in the security models of the backend systems.

   - For Oracle VM targets, this operation is only performed by the Cloud Admin and is used to control what users can access the pool. This is a Cloud Admin operation because the credentials supplied by the Cloud Admin must be protected from general users.
   - For Oracle Exalogic targets, this is a user operation and is used to specify user credentials (in this case, the Cloud Admin does not have to give specific access because the backend will be checking the credentials). This is an Application Admin call because the user's credentials must be protected from others, including the Cloud Admin.

4. `uploadAssemblyArchive` - This is a Cloud Admin call used to upload an assembly archive to Oracle Virtual Assembly Builder Deployer (note that non-admin users are not allowed to upload assembly archives).

For procedures for configuring Security refer to Section 5.4.1, "Configuring Targets".

3.3 Roles and Groups

Oracle Virtual Assembly Builder defines security roles and groups. The product installer sets up the roles and groups for the embedded LDAP case and you create the users and add them to the groups through the Oracle WebLogic Server console.

Follow this process to create roles and groups:

1. Use the procedures in *Oracle® Fusion Middleware Securing Oracle WebLogic Server* to configure Oracle WebLogic Server for external LDAP.
2. Create groups for "Cloud Admins" and "Application Admins" in the LDAP server.
3. Add the users defined in the LDAP server to these groups.
4. Place the groups into the security roles using the role expression `Grp(GroupName|GroupName|GroupName)`.

   The process of computing and granting roles in Oracle WebLogic Server is referred to as role mapping. An access decision is the component of an Authorization provider that determines whether a subject has permission to perform a given operation on a WebLogic resource. (See "Access Decisions" in Developing Security

Providers for Oracle WebLogic Server, in *Oracle® Fusion Middleware Securing Oracle WebLogic Server*).

3.3.1 Application Admin Group

You create the `Application Admins` group out-of-band in LDAP.

You create `Application Admin` users out-of-band in LDAP. You include members of this group into the `Application Admin` group.

3.3.2 Cloud Admins Group

You create the `Cloud Admins` group out-of-band in LDAP.

You create `Cloud Admin` users out-of-band in LDAP. You include the users in the `Cloud Admins` group, which results in the `Cloud Admin` role being assigned at login.

3.3.3 Cloud Admin Role

You create the `Cloud Admin` role by an auth-constraint in the web.xml and a security-role-assignment of the `Cloud Admins` group to the `Cloud Admin` role in weblogic.xml.

3.3.4 Application Admin Role

You create the `Application Admin` role by an auth-constraint in the web.xml and a security-role-assignment of the `Application Admins` group to the `Application Admin` role in weblogic.xml.

4

Appliance and Assembly Structure

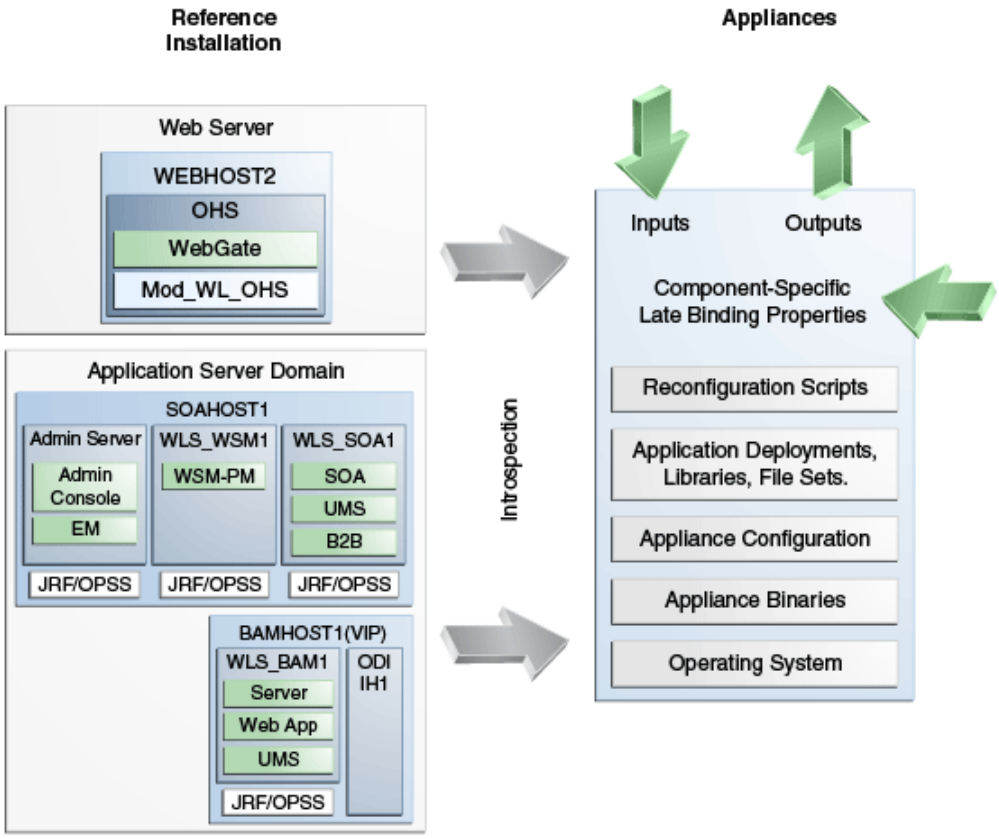
This chapter describes the structure of appliances and assemblies in Oracle Virtual Assembly Builder, and contains the following sections:

- Section 4.1, "Appliance Structure"
- Section 4.3, "New Concepts in Deployments to Oracle VM 3.0"

4.1 Appliance Structure

An appliance is a set of self-contained virtual disk images with all the software to run a single instance of a single component. An appliance includes appliance configuration, appliance binaries, and an operating system (Figure 4–1).

Figure 4–1 Appliance Structure



4.1.1 Appliance Configuration

Appliance configuration is captured during introspection of the appliance, and includes introspection properties, system and user properties, file sets, and optionally, customized scripts and properties files.

4.1.2 Appliance Binaries

Appliance binaries are software binaries included in the appliance. These include both the container or product binaries as well as the configured customer application binaries.

4.1.3 Operating System

A customer provides the operating system base image for the appliance, based on an Oracle Enterprise Linux template, and customized for his particular requirements. Oracle Virtual Assembly Builder publishes several sample base images.

4.2 Assembly Structures

Appliances can be grouped, and their relationships defined into artifacts called software *assemblies* which provide a blueprint describing a complete multi-tier application topology.

Assemblies include appliances, as well as references to external systems which represent infrastructure such as databases, servers or security providers that cannot or should not be included in an assembly.

4.3 New Concepts in Deployments to Oracle VM 3.0

This section describes structures introduced for Oracle VM 3.0 deployments:

- Section 4.3.2, "Configuring your Network Interface (NIC)"
- Section 4.3.3, "Shared File Sets"
- Section 4.3.4, "Zero-count Appliances"
- Section 4.3.5, "Anti-Affinity"

4.3.1 Terminology

The following alphabetical list of terms are defined:

- anti-affinity: placing multiple instances of a particular appliance across different physical hosts in a target, with even distribution of instances across physical machines.
- NIC: network interface. A NIC may only be connected to one Vnet.
- shared file sets: file sets shared with individual appliances within assemblies.
- Vnet: virtual network. One or more NICs or vNICs connect to the Vnet.
- vNIC: virtual network interface. A vNIC may only be connected to one Vnet.
- zero-count appliance: an appliance initially deployed with zero appliance instances.

4.3.2 Configuring your Network Interface (NIC)

Creating a new assembly automatically creates one Vnet for that assembly, which can consist of one or more network interfaces or virtual network interfaces connecting to that Vnet. That Vnet becomes the default virtual network. You can create additional Vnets, or delete ones that you have created. You can bind network interfaces to a Vnet during assembly editing.

4.3.3 Shared File Sets

If supported by the underlying infrastructure platform, you can specify that file sets be shared with individual appliances within assemblies.

4.3.4 Zero-count Appliances

You can initially deploy an appliance with zero appliance instances. However, in subsequent scaling operations you could add appliance instances to those appliances that are part of the assembly configuration but were initially "deployed" with a zero-instance count. Only appliances that no other appliance in the assembly references can have a zero-instance count. This means no other appliance depends on it or connects to any of its inputs.

4.3.5 Anti-Affinity

You can specify the requirement to place multiple instances of a particular appliance across different physical hosts in a target.

You can specify the minimum number of physical machines across which the instances of an appliance will be distributed. The Deployer ensures that the instances are distributed evenly.

For example, if the minimum number of physical machines is specified to be two, and there are four instances, then no physical machine would have more than two instances. Also, if there are not enough physical machines to satisfy the minimum requirement then deployment or scaling operations requesting a new instance for the appliance will be rejected.

5

Using Oracle Virtual Assembly Builder

This chapter describes how to use Oracle Virtual Assembly Builder, and includes the following sections:

- Section 5.1, "Oracle Virtual Assembly Builder Interfaces"
- Section 5.2, "Typical Workflow"
- Section 5.3, "Operations Related to Creating an Assembly"
- Section 5.4, "Operations Related to Deployment"

5.1 Oracle Virtual Assembly Builder Interfaces

Oracle Virtual Assembly Builder provides the following user interfaces depending on which parts of the product you have installed:

- When you have installed Oracle Virtual Assembly Builder Studio the following interfaces are supported:
  - Oracle Virtual Assembly Builder Studio graphical user interface
  - `abctl`, a command-line tool.
  - Oracle Virtual Assembly Builder Studio and `abctl` interact with Oracle Virtual Assembly Builder Deployer as a Web client. In a suitably configured Oracle Virtual Assembly Builder Studio environment, you can perform deployment through the configured connection to Oracle Virtual Assembly Builder Deployer using either the graphical user interface or abctl.
- When you have installed Oracle Virtual Assembly Builder Deployer the following interfaces are supported:
  - `abctl`, a command-line tool.
  - A Web service API to interface with Oracle Virtual Assembly Builder Deployer Web service, as described in *Oracle Virtual Assembly Developer's Guide*.

  When only Oracle Virtual Assembly Builder Deployer is installed, you can access only a subset of the commands. When Oracle Virtual Assembly Builder Studio is installed, or Oracle Virtual Assembly Builder Studio and Oracle Virtual Assembly Builder Deployer are installed, you can access all the commands. See Appendix A, "Command Line Reference".

> **Note:** You cannot launch two sessions of either the Oracle Virtual Assembly Builder Studio or `abctl` interfaces at the same time.

5.1.1 Accessing Oracle Virtual Assembly Builder Studio

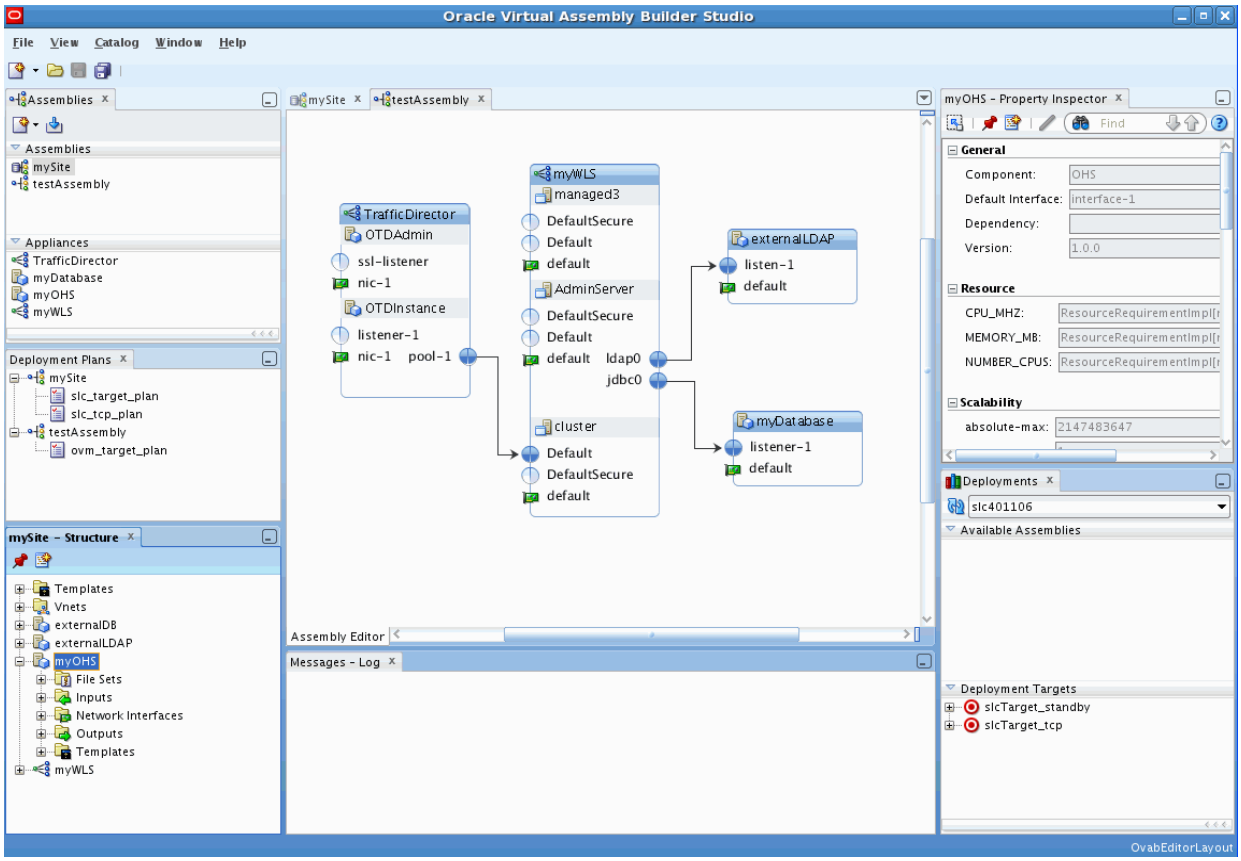
Launch Oracle Virtual Assembly Builder Studio by executing the command:

```
$AB_INSTANCE/bin/abstudio.sh
```

> **Note:** For information on installation, see *Oracle Virtual Assembly Builder Installation Guide*.

Figure 5–1 shows Oracle Virtual Assembly Builder Studio.

Figure 5–1 Oracle Virtual Assembly Builder Studio



5.1.2 Accessing the abctl Command-Line Tool

Launch the `abctl` command-line tool by executing the command:

```
$AB_INSTANCE/bin/abctl
```

5.1.3 Accessing Logs

The log file is stored at `$AB_INSTANCE/logs/assemblybuilder.log`.

Access the log file manually, or view its messages through Oracle Virtual Assembly Builder Studio in the *Messages* window.

5.1.4 Differences Between the Interfaces

The Oracle Virtual Assembly Builder Studio and `abctl` interfaces complement each other but do not include identical functionality. Here are the main differences:

- When running in Deployer-only mode only the `abctl` interface is supported.
- Only Oracle Virtual Assembly Builder Studio provides editing capability. That is, the following operations are not supported in `abctl`:
  - managing file set definitions: creating, updating and deleting file set definitions
  - managing target connections: creating, deleting or editing existing target connections
  - editing property values
  - creating/editing a deployment plan
- In Oracle Virtual Assembly Builder Studio, you can introspect multiple reference systems and put the results into a new or existing assembly. In `abctl`, you must introspect reference systems one-by-one and subsequently add them to an assembly.
- Only `abctl` provides the ability to create a 'target' connection to an Oracle VM 3.0 environment. Oracle Virtual Assembly Builder Studio does not have a connection wizard.

  Also, you can only make a Deployer connection using `abctl`.
- External virtual machine templates can only be imported into the catalog as external appliances using `abctl`.

These differences will be further detailed in Section 5.3, "Operations Related to Creating an Assembly".

5.1.5 Naming Rules

Any user-provided names must follow these rules:

- The name must begin with an alphabetic character.
- The name may only contain alphanumeric characters, or the underscore (_) or hyphen (-) characters.
- The name must be 4 to 40 characters long.

5.1.5.1 Naming Conflicts

You may experience a name conflict between appliances or assemblies in a catalog if you import an appliance or assembly into a catalog where you already have an appliance or assembly with the same name. If you want to overwrite the existing appliance or assembly you can use the `force` option.

5.1.6 Symbolic Links

Symlinks are not supported by Oracle Virtual Assembly Builder, and can lead to errors during introspection, capturing file sets, and deployment. Avoid symlinks in your Linux reference systems.

5.2 Typical Workflow

Users will typically use Oracle Virtual Assembly Builder in these ways:

- Create assemblies and appliances:
  - introspect a reference system to capture the necessary metadata and configuration information for all components that make up the appliances within an assembly.
- Edit assemblies and appliances to configure the relationships among the appliances and any external resources.
  - create networks within an assembly
  - create network interfaces within an appliance
  - bind appliance inputs to network interfaces and bind network interfaces to networks
  - create external resources from an appliance output
- Prepare deployment artifacts necessary for the assembly:
  - Create bootable virtual machine disk images with customized Oracle Enterprise Linux operating system distributions and configurable metadata allowing for deploy-time customization of the software component
- Deploy: deploy the assembly into your environment.
  - configure security: define the connection to Oracle VM or Oracle Exalogic backend endpoints and add deployment targets in the backend.
  - create and edit a deployment plan
  - register an assembly archive to a target
  - deploy assembly instances
  - perform other lifecycle operations on assembly instances

5.3 Operations Related to Creating an Assembly

This section details how you will use Oracle Virtual Assembly Builder Studio or `abctl` command line utility to perform operations related to creating an assembly.

- Section 5.3.1, "Introspect a Reference System"
- Section 5.3.2, "Capture File Sets for an Appliance or an Assembly"
- Section 5.3.3, "Create Templates for an Appliance or an Assembly"
- Section 5.3.4, "Edit an Assembly Using Oracle Virtual Assembly Builder Studio"

5.3.1 Introspect a Reference System

The introspection operation results in appliance(s) and/or an assembly (if you performed the operation using Oracle Virtual Assembly Builder Studio and created an assembly) being created in the catalog.

During introspection, the metadata for appliances and assemblies is created in the `$AB_INSTANCE/catalog/metadata` directory. A unique ID (called the capture ID or *cid*) is generated for each appliance or assembly, and is stored in its metadata. In addition, a file set definition is created in the shared area of the catalog.

Note: You should not change any configuration or content of the reference system between introspection and capturing file sets, as that may create undesired results. For instance, introspecting a reference system on one date and capturing file sets in the "same" reference system at some arbitrary future date is not supported.

For introspection to succeed, some introspection plug-ins have specific requirements for the reference system's running state. Table 5–1 lists the preconditions for the products supported by Oracle Virtual Assembly Builder.

Table 5–1 Introspection Plug-in Requirements

Introspected Product	Running State Pre-Condition
Oracle WebLogic Server	Administration Server must be up and in the running state (not in the admin state). Managed Server(s) may be up or down.
Oracle Coherence*Web	Administration Server must be up and in the running state (not in the admin state). Managed Server(s) may be up or down.
Oracle Forms*Web	Administration Server must be up and in the running state (not in the admin state). Managed Server(s) may be up or down.
Oracle SOA for WebLogic Server	Administration Server must be up and in the running state (not in the admin state). Managed Server(s) may be up or down.
Oracle HTTP Server (OHS)	No requirement; Oracle HTTP Server may be up or down.
Oracle Web Cache	No requirement; Oracle Web Cache may be up or down.
Oracle RDBMS (DB)	In the introspection phase, the database can be up or down.
Oracle RACDB	In the introspection phase, the database can be up or down.
Oracle Reports	Administration Server must be up and in the running state (not in the admin state). Managed Server(s) may be up or down.
Oracle Traffic Director	In the introspection phase, the Oracle Traffic Director application can be up or down.
Oracle Tuxedo	In the introspection phase, the Tuxedo application can be up or down.

5.3.1.1 No Support for Mounted NFS File Systems

Oracle Virtual Assembly does not support introspection or deployment of reference systems with mounted NFS file systems.

5.3.1.2 Custom Reconfiguration Scripts

Custom reconfiguration scripts provide you the ability to add custom scripts to an appliance that will be run on a virtual machine as part of deployment of the appliance. With this feature you can configure and operate a custom product or appliance that gets deployed with an Oracle product in an appliance.

To use this feature, place shell scripts in a well-known location on a reference system so that those scripts are picked up during introspection of that system. The captured scripts are executed on the VM when the corresponding operation is performed on the deployed Oracle product.

Note: This feature is not supported for external virtual machine templates imported in Oracle Virtual Assembly Builder catalog as External Appliances.

5.3.1.2.1 Custom Scripts Directory Place custom scripts into a set of sub-directories under a well-known root custom script directory named `/ovab/scripts.d/`. This directory is analogous to the Linux `/etc/rc.d/` root directory which contains a set of sub-directories with well-known names (`rc0.d/, rc1.d/, rc2.d/,...`). Similar to `/etc/rc.d/`, each subdirectory contains a set of one or more scripts that get executed at the appropriate time. You can create the following subdirectories within the `/ovab/scripts.d/` directory:

- pre-config.d/
- post-config.d/
- pre-start.d/
- post-start.d/
- pre-stop.d/
- post-stop.d/

Note: It is not necessary to create the custom script directories that you do not need.

These directories correspond to three actions performed on the VM: config, start, and stop. Custom scripts located in directories that start with "pre-" get executed before the corresponding action is performed on the deployed Oracle product and custom scripts located in directories that start with "post-" get executed afterwards.

The "config" action is executed only once at initial deployment after the VM has fully started. The "start" action is executed after a "config" action and at any other time when the deployed Oracle product is started as part of assembly start operation initiated from Oracle Virtual Assembly Builder. The "stop" action is executed when deployed Oracle product is stopped as part of assembly stop operation initiated from Oracle Virtual Assembly Builder.

At the end of introspection, Oracle Virtual Assembly Builder checks for the existence of custom script directories on the reference system and adds any found scripts to the appliance.

Example 5–1 shows a root custom script directory:

Example 5–1 Root Custom Script Directory

```
/ovab/scripts.d/
    pre-config.d/
        00configthis.sh
        01configthat.sh
    post-config.d/
        00configotherthing.sh
    pre-start.d/
        00startthisfirst.sh
        01startthatsecond.sh
    post-start.d/
        00startotherthinglast.sh
```

```
...
```

The scripts are added automatically to the appliance template with the rest of the appliance metadata at introspection time.

5.3.1.2.2 Script Execution Oracle Virtual Assembly Builder executes scripts as follows:

- All scripts are launched as root at deploy time by Oracle Virtual Assembly Builder. Custom scripts are responsible for switching to another user as needed.
- Scripts are executed one at a time in lexicographical order.
- Custom scripts must complete in a timely fashion. The action being performed can only complete after all necessary custom scripts have executed and the action has also been performed against the deployed Oracle product.
- No arguments are passed to scripts.
- The exit status of custom scripts is ignored.
- As scripts are launched, details of each launch is recorded on the VM in a file named "command.out" located in the `/assemblybuilder/logs/` directory. The output of each script, unless otherwise redirected by the script, is sent to a separate file in the `/assemblybuilder/logs/` directory. The name of each file is recorded in "command.out" as each script is executed.

5.3.1.3 Custom Appliance Properties

This feature allows an introspection to pick up properties from one or more property file on the reference system during introspection, and for those properties to be added to the resulting appliance/assembly as user properties so they can be edited.

During deployment, the set of original properties including any modifications by the user are made available to scripts provided by the user to perform custom processing based on those properties.

5.3.1.3.1 Properties Directory To get property files picked up automatically during dehydration, the user will need to place the property files in the well-known directory:

```
/ovab/scripts.d/properties/
```

This directory must reside on the same machine as the underlying product that is being captured. Within that directory, property files must conform to the following naming scheme:

```
<filename>.userprops
```

The `<filename>` must not contain ':' as this character will serve as a delimiter in property name generation.

Files within the properties directory that do not conform to the above naming scheme will be blindly transferred with the Appliance without reading it to generate additional user properties. This allows the user to provide additional files that may contain internal properties or other information to aid in the processing of user properties during reconfiguration.

5.3.1.3.2 Property File A property file must consist of zero or more lines where each line must be a property declaration, a comment, or a blank line. More formally, a property file must comply with the following syntax:

```
property-file   = *line
line            = prop-decl | comment | blank-line
prop-decl       = name "=" value NL
```

```
comment         = *WS "#" *CHAR NL
blank-line      = *WS NL
name            = <any character in "a".."z", "A".."Z", "0".."9", "_", "-", ".">
value           = *XATTRCHAR
XATTRCHAR       = <any CHAR, escaped as necessary for XML element attribute
interpretation>
CHAR            = <any character, excluding CTL (and NL), but including WS>
CTL             = <any control character (octets 0 -31) and DEL (127)>
NL              = <platform dependent line termination sequence>
WS              = <white space character>
```

Any property file that does not comply with the above syntax rules will result in an error and an appliance will not be created.

Property declarations must be contained on a single line. Ending a line with "\" will not result in line continuation.

All properties will be marked as "required" in the appliance metadata. Property declarations without any assigned value (nothing after "=") will be set to null in the appliance metadata requiring that the user assign a value to that property prior to deployment.

Whitespace is not permitted anywhere to the left of "=" in a property declaration. Whitespace to the right of "=" is assumed to be part of the intended value and will be preserved.

Comments and blank lines are preserved at dehydration and will be reproduced when the file is regenerated at rehydration.

5.3.1.3.3 Property Names Each *.userprops file in the properties directory will be read and an appliance user property will be generated for each property in each file. Property names will be modified by adding a prefix to designate that the property is a custom property and that the property belongs to a specific properties file as follows:

```
custom:<filename>:<propname>
```

The `<filename>` part comes from the properties filename with the ".userprops" suffix removed. The `<propname>` part is copied directly without modification from the property name found in the property file.

The user, during editing, will see the entire property name. At reconfiguraiton when the property files are recreated, the "`custom:<filename>:`" prefix will be removed and will not appear in the property files (that is, the property names originally found in the files will be preserved in the recreated files).

5.3.1.3.4 Property Values As indicated in the Property File section above, property values must conform to the requirements of XML element attributes. Any necessary escaping of characters in property values is the responsibility of the user when creating property files.

5.3.1.3.5 At Deployment During reconfiguration, the user properties in the appliance will be traversed and all properties with a "`custom:<filename>:`" prefix will be added to a properties file under the indicated filename (with ".userprops" suffix added).

The order of properties and the comments within the original properties file are preserved in the regenerated properties file.

All generated properties files will be placed into the same directory. The full path to that directory will be passed to all reconfiguration scripts as an environment variable with the following name:

```
$AB_CUSTOMPROPS_DIR
```

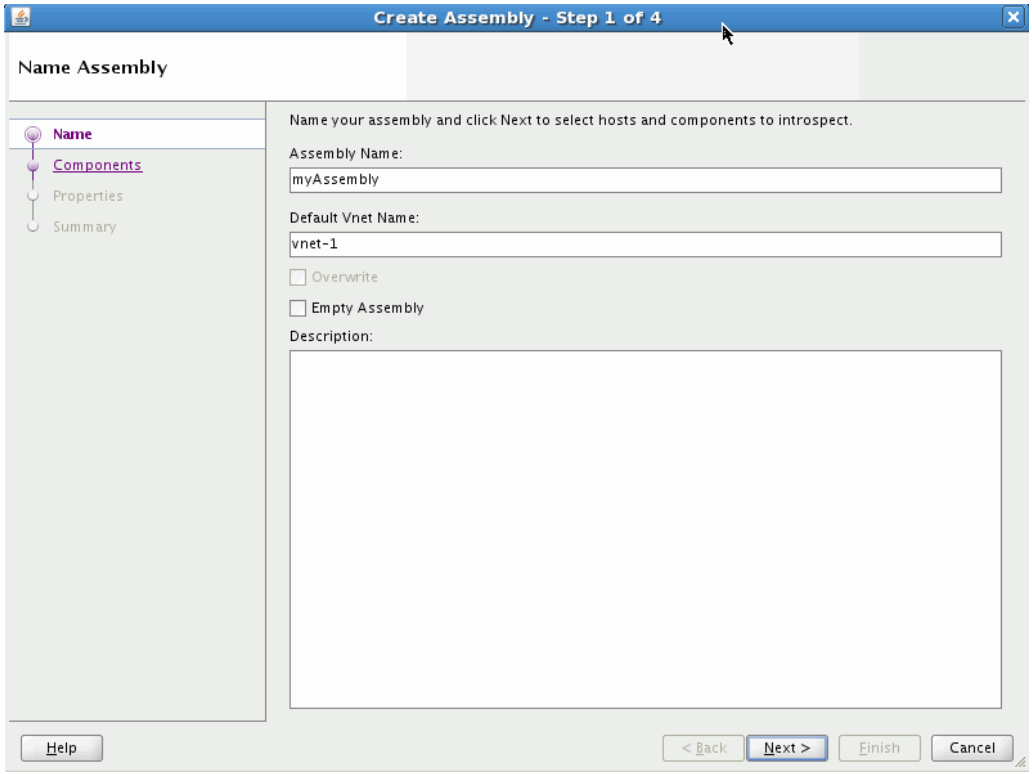
5.3.1.4 Introspect Using Oracle Virtual Assembly Builder Studio

The Create Assembly dialog allows you to create a new assembly, name it, select which appliances to introspect, and provide values required by the Introspector for those appliances. You can access this dialog by selecting **File** > **New** > **Assembly**.

In the *Name Assembly* pane (Figure 5–2), enter information for the following fields:

- Assembly Name: a name for the new assembly
- Default Vnet Name: the default name of the Vnet
- Description: Optional. Enter a textual description.
- Empty Assembly: select to create an empty assembly. You do not define any appliances or properties for the assembly, but instead select **Finish**.
- Overwrite: If an appliance or assembly with the same name already exists, and it has not been registered, you may overwrite it.

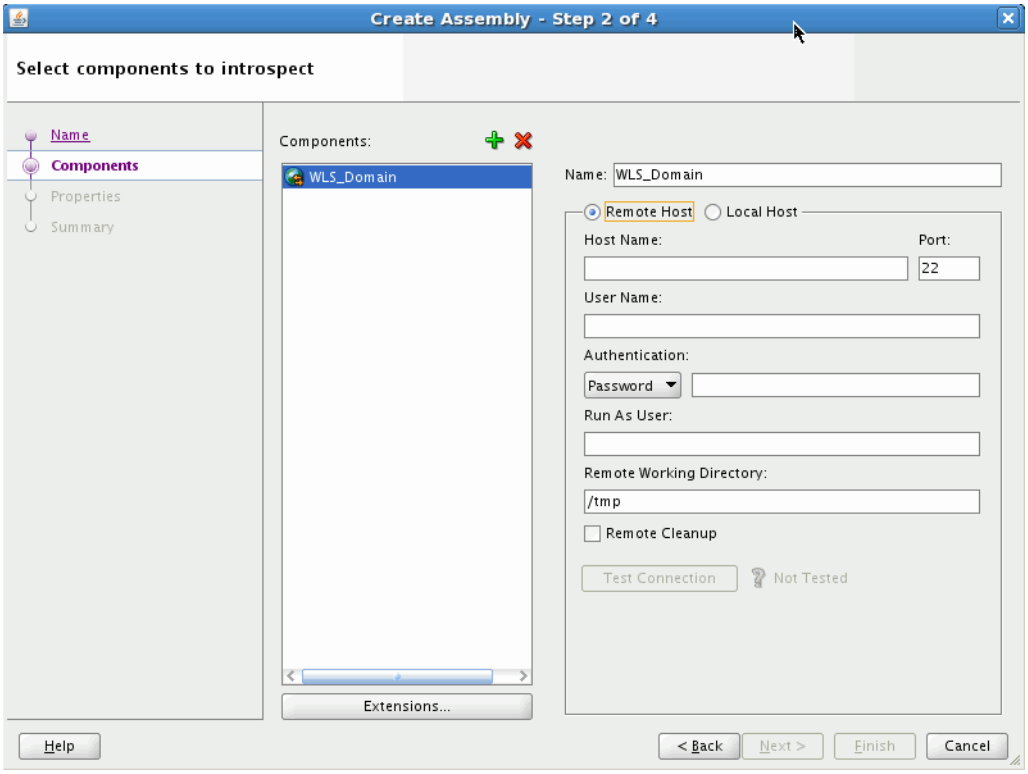
Figure 5–2 Create Assembly Wizard: Name Assembly



Click **Next** to define appliances in the assembly for introspection.

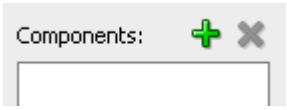
In the *Select Appliance to Introspect* window (Figure 5–3), you can define one or more appliances in the new assembly.

Figure 5–3 Create Assembly Wizard: Select Appliances



Add appliances to the assembly by selecting the + icon (Figure 5–4):

Figure 5–4 Defining Appliances in the New Assembly



This displays the list of appliances that are supported for introspection:

- Generic Product
- Oracle Coherence*Web (Alias of WLS)
- Oracle HTTP Server
- Oracle Database
- Oracle Forms*Web
- Oracle RAC Database
- Oracle Reports
- Oracle SOA (Alias of WLS)
- Oracle Web Cache
- Oracle WebLogic Server
- Oracle Traffic Director
- Oracle Tuxedo

You can name the appliance, specify a local or remote host, and a working directory (this is a directory used during remote introspection to copy configuration files locally for caching purposes).

If you specified a remote host, you must define its connection and authentication parameters.

In the *Run As User* field you may enter the name of a user on the remote machine to sudo as before executing operations. For example, if you log in with 'User Name' bob and 'Run As User' jill, the introspection process will run as jill, not bob. In that case, bob must do a sudo operation to jill.

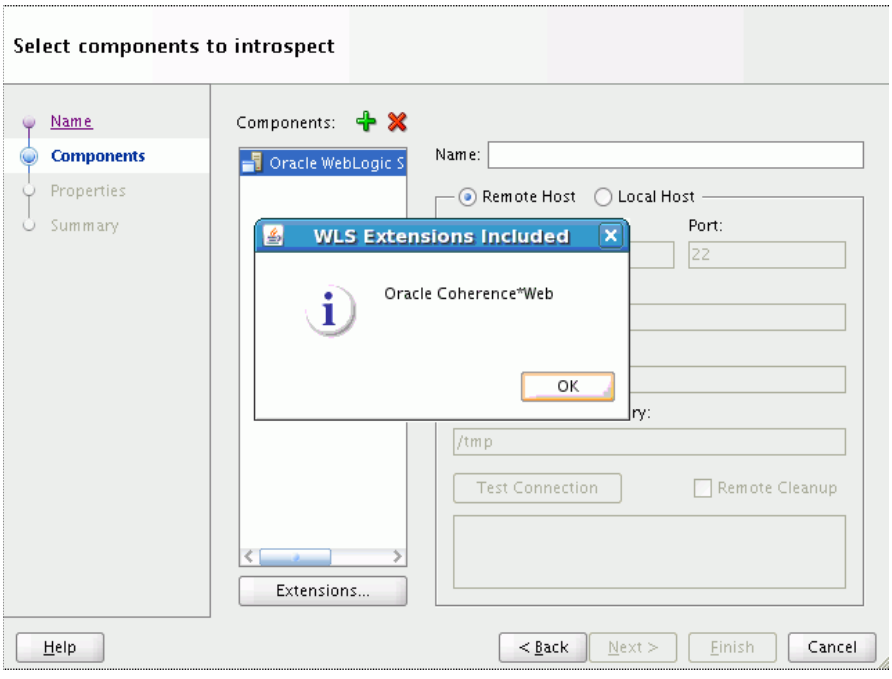
After defining parameters for the remote host, select **Test Connection** to verify that you can create an SSH connection using the supplied credentials to the remote host.

You can select **Remote Cleanup** to remove the artifacts copied over to the Remote Working Directory once the Introspection is complete.

Note: You cannot perform remote introspection of a database if you cannot log into the database machine with the database installation owner's account. If remote introspection is required, you must enable the account for remote access.

5.3.1.4.1 Viewing Extensions Click the **Extensions** button to see a list of included extensions for the selected type (that will automatically get executed).

Figure 5–5 Viewing Extensions for an Appliance

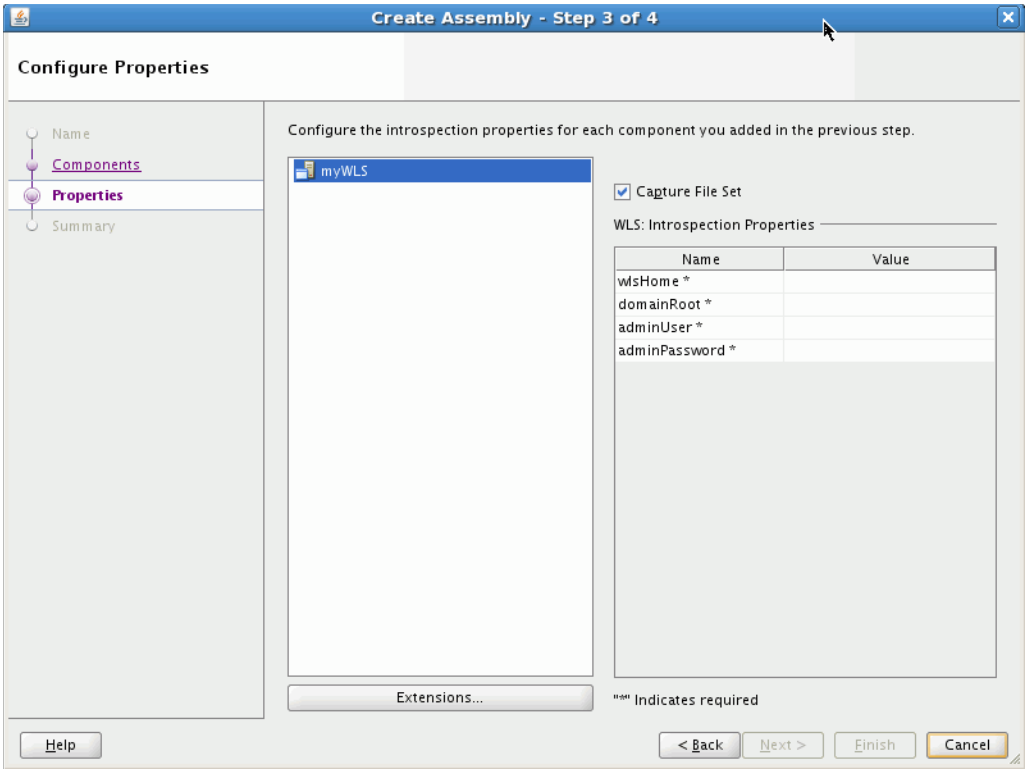


5.3.1.4.2 Authentication

Select **Password** and enter a password or select **Private Key** to reference the SSH key to use rather than providing a password. If selecting Private Key, select the browse button and navigate to the location of a private SSH key file on the local machine. The use of a private key file provides added security because no password handling is required by Oracle Virtual Assembly Builder.

5.3.1.4.3 Configuring Properties In the *Configure Properties* window (Figure 5–6) you can set the introspection properties for the appliances you included for introspection in the previous window. Required properties are identified with an asterisk next to the property name.

Figure 5–6 Create Assembly Wizard: Configure Properties



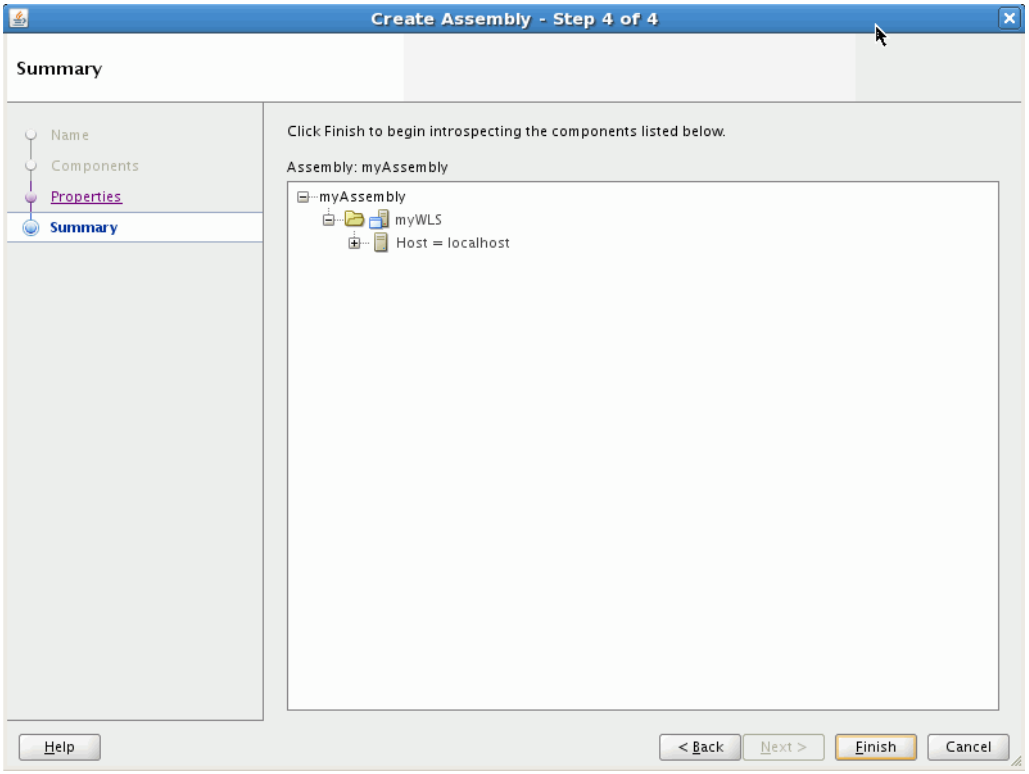
To edit a value for a property, select the appliance from the Appliances pane, and select the property from the Introspection Properties pane. Enter a value for the property. See Appendix B, "Oracle Virtual Assembly Builder Introspection Plug-ins", for information on introspection properties.

5.3.1.4.4 Capturing File Sets The **Capture File Set** checkbox is selected by default. This option captures the files from the reference system specified by the file set definition created during introspection. The files are archived into one or more zip (or other raw) files which are stored in the shared area of the catalog. You should only unselect this checkbox if you do not want the file sets captured during introspection because you intend to customize your file sets.

5.3.1.4.5 Summary of Appliances for Introspection Once you have set values for all required properties for all appliances, click **Next** (to see a summary) or **Finish** (to begin introspection without seeing a summary).

The *Summary* window (Figure 5–7) displays a logical tree view of the appliances you selected for introspection, their hosts, and the introspection properties entered.

Figure 5–7 Create Assembly Wizard: Summary



Click **Finish** to begin the introspection. A confirmation box appears informing you that the operation is time and resource intensive. Once you select **OK** to confirm, introspection starts.

You can see the progress of the introspection in the catalog navigator. Oracle Virtual Assembly Builder Studio displays a node for the appliance being introspected. If introspection fails, Oracle Virtual Assembly Builder Studio provides a link to a log for that appliance. See also Appendix D, "Troubleshooting".

5.3.1.5 Introspect Additional Appliances

The introspection wizard is a standalone interface to allow you to add a single appliance to a new or an existing assembly. To access it, select **File** > **New** > **Appliance Introspection**.

In the *Name Appliance* window, name your component and decide whether to create as a child of an existing parent assembly, or as a standalone component. Enter the following information:

- *Appliance Name*: Name your appliance; any string is acceptable. The name can be 4 to 40 characters, may not start with a digit, and no spaces or special characters are allowed (underscores are allowed). Assembly and appliance names are not allowed to be localized.
- *Overwrite*: If introspecting at the top level, you can check this box to overwrite any top-level assembly or appliance object, provided that it is not registered. If you are introspecting into an existing assembly, checking this box overwrites only assemblies and appliances inside that assembly.
- *Parent Assembly*: (Optional) Choose a Parent Assembly from the drop-down list or select <no selection> to place the new appliance at the top of the catalog.

- *Description*: Enter an optional description.

Click **Next**.

In the *Identify Host* window, you identify the host on which the appliance you want to introspect is running, by entering the following information:

- *Remote* or *Local Host*: Select *Remote Host* or *Local Host*.

  If you selected Remote Host, configure the following information:

  - *Host Name*: Enter the name of the host that you want to introspect.
  - *Port*: Enter the port number for SSH for this host. The default port number is 22.
  - *User Name*: Enter the username for the SSH user to log into the remote host. This user must have permissions to access the introspected configuration.
  - Authentication: Select **Password** and enter a password or select **Private Key** to reference the SSH key to use rather than providing a password. If selecting Private Key, select the browse button and navigate to the location of a private SSH key file on the local machine. The use of a private key file provides added security because no password handling is required by Oracle Virtual Assembly Builder.
  - *Run As User*: Enter the name of a user on the remote machine to sudo as before executing operations.
  - *Remote Working Directory*: Enter the path to a directory on the remote host in which Oracle Virtual Assembly Builder may stage files required for introspection. The files may be reused.
  - *Remote Cleanup*: Click this check box to remove the artifacts copied over to the Remote Working Directory once the Introspection is complete.

  Note: You cannot perform remote introspection of a database if you cannot log into the database machine with the database installation owner's account. If remote introspection is required, you must enable the account for remote access.

In the *Select Appliance Type* window, you identify the type of appliance you want to introspect, by entering the following information:

- *Type*: Choose the appliance type you want to introspect from the *Type* drop-down menu.

Depending on the type of appliance chosen, different sets of properties are displayed. Set the properties for that appliance by selecting the cell for the property and entering a value for the property. Required properties are identified with an asterisk.

5.3.1.5.1 Viewing Extensions Click the **Extensions** button to see a list of included extensions for the selected type (that will automatically get executed).

5.3.1.5.2 Capturing File Sets During Introspection The **Capture File Set** checkbox is selected by default. This option captures the file set definitions generated from introspection, archives the file sets into one or more zip (or other raw) files, and stores the resulting files in the shared area of the catalog. You should only unselect this checkbox if you do not want the file sets captured during introspection because you intend to customize your file sets.

Click **Finish**.

5.3.1.6 Introspect Using abctl

`abctl` provides both local and remote introspection capability. For remote introspection, the Oracle Virtual Assembly Builder host must have SSH access to the subject machine.

The `-name` flag is optional.

Here are two examples:

Example 5–2 Introspect Oracle HTTP Server Remotely

```
$ ./abctl introspectOHS -oracleInstance /path/to/oi -componentName ohs1 -name
myOHS -remoteHost myReferenceSystemHost -remoteUser abdemo
```

Example 5–3 Introspect Oracle WebLogic Server Locally

```
$ ./abctl introspectWLS -wlsHome /path/to/wls/wlserver_10.3 -domainRoot
/path/to/user_projects/domains/basic_domain -adminUser weblogic -name myWLS
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.3.2 Capture File Sets for an Appliance or an Assembly

Capturing file sets takes the file set definitions generated from introspection, archives these file sets into one or more zip (or other raw) files and stores the resulting files in the shared area of the catalog.

For the capture to succeed, some plug-ins have specific requirements for the reference system's running state. Table 5–2 lists the preconditions for the products supported by Oracle Virtual Assembly Builder.

Table 5–2 Capture Plug-in Requirements

Introspected Product	Running State Pre-Condition
Oracle WebLogic Server	No requirement; Oracle WebLogic Server may be up or down.
Oracle Coherence*Web	No requirement; Oracle WebLogic Server may be up or down.
Oracle HTTP Server (OHS)	No requirement; Oracle HTTP Server may be up or down.
Oracle Web Cache	No requirement; Oracle Web Cache may be up or down.
Oracle RDBMS (DB)	For both Oracle Virtual Assembly Builder Studio and abctl, the database must be down when capturing file sets is done as part of introspection. For `abctl`, the database must be down in the capture file sets phase (`abctl captureFileSets` command). For Oracle Virtual Assembly Builder Studio, the database must be down in the creating template phase.
Oracle Tuxedo	No requirement; Oracle Tuxedo may be up or down.

5.3.2.1 Capturing File Sets Using Oracle Virtual Assembly Builder Studio

The capturing file sets operation is available in the Template Creation Wizard, which is described in Section 5.3.3, "Create Templates for an Appliance or an Assembly".

Configuring Local and Shared File Sets

In the *Configure File Set Definitions* window, you configure local and shared file sets for appliances (Figure 5–8). The file sets that appear are configured by the introspector plug-in (at this point they are read-only).

You can add file sets. For example, in Oracle HTTP Server there is a `DocumentRoot` path, and you may want to capture this file set. The new file set is marked *Editable* and *Sharable*.

File Set Details

Configure these parameters for a file set:

- Name: Enter a name for the file set.
- Root Directory: Enter a root directory.
- OS Owner and OS Group: For each file set, a you can specify an OS owner and group. The product makes no guarantees that the owner and/or group is defined in the base image. During deployment, the file set is expanded with the owner and group specified. The owner and groups defaults to "oracle."
- Exclusions: For each file set you can specify multiple locations under the root directory that should *not* be captured. The locations are relative to the root directory. The following patterns are allowed in the exclusions:
  - a literal path, for example foo/bar
  - any * in a trailing file/directory name, for example:
    * foo/bar/*
    * foo/bar/*.log
    * foo/bar/tmp.*

    The difference between foo/bar and foo/bar/* is that foo/bar removes bar, whereas foo/bar/* removes everything under foo/bar, but not foo/bar itself.

Note: The base image must have the owners and groups defined.

Capture File Set

Creates a directory where you can capture the file set definition in this file set. The file set definition is the set of instructions used to build a file set. Both local and remote file sets can be captured if the file system type allows it. See "File System Type".

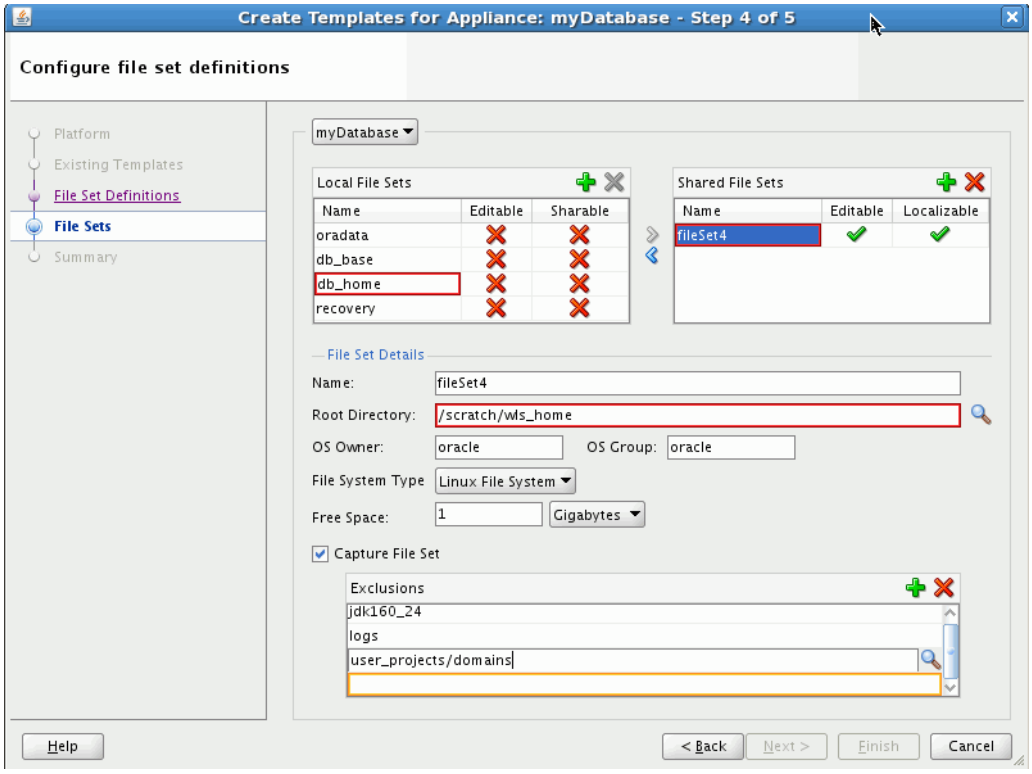
Shared File Sets

Editable describes a file set where you can modify any of the File Set Details. *Sharable* describes a local file set that you can change to a shared file set, and Localizable describes a shared file set that you can change to a local file set.

You can configure each file set as either shared or local. You can use the shuttle buttons to migrate a file set in one direction or the other. If shared, users may or may not decide to capture the file set. In some cases a file set may not be allowed to migrate from local to shared or vice versa. This is defined by the introspector plug-in that created the initial file set and Oracle Assembly Builder Studio follows that setting.

You can also specify mount options for a shared file set. This occurs only in the property inspector view during deployment plan editing.

Figure 5–8 Local and Shared File Sets



File System Type

You can specify the free space size for a given file set (Figure 5–9). Each file set can have a defined free space. This value is set on a file set by file set basis. For local file sets, you can also elect to not capture the file set, and create an empty space on the VM.

Select the file system type. For local file sets, valid choices are:

- Linux
  - Can capture a file set
  - Can define free space

For Shared File Sets, valid choices are:

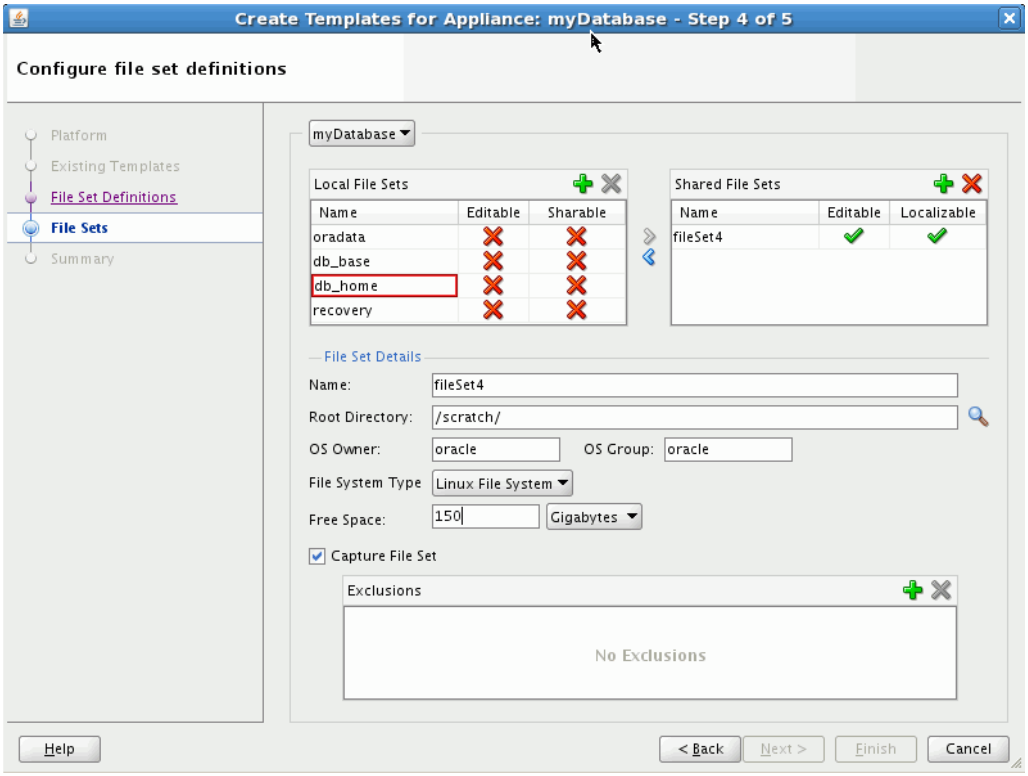
- NFS
  - Can capture a file set
  - Cannot define free space
- RAW
  - Cannot capture a file set
  - Can define free space
- Linux
  - Can capture a file set
  - Can define free space

Define Free Space for a File Set

Define a free space for the given file set. Select Megabyte, Gigabyte, or Percent for the free space unit, and enter a value.

The free space unit can only be defined as Percent if the file set is captured. A RAW file system type cannot use Percent because RAW can never capture a file set.

Figure 5–9 Defining Free Space for a File Set



5.3.2.2 Review

In the *Review* window, you can review the capture file sets decisions you have made, then click **Finish**.

Progress messages are posted in the message log window. You can open and review the Assembly Status Overview by selecting the **Template Creation** tab to verify that progress is occurring.

5.3.2.3 Capture File Sets Using abctl

The `introspect*` commands in `abctl` currently capture file sets at the end of introspection by default. This can be overridden with a flag to allow separate capture of file sets via the `captureFileSets` command.

`abctl` provides both local and remote file set capture capability. For remote file set capture, the Oracle Virtual Assembly Builder host must have SSH access to the subject machine.

Here are two examples:

Example 5–4 Capture File Sets for Oracle HTTP Server Remotely

```
$ ./abctl captureFileSets -name myOHS -remoteHost myReferenceSystemHost
-remoteUser abdemo
```

Example 5–5 Capture File Sets for Oracle WebLogic Server Locally

```
$ ./abctl captureFileSets -name myWLS -force
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command along with a sample output of the command.

5.3.3 Create Templates for an Appliance or an Assembly

Template creation generates virtual machine templates that are ready to be deployed into virtualized platforms. In Oracle Virtual Assembly Builder, Oracle VM is the only supported platform. Oracle Virtual Assembly Builder supports Oracle Enterprise Linux guest OS for all appliances.

To create a template, you must provide a system base image that contains the operating system. You may create your own system base image if the sample system base image does not meet your needs.

Oracle Virtual Assembly Builder provides a sample system base image for Oracle Enterprise Linux templates. When creating Oracle Enterprise Linux templates, Oracle Virtual Assembly Builder transparently invokes Oracle VM's `modifyjeos` tool to create the virtual machine templates. The tool allows you to modify or customize the base image (for example, adding disk space to the base image, or specifying certain RPMs). Refer to "System Base Images" in *Oracle Virtual Assembly Builder Installation Guide* for details on how to create a custom system base image.

5.3.3.1 Storage of Templates

Templates are stored in the Oracle Virtual Assembly Builder instance's catalog directory. Template creation must be done on an Oracle Virtual Assembly Builder Host, where Oracle VM's modifyjeos is installed.

Note: Base images are stored in either $AB_INSTANCE, or in $ORACLE_HOME. Here is the order of precedence for base image detection:

- location specified by `-baseImage` flag
- $AB_INSTANCE/templates/baseImage/OVM/OEL
- $ORACLE_HOME/templates/baseImage/OVM/OEL

5.3.3.2 Selecting a Target Platform

You can select either an Oracle VM or Oracle Exalogic target platform. If you select Oracle Exalogic, you must choose a base image which is compatible (that is, has the proper RPMs installed) with the Oracle Exalogic platform. You can also use the wizard to validate a base image for Oracle Exalogic.

5.3.3.3 Create Templates Using Oracle Virtual Assembly Builder Studio

This operation allows you to create templates for an assembly by selecting **Create Template** from the Assembly Node Context Menu, or **Catalog** > **Create Template**.

In the *Specify Image Location* window (Figure 4-x), you select a target platform and configure the base images as follows:

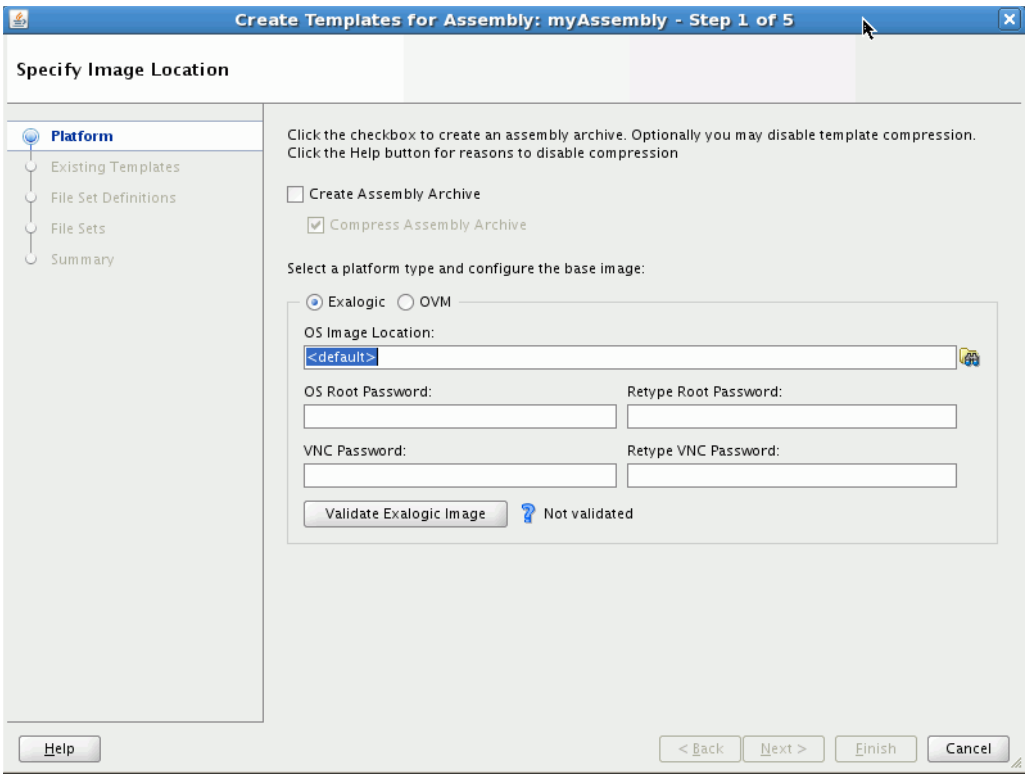
- Select either an Oracle VM or Oracle Exalogic target platform. If you select Exalogic, you must choose a base image which is compatible (that is, has the

proper RPMs installed) with the Oracle Exalogic platform. This value is meaningful to both the individual appliance templates as well as the assembly archive. An appliance may have both Oracle Exalogic and Oracle VM templates at the same time. However only one assembly archive may exist at any given time.

- Specify a location for the base images by selecting the browse icon, navigating, and selecting the base image. If you have placed your base image in one of the default locations you can leave this field as is.
- Enter OS Root and VNC passwords.

If you selected an Oracle Exalogic target platform, click **Validate Exalogic Image** to verify whether the base image has the proper RPMs installed for the Oracle Exalogic platform.

Figure 5–10 Create Template Wizard: Specify Image Location



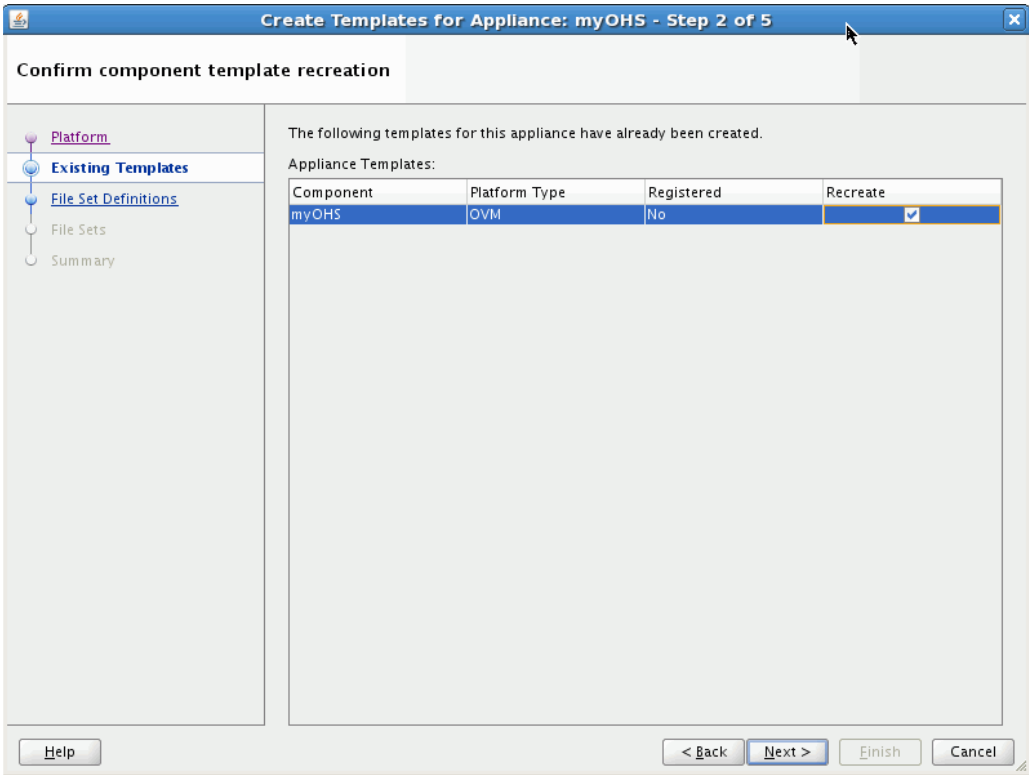
Click **Next** to continue.

> **Note:** Refer to Section 5.3.2, "Capture File Sets for an Appliance or an Assembly" for descriptions of capturing file sets in the *Create Templates* wizard.

The *Summary* window lists all the templates that will be created after you click **Finish**. It also shows a warning that creating templates can take some time.

5.3.3.3.1 Recreating a Template If the assembly appliances have already had a template created, you can recreate the templates. The *Confirm Component Template Recreation* window identifies appliances that have an associated template that has been already created. You can select those that need to be recreated by placing a check in the **Recreate** box.

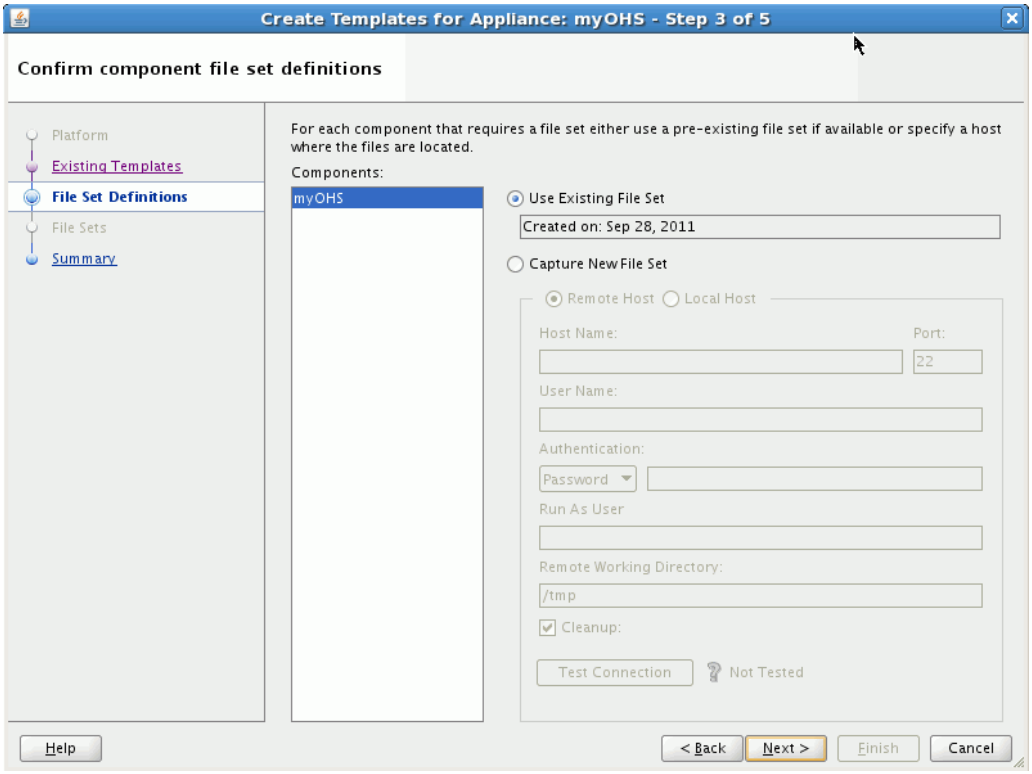
Figure 5–11 Recreating an Appliance Template



Click **Next**.

In the *Confirm Component File Set Definitions* window, select whether to use a pre-existing file set if available, or specify a host where the files are located.

Figure 5–12 Confirming Appliance File Set Definitions



Click **Next**, then click **Finish** to recreate the templates.

Progress messages are posted in the message log window. You can open and review the Assembly Status Overview by selecting the **Template Creation** tab to verify that progress is occurring.

5.3.3.4 Create Assembly Archives Using Oracle Virtual Assembly Builder Studio

You can create an assembly archive for a given assembly when you initiate the create template wizard in the context of a non-atomic assembly. Additionally you have the option of compressing the assembly archive.

When the assembly archive completes successfully, the assembly is locked. At this point you will not be able to make any changes to the assembly without explicitly deleting the assembly archive.

You can explicitly delete an assembly archive by selecting the assembly archive in the structure pane and hitting the delete key, or right-click and select **Delete**.

5.3.3.5 Create Templates Using abctl

Example 5–6 through Example 5–7 are `createTemplate` command examples:

Example 5–6 create Oracle VM Guest OS template for Oracle WLS

```
$ ./abctl createTemplate -name myWLS -platform OVM
```

Example 5–7 create Oracle VM Guest OS template for OHS

```
$ ./abctl createTemplate -name myOhs -platform OVM -baseImage
/private/baseImage/OVM/OEL/System.img
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.3.3.6 Create Assembly Archives Using abctl

You can use `abctl` to create an assembly archive for the named top-level assembly. This command can only be invoked on a top-level assembly. Additionally, you must have previously templated all the subappliances within the assembly using the `createTemplate` command.

Use the `-platform` option to specify the platform as Oracle VM or Oracle Exalogic.

Example 5–8 is a `createAssemblyArchive` command example:

Example 5–8 Create Assembly Archive

```
$ ./abctl createAssemblyArchive -name myWlsAssembly -platform OVM -nocompress
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.3.4 Edit an Assembly Using Oracle Virtual Assembly Builder Studio

This section describes how to edit an assembly, using Oracle Virtual Assembly Builder Studio.

After creating an assembly, you may need to edit the assembly before it can be deployed to create connections, and optionally, to make other changes.

5.3.4.1 Creating Connections

Managing connection configuration includes the connecting of inputs to outputs, setting or changing the property values of inputs and outputs (such as JDBC connection strings), and creating external resource appliances for those outputs that connect to components not deployed as part of deployment of the assembly.

Defining a connection means generally to connect an output to an input and set/modify the properties of the input and output as necessary. The "connect output to input" step can be achieved by either connecting an output of an appliance to an input of another or connecting an output of an appliance to the input of an external resource appliance.

All input/output properties that require values must have values set. All other properties may also need to be set/modified according to the environment that will be in place at deployment time.

For example:

- **Configure Web server port forwarding**: select an output on *myohs* and connect it to a managed server input on the *mywls* assembly by drawing an arrow between the two.
- **Specify JDBC connect strings for each JDBC connection**: open the JDBC output of an Oracle WebLogic Server assembly by selecting it.

  For JDBC connections, you either create external resources or introspect the database, then make the connection between the JDBC output of Oracle WebLogic Server to the external resource or the introspected database appliance.

  Each of the JDBC connections has a different description. Use that description to figure out which JDBC database schema to connect each to. For each of the JDBC

entries, you can look at the `original-url`, and then copy the `hostname` and `global-db-name`, into *mydb*.

For example: in `jdbc:oracle:thin:@machine999.example.com:1521/orcl`, the `hostname` is machine999.example.com, the `port` 1521, and the `global-db-name` orcl.

Also specify the `global-db-name` and `port` as properties of the external database resource input, and the host as a property of the database appliance itself.

The port is a property of the external database resource input. The hostname is the only property that belongs to the database external resource itself.

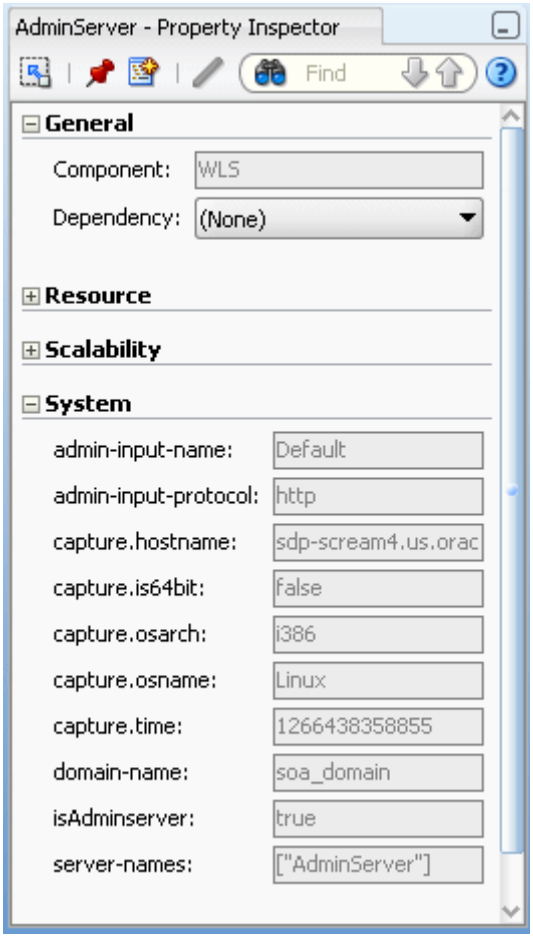
Note: Appliance-specific connection information is described in Appendix B, "Oracle Virtual Assembly Builder Introspection Plug-ins"

5.3.4.2 Property Inspector

You may not need to make changes to properties if the values from the reference system are appropriate. If required, make changes using the property inspector.

The property inspector (Figure 5–13) displays the property values. Set the properties as required.

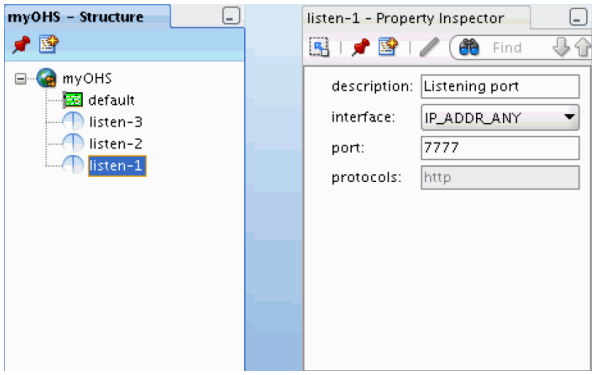
Figure 5–13 Property Inspector



5.3.4.3 Structure Pane

The Structure Pane is populated whenever you select a catalog node. When you select an assembly, the Structure Pane shows the structure of that assembly, including all appliances as well as those appliances' inputs, outputs and network interfaces. Selecting an item on the structure pane populates the property inspector with the properties scoped to that specific selection.

Figure 5–14 Structure Pane



5.3.4.4 Editing Assemblies Containing Oracle HTTP Server/Oracle Web Cache and Oracle WebLogic Server

If you have an assembly that contains Oracle HTTP Server/Oracle Web Cache and Oracle WebLogic Server with Enterprise Manager deployed, as part of deployment of Oracle HTTP Server/Oracle Web Cache "opmnctl registerInstance" is called to register that appliance with an Enterprise Manager application hosted in Oracle WebLogic Server. To enable this operation to complete successfully, you must perform the following steps while editing the assembly:

1. Define connections between Oracle HTTP Server/Oracle Web Cache's EMRegistration and Oracle WebLogic Server.
2. Use the property inspector to set the Oracle HTTP Server/Oracle Web Cache dependency on Oracle WebLogic Server. You can do this by selecting the *Dependency* drop-down menu in the *General* section.

   Without this configuration, Enterprise Manager registration will fail because the Admin Server has not been started.
3. Verify that the Oracle WebLogic Server Admin Server has not been configured to accept only SSL connections. The "opmnctl registerInstance" does not support SSL connection to Oracle WebLogic Server.

5.3.4.5 Application Routing between Oracle HTTP Server and Oracle WebLogic Server

If the Oracle HTTP Server configuration file `mod_wl_ohs.conf` defines application routing between Oracle HTTP Server and Oracle WebLogic Server, you need to connect Oracle HTTP Server to Oracle WebLogic Server in the editor.

5.3.4.6 Creating Vnets within an Assembly

When an assembly is created, a default Vnet is created for the assembly. You can change the name and description of the default Vnet when you create the assembly. Once the assembly is created, you may also create additional Vnets other than the one created by default.

To create a new Vnet right-click the assembly and choose **New** > **Vnet**... or select the assembly and right-click the Vnets folder in the Structure Pane and choose **New Vnet**... from the context menu (Figure 5–15). Enter the name and description of the new Vnet.

Figure 5–15 Create Vnet

5.3.4.7 Creating Network Interfaces within an Appliance

You can create physical or virtual network interfaces while editing an assembly. Physical interfaces are physical from the perspective of the vServer's OS stack, but since that OS is running in a vServer, that physical interface is actually a virtual network interface - thus this refers to a virtual network interface on top of another virtual network interface. To create interfaces:

- Create a physical network interface by right clicking on an appliance in the assembly editor and selecting **Add Network Interface**, or by selecting the assembly in the Assembly navigator and right clicking on the appliance or the Network Interfaces folder under an appliance and selecting **Add Network Interface**.
- Create virtual network interfaces by right-clicking on a physical network interface and choosing **Add Network Interface**. You are prompted for the name and description of the new interface. Set the other properties using the Property Inspector. The result is displayed in the Structure Pane as the child of the physical network interface. Virtual network interfaces exist to model the concept of VLAN.

5.3.4.8 Binding Appliance Inputs to Network Interfaces

When configuring an appliance Input you need to specify the network interface and port on which the Input will receive (or listen for) data. You can either select a specific network interface, in which case the Input will only listen for inputs coming from that network interface or you can specify `INADDR_ANY` in which case the Input will listen for inputs coming from any network interface configured for the appliance. `INADDR_ANY` is the default.

To set the default interface on an appliance: right-click the interface in the Structure Pane and choose **Set as Default** from the context menu. The default may be either a virtual or a physical interface. Note that virtual interfaces do not appear in the Assembly Editor. They may only be selected in the Structure Pane.

5.3.4.9 Binding Network Interfaces to Vnets

Assemblies and appliances have Vnets and network interfaces. Each network interface may be bound to one and only one Vnet. The deployment plan is used to associate Vnets and network interfaces with physical networks and IP addresses that exist in the deployment environment.

The binding can be changed to a different Vnet (if a different one exists) through the Property Inspector. Each logical network must be resolved at deploy time using the deployment plan (that is, each network will have properties in the deployment plan).

5.3.4.10 Creating External Resources from an Appliance Output

Create an external resource by right-clicking on an unconnected appliance output in the assembly editor, and selecting **External Resource**.

5.3.5 Edit an Assembly Using abctl

This section describes assembly editing operations you can perform using `abctl`.

Creating Empty Top-level Assemblies

Use the `createAssembly` command to create an empty top-level assembly. Example 5–9 shows the `createAssembly` command.

Example 5–9 createAssembly Command

```
$ ./abctl help -command createAssembly
$ ./abctl createAssembly -name myAssembly -defaultNetwork intranet
```

Adding Appliances (or WLS Assemblies) to Top-level Assemblies

Use the `addToAssembly` command to add appliances to a top-level assembly. Example 5–10 shows the `addToAssembly` command.

Example 5–10 addToAssembly Command

```
$ ./abctl help -command addToAssembly
$ ./abctl addToAssembly -name myAppliance -into myAssembly
```

Connecting Outputs to Inputs

Use the `connectEndpoints` command to creates a new connection between an output and an input. The protocols of the output and input must match, and the owners of the output and input must be part of the same assembly. Example 5–11 shows the `connectEndpoints` command.

Example 5–11 connectEndpoints Command

```
$ ./abctl help -command connectEndpoints
$ ./abctl connectEndpoints -from mySite/myOhs -fromOutput output1 -to mySite/myWls
-toInput default
```

5.4 Operations Related to Deployment

This section details how you will use Oracle Virtual Assembly Builder Studio or the `abctl` command line utility to perform operations related to deployment.

- Section 5.4.1, "Configuring Targets"
- Section 5.4.2, "Creating a Deployment Plan"
- Section 5.4.4, "Registering an Assembly Archive to a Target"
- Section 5.4.5, "Deploying an Assembly Instance"
- Section 5.4.6, "Stopping an Assembly Instance"
- Section 5.4.7, "Starting an Assembly Instance"
- Section 5.4.9, "Scale Appliance"
- Section 5.4.10, "Undeploying an Assembly Instance"
- Section 5.4.11, "Unregistering an Assembly Archive from a Target"
- Section 5.4.12, "Uploading an Assembly Archive to the Deployer Repository"
- Section 5.4.13, "Deleting an Assembly Archive from the Deployer Repository"
- Section 5.4.14, "Export Operations"
- Section 5.4.15, "Interacting with EM Software Library"

5.4.1 Configuring Targets

This section describes how to define the connection to the Oracle VM or Oracle Exalogic backend endpoints, to provide credentials if required, and add deployment targets in the backend.

5.4.1.1 Connection URL

Oracle recommends that you configure your target connections for Oracle VM 3 with TCP instead of HTTP protocol.

To configure with TCP, specify a URL of the form "tcp://their-ovm-host:54321".

5.4.1.2 Connection Credentials

To configure connection credentials:

1. For Oracle VM, create the target using the `createTarget` command in abctl.

   This operation, which can only be performed by the Cloud Admin, defines the connection information, and, depending on the backend type, user credentials for the backend. For Oracle VM, credentials are supplied here.

   For Oracle Exalogic, you do not perform the `createTarget` command because a single target is preconfigured. Individual users will provide their own credentials in the `addTargetUser` step.

2. Add users to the target using the `addTargetUser` abctl command.

   Depending on the backend type, this may be a Cloud Admin call or an Application Admin, depending on differences in the security models of the backend systems:

   - For Oracle VM targets, this an operation that can only be performed by the Cloud Admin, and is used to control what users can access the pool. This is an Cloud Admin operation because the credentials supplied by the Cloud Admin must be protected from general users.
   - For Oracle Exalogic targets, this is an Application Admin operation and is used to specify Application Admin credentials (in this case, Cloud Admin does not have to give specific access because the backend will be checking the credentials). This is an Application Admin call because the user's credentials must be protected from others, including the Cloud Admin.

5.4.1.3 Examples

Example 5–12 shows how to create a target for Oracle VM:

Example 5–12 Create Target for Oracle VM

```
./abctl createTarget -name slcTarget_tcp -type ovm -properties ovm.poolName=ab_
ovm_30_stand_alone_pool ovm.vmOperationTimeout=3600 ovm.vmmversion=3.0
ovm.user=admin ovm.pwd ovm.url=tcp://example.oracle.com:54321 -connectionName
localDeployer
```

Example 5–13 shows how to add users to a target for Oracle VM:

Example 5–13 Add Users to a Target for Oracle VM

```
./abctl addTargetUser -user Username -target Targetname
```

Example 5–14 shows how to add users to a target for Oracle Exalogic, where the `-properties` option contains the user's credentials as a property=value pair:

Example 5–14 Add Users to a Target for Oracle Exalogic

```
./abctl addTargetUser -user Username -target Targetname -properties String...
```

5.4.2 Creating a Deployment Plan

A deployment plan allows you to override property values defined in the metadata of an assembly. The plan is applied when the assembly is deployed. Only top-level assemblies can have deployment plans.

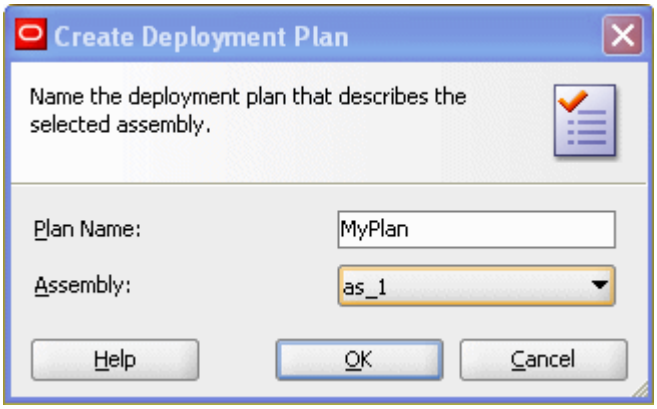
5.4.2.1 Creating a Deployment Plan Using Oracle Virtual Assembly Builder Studio

Create a deployment plan:

1. In the Deployment Plans navigator, right-click an assembly and select **New Deployment Plan**.

   The *Create Deployment Plan* wizard appears (Figure 5–16).

Figure 5–16 Create Deployment Plan



This graphics displays the Create Deployment Plan window, which is described in the surrounding text.

****

2. Enter the name for the deployment plan.
3. Select the associated assembly from the Assembly drop-down menu.
4. Click **OK**. The Deployment Plan editor opens.

5.4.3 Editing a Deployment Plan

The Deployment Plan Editor displays a read-only view of the assembly. This view is useful as an overview and for selecting items so that their property values may be overridden.

5.4.3.1 Editing a New Deployment Plan

When you create a new deployment plan, the Deployment Plan editor opens automatically.

5.4.3.2 Editing an Existing Deployment Plan

Edit an existing deployment plan:

1. In the Deployment Plans navigator, right-click the plan and select **Open**.

   Alternatively, in the Deployment Plans navigator you may double-click the plan icon.

5.4.3.3 Required Views

Deployment plan editing makes use of both the Structure Pane and the Property Inspector. You should ensure both views are visible.

1. Open the Structure Pane by selecting **View** > **Structure**.
2. Open the Property Inspector by selecting **View** > **Property Inspector**.

5.4.3.4 Selecting items in the Deployment Plan Editor

To override the properties of an appliance, input, output or other item, you must first select the item.

To select an item in the Deployment Plan editor, click on it.

5.4.3.5 Selecting items in the Structure Pane

When populated, the Structure Pane shows additional details of the plan.

Populate the Structure Pane:

1. In the Deployment Plans navigator, select the plan.

   The Structure Pane populates with the assembly structure. To select an item in the populated Structure Pane, click on it.

   Selecting any item in the Deployment Plan editor also populates the Structure Pane.

5.4.3.6 Overriding Property Values

The properties of the selected item are displayed in the Property Inspector.

To override a value:

1. In the property inspector, type a new value in the field.

   A blue bullet is displayed next to overridden values.

5.4.3.7 Removing an Override Value

To remove an override:

1. In the Property Inspector, click the downward-pointing arrow (chevron) to the right of the property value.
2. From the pop-up menu, select **Reset to**...

   The value shown in the **Reset to**... menu item is always the original value specified in the assembly metadata. The override is removed.

5.4.3.8 Synthetic Properties

You may override the scalability properties of an appliance. Doing so may change the number of potential appliance instances that can exist when the assembly is deployed. Each of these potential appliance instances may require its own network settings including hostname, IP address, MAC address, and netmask properties. The Property Inspector automatically displays the appropriate number of instance-specific network properties. When the scalability values are changed, the Property Inspector automatically adjusts the number of network properties to match the number of appliances instances that may potentially exist.

5.4.3.8.1 Appliance Properties To display the network properties of an appliance:

1. In the Structure pane or Deployment Plan editor, click to select the appliance.
2. In the Property Inspector, click to open the Network category.

The *default-gateway*, *dns-domains*, and *dns-servers* property values are required unless all network interfaces of the appliance are configured to use DHCP. These values are shared by all instances of the appliance.

The *hostname.0* property is required. If the appliance is scalable, the Property Inspector will display an appropriate number of properties (*hostname.0*, *hostname.1*, etc.) All values are required.

5.4.3.8.2 Network Interface Properties To display the Deployment Plan properties of a network interface, click on the network interface in the Deployment Plan editor or expand the Network Interfaces folder in the Structure pane and click the network interface within the folder.

Setting the *usedhcp* property of the Interface to **true** asserts that these network properties will be automatically configured in the deployment target environment. When *usedhcp* is **true**, the Property Inspector displays previously-established values of the other network interface properties, but does not permit the values of these properties to be modified.

If *usedhcp* is false, the *ip_address.0*, *mac_address.0*, and *netmask* properties are required. If the appliance is scalable, the Property Inspector will display an appropriate number of address properties (*ip_address.0*, *ip_address.1*, ..., *mac_address.0*, *mac_address.1*, ...) and all values are required.

5.4.3.8.3 Vnet Properties Vnets are not displayed in the Deployment Plan editor so cannot be selected there. To display the Deployment Plan properties of a Vnet, expand the Vnets folder in the Structure pane and select the Vnet.

The *is_private* property specifies that the network should be automatically configured as a high-performance network for internal use within the deployed Assembly. This value must be **false** if the deployment target environment does not provide the necessary platform support for high-performance private networks. Set this value to **false** when deploying directly to Oracle Virtual Machine version 3.0.

The *network_name* property specifies the name of the network defined in the deployment target environment. This property is required.

Some deployment target environments may not require that network names be unique. When network names are not unique, the optional *network_id* property can be used to uniquely specify the network defined in the deployment target environment. The value of this property is ignored when the *is_private* property is true.

5.4.3.9 Validating a Deployment Plan

To validate the deployment plan:

1. In the Deployment Plans navigator, right-click the plan and select **Validate**.

   The validation results are displayed in a dialog box.

5.4.3.10 Saving a Deployment Plan

When the deployment plan has been modified, its name is displayed in italic font in both the Deployment Plan navigator and the tab of the Deployment Plan editor.

To save the deployment plan:

1. Select **File** > **Save**.

If the File > Save menu item is not enabled, the deployment plan may not be selected. In the Deployment Plan navigator, select the plan, then select **File** > **Save**.

You can create, edit and save multiple deployment plans for a single assembly. At deployment time, you select one plan for the deployment. In this way, a single assembly may serve as the basis of multiple deployments, each with their own specific network and other property settings.

5.4.4 Registering an Assembly Archive to a Target

Once you upload an assembly archive to Oracle Virtual Assembly Builder Deployer you can register the assembly archive to a particular target.

5.4.4.1 Registering Using Oracle Virtual Assembly Builder

Perform this in one of two ways:

- In the *Deployments* navigator, under Available Assemblies, right-click on a specific assembly archive version and select **Register**.
- Drag and drop a specific assembly archive version to the intended target. In the *Deployments* navigator, select the assembly archive under Available Assemblies, and drop it onto the target in Deployment Targets.

Both methods start the registration. The target node is updated with a new child representing the assembly archive version. The target node displays a throbber icon which will switch to the standard assembly archive icon once the registration is complete. You receive message feedback in the Message Log window as the registration progresses. If the registration fails, the node disappears and a pop-up menu appears explaining the failure.

To unregister, right-click on a registered node and select **Unregister**. The node will be removed once the unregistration is complete. Progress messages are printed in the Message Log and a dialog appears in the event of a failure.

5.4.4.2 Registering Using abctl

Use the `registerAssemblyArchive` command to unregister templates for an assembly. Example 5–15 shows the `unregisterAssemblyArchive` command:

Example 5–15 unregisterAssemblyArchive Command

```
$ ./abctl help -command registerAssemblyArchive
$ ./abctl registerAssemblyArchive -connectionName MyDeployerConnection -name
TheAssembly -version 1
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.5 Deploying an Assembly Instance

This section describes how to deploy an assembly instance, using Oracle Virtual Assembly Builder Studio, or `abctl`.

When an assembly instance is deployed, appliance instances for the assembly are created and started. Furthermore, applications within the appliance instances are also started. Deploying an assembly instance can be a long running operation that can take several minutes. The time taken to deploy an assembly instance will vary depending upon the number of VMs that need to be created and started.

5.4.5.1 Ability to Set Appliance 'target' Count to Zero (Zero-Count Appliances)

You can initially deploy an assembly with zero appliance instances. However, in subsequent scaling operations you could add appliance instances to those appliances that are part of the assembly configuration but were initially "deployed" with a zero instance count.

5.4.5.2 Deploying Using Oracle Virtual Assembly Builder Studio

Once an assembly archive is registered to a specific target, you can deploy that assembly archive one or more times.

1. Open the *Deployments* navigator by selecting **View** > **Deployments**.

   In the *Deployments* navigator, under the Deployment Targets pane, you can expand the targets and see a list of assembly archives that you can deploy.

2. Right-click an assembly archive and select **Deploy**.

   You can also initiate an assembly instance by dragging a deployment plan from the Deployment Plan navigator and dropping it on a registered assembly archive.

3. Enter the following information in the *Deploy Assembly Archive* window.

   - **Deployment Plan**: select a deployment plan. The dialog populates the drop-down list with deployment plans matching the name of the assembly being deployed. If no plans exist, you must create a plan before you can deploy.
   - Vnet Mappings: verify (and possibly correct) the Vnet mappings in the deployment plan. At the time the deployment plan is created Oracle Virtual Assembly Builder cannot determine where you intend to deploy the assembly so it cannot validate the name of the target Vnet.

     You see one row for each Vnet defined in your assembly. The logical name of the Vnet (as defined in the assembly) is shown as the label and the corresponding drop-down contains the Vnets defined by the deployment target. The value from the deployment plan is included in the drop-down list, even though it may be invalid, for the purposes of displaying to you the value from the deployment plan. The value, or mapping, from the deployment plan has an asterisk (*) suffix.

     If the current mapping is invalid, the value displays a red error border and the "Deploy" button becomes disabled. You cannot deploy an assembly if the deployment plan contains invalid mappings. Any changes made to the Vnet mapping are saved to the deployment plan before the assembly instance initiates.

4. Click **Deploy** to update the deployment plan with the new mappings and initiate the assembly instance.

   Once you confirm the deployment options, the assembly instance is initiated. A new child node is created under the registered node to indicate the progress. The name of the new child node is the Deployment ID. A throbber icon is shown until the assembly instance reaches a final state.

   When the network is initialized, you can expand an appliance to see the IP addresses of each virtual machine started for that appliance.

   Success is indicated by a green icon. If the initiation of the assembly instance fails, a red icon indicates failure. Progress messages are shown in the Message Log window.

5.4.5.3 Deploying Using abctl

Create an assembly instance by using the `createAssemblyInstance` command, as shown in Example 5–16. The `createAssemblyInstance` command will return the `assemblyInstanceId` needed for the `deployAssemblyInstance` command.

Example 5–16 createAssemblyInstance Command

```
$ ./abctl help -command createAssemblyInstance
$ ./abctl createAssemblyInstance -deploymentPlan c:/zeroAppliancesSite_plan.xml
-name SMALLOVA -version 1 -c cloudAdmin
Plan upload File Size: 700
Assembly Instance Id: gdc4_29x5_SMALLOVA_1
Assembly instance has been created.
```

Once you have created the assembly instance, you can deploy an assembly instance by using the `deployAssemblyInstance` command as shown in Example 5–17.

Example 5–17 deployAssemblyInstance Command (without Waiting for Completion)

```
$ ./abctl help -command deployAssemblyInstance
$ ./abctl deployAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin
Request ID: 1d1599a0-434b-426a-ab29-7c6230b5fa33
Request to deploy assembly instance has been submitted to deployer.
```

The `deployAssemblyInstance` command is an asynchronous operation. It will initiate a deploy request and return a request ID. The status of the request can be queried using the `describeRequests` operation.

If you want to wait until the completion of the operation, you may do so, optionally, by specifying additional parameters as follows:`-waitForComplete -pollTime 30`

Example 5–18 deployAssemblyInstance Command (Waiting for Completion)

```
$ ./abctl help -command deployAssemblyInstance
$ ./abctl deployAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin -waitForComplete -pollTime 30
Request ID: 1d1599a0-434b-426a-ab29-7c6230b5fa33
Request to deploy assembly instance has been submitted to deployer.
```

You can list the current deployments with the `describeAssemblyInstances` command, as shown in Example 5–19:

Example 5–19 describeAssemblyInstances Command

```
$ ./abctl help -command describeAssemblyInstances
$ ./abctl describeAssemblyInstances -c cloudAdmin
--------------------------------------------------------------------------------
 Name     | Version | State      | Assembly Instance Id | Target  | Appliances
--------------------------------------------------------------------------------
 SMALLOVA | 3       | Undeployed | c1Lm-GyML_SMALLOVA_3 | LOCBOX1 | c1Lm-GyML_
SMALLOVA_3:zeroAppliancesSite/myWls/Server_3
Assembly Instances have been described.
$
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.6 Stopping an Assembly Instance

This section describes how to stop an assembly instance, using Oracle Virtual Assembly Builder Studio, or `abctl`.

When an assembly instance is stopped, the VMs and the applications that are running within the VMs are stopped. VMs that are in a stopped state retain their context. Stopped VMs can be restarted much more quickly than the original deployment because the VMs do not need to be created.

5.4.6.1 Stopping an Assembly Instance with Oracle Virtual Assembly Builder Studio

In the Deployment Targets pane of the *Deployments* navigator you can start, stop, deploy, or undeploy an assembly instance. To stop an assembly instance, select the assembly instance that needs to be stopped and click **Stop**.

5.4.6.2 Stopping an Assembly Instance with abctl

Use the `stopAssemblyInstance` command to stop an assembly instance. The assembly instance is referred to by its `assemblyInstanceId`. You can retrieve a list of assembly instances by using the `describeAssemblyInstances` command. Example 5–20 and Example 5–21 show the `stopAssemblyInstance` command:

Example 5–20 stopAssemblyInstance Command (without Waiting for Completion)

```
$ ./abctl help -command stopAssemblyInstance
$ ./abctl stopAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin
Request ID: 8486522b-8a5e-4348-bdf1-a7d55fccf848
Request for stop has been submitted to deployer.
```

Example 5–21 stopAssemblyInstance Command (Waiting for Completion)

```
$ ./abctl help -command stopAssemblyInstance
$ ./abctl stopAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin -waitForComplete -pollTime 30
Request ID: 8486522b-8a5e-4348-bdf1-a7d55fccf848
Request for stop has been submitted to deployer.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.7 Starting an Assembly Instance

This section describes how to start an assembly instance, using Oracle Virtual Assembly Builder Studio, or `abctl`.

5.4.7.1 Starting an Assembly Instance with Oracle Virtual Assembly Builder Studio

In the Deployment Targets pane of the *Deployments* navigator you can start, stop, deploy, or undeploy an assembly instance. To start an assembly instance, select the assembly instance and click **Start**.

5.4.7.2 Starting an Assembly Instance with abctl

The `startAssemblyInstance` command is used to start a deployment. The assembly instance is referred to by its `assemblyInstanceId`. You can retrieve the list

of deployments by using the `describeAssemblyInstances` command. Example 5–22 and Example 5–23 show the `startAssemblyInstance` command:

Example 5–22 Start an Assembly Instance (without Waiting for Completion)

```
$ ./abctl help -command startAssemblyInstance
$ ./abctl startAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin
Request ID: 1936dff2-f8a7-4407-83f8-08521bb48fef
Request for start has been submitted to deployer.
```

Example 5–23 Start an Assembly Instance (Waiting for Completion)

```
$ ./abctl help -command startAssemblyInstance
$ ./abctl startAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin -waitForComplete -pollTime 30
Request ID: 1936dff2-f8a7-4407-83f8-08521bb48fef
Request for start has been submitted to deployer.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.8 Restarting an Assembly Instance

This section describes how to restart an assembly instance, using Oracle Virtual Assembly Builder Studio, or `abctl`.

An assembly instance that has been stopped can be restarted. Restarting an assembly instance starts up all the VMs that were stopped and also starts up the applications within the VMs. The assembly instance gets restored to the state it was in before it was stopped. This operation completes more quickly than a deployment operation.

5.4.8.1 Restarting an Assembly Instance with Oracle Virtual Assembly Builder Studio

In the Deployment Targets pane of the *Deployments* navigator you can start, stop, deploy, or undeploy an assembly instance. To restart an assembly instance, select the assembly instance and click **Start**.

5.4.8.2 Restarting an Assembly Instance with abctl

The `restartAssemblyInstance` command is used to restart an assembly instance. The assembly instance is referred to by its `assemblyInstanceId`. You can retrieve the list of assembly instances by using the `describeAssemblyInstances` command. Example 5–24 and Example 5–25 show the `restartAssemblyInstance` command:

Example 5–24 Restart an Assembly Instance (without Waiting for Completion)

```
$ ./abctl help -command restartAssemblyInstance
$ ./abctl restartAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin
Request ID: 126a97ef-89db-4b05-88d5-17b70e5cc3d2
Request to restart assembly instance has been submitted to deployer.
```

Example 5–25 Restart an Assembly Instance (Waiting for Completion)

```
$ ./abctl help -command restartAssemblyInstance
```

```
$ ./abctl restartAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin -waitForComplete -pollTime 30
Request ID: 126a97ef-89db-4b05-88d5-17b70e5cc3d2
Request to restart assembly instance has been submitted to deployer.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.9 Scale Appliance

This section describes how to scale the selected appliance instance after an initial deployment of an assembly, using Oracle Virtual Assembly Builder Studio, or `abctl`.

After you deploy an assembly, the target number of VM instances for each appliance is started. The initial target for each appliance instance is specified in the deployment plan. You can dynamically specify a new target after an assembly has been deployed. Oracle Virtual Assembly Builder dynamically starts or stops VM instances to reach the new target (thus scaling up or scaling down). A scale down operation will only stop the properly deployed instances.

5.4.9.1 Scale Appliance with Oracle Virtual Assembly Builder Studio

To scale the number of instances of an appliance:

1. Select an assembly instance in the *Deployments* navigator and expand the assembly structure in the Structure Pane.
2. Right-click on a scalable appliance and select **Scale appliance**.

   In the *Scale* dialog, the minimum and maximum number of VM instances are displayed, as is the number of currently running VMs.
3. Set the target number of VM instances for the appliance from the *Target* drop-down list. Select a value between the between minimum and maximum.
4. Click **OK**.

5.4.9.2 Retrieve the scalingGroupId for Use in the Scale Command

Use the `describeScalingGroups` command to retrieve the `scalingGroupId` to pass in to the `scale` command. (Example 5–26):

Example 5–26 describeScalingGroup Command

```
$ ./abctl help -command describeScalingGroups
$ ./abctl describeScalingGroups
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.9.3 Scale Appliance(s) in an Assembly Instance with abctl

Use the `scale` command to scale the appliance (Example 5–27 and Example 5–28):

Example 5–27 scale Command (without Waiting for Completion)

```
$ ./abctl help -command scale
$ ./abctl scale -scalingGroupId 1gWT-t0Np_SMALLOVA_
1:zeroAppliancesSite/myWls/Server_3 -target 1 -c cloudAdmin
Request ID: c1d2c742-d2fe-4698-bf61-99d619be4fca
Request for scaling operation has been submitted to deployer.
```

Example 5–28 scale Command (Waiting for Completion)

```
$ ./abctl help -command scale
$ ./abctl scale -scalingGroupId 1gWT-t0Np_SMALLOVA_
1:zeroAppliancesSite/myWls/Server_3 -target 1 -c cloudAdmin -waitForComplete
-pollTime 30
Request ID: c1d2c742-d2fe-4698-bf61-99d619be4fca
Request for scaling operation has been submitted to deployer.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.10 Undeploying an Assembly Instance

This section describes how to undeploy an assembly instance, using Oracle Virtual Assembly Builder Studio, or `abctl`.

Undeploying an assembly instance stops all the running VMs. It also cleans up any failed VMs that may exist.

5.4.10.1 Undeploy a Deployment with Oracle Virtual Assembly Builder Studio

In the *Deployments* navigator, you can undeploy an assembly instance by right-clicking on a deployed assembly instance and selecting **Undeploy**. If successful, the node is removed. If the operation fails, a red icon appears. Progress messages are shown in the Message Log window.

5.4.10.2 Undeploy an Assembly Instance with abctl

You can use the `undeployAssemblyInstance` command to undeploy an assembly instance. The assembly instance is referred to by its `assemblyInstanceId`. You can retrieve a list of assembly instances by using the `describeAssemblyInstances` command. Example 5–29 shows the `undeployAssemblyInstance` command:

Example 5–29 undeployAssemblyInstance Command (without Waiting for Completion)

```
$ ./abctl help -command undeployAssemblyInstance
$ ./abctl undeployAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin
Request ID: 6b0f1b14-466e-4b23-bcc3-8b8506fd40ac
Request to undeploy assembly instance has been submitted to deployer.
```

Example 5–30 undeployAssemblyInstance Command (Waiting for Completion)

```
$ ./abctl help -command undeployAssemblyInstance
$ ./abctl undeployAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin -waitForComplete -pollTime 30
Request ID: 6b0f1b14-466e-4b23-bcc3-8b8506fd40ac
Request to undeploy assembly instance has been submitted to deployer.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.11 Unregistering an Assembly Archive from a Target

This section describes how to unregister an assembly archive from Oracle Virtual Assembly Builder Deployer using `abctl`. This operation unregisters the assembly archive, but does not delete the uploaded archive.

5.4.11.1 Unregistering Assembly Archives with Oracle Virtual Assembly Builder Studio

Unregister an assembly archive version (from a deployment target) by right-clicking on a registered version and selecting **Unregister**.

Confirm unregistering the assembly when prompted.

5.4.11.2 Unregistering Assembly Archives with abctl

Use the `unregisterAssemblyArchive` command to unregister templates for an assembly. Example 5–31 and Example 5–32 show the `unregisterAssemblyArchive` command:

Example 5–31 unregisterAssemblyArchive Command (without Waiting for Completion)

```
$ ./abctl help -command unregisterAssemblyArchive
$ ./abctl unregisterAssemblyArchive -name SMALLOVA -version 1 -target LOCBOX1 -c
cloudAdmin
Request ID: f9f9d0b7-e334-4020-a038-2b728e9a0a37
Request to unregister assembly has been submitted to deployer.
```

Example 5–32 unregisterAssemblyArchive Command (Waiting for Completion)

```
$ ./abctl help -command unregisterAssemblyArchive
$ ./abctl unregisterAssemblyArchive -name SMALLOVA -version 1 -target LOCBOX1 -c
cloudAdmin -waitForComplete -pollTime 30
Request ID: f9f9d0b7-e334-4020-a038-2b728e9a0a37
Request to unregister assembly has been submitted to deployer.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.11.3 redeployAssemblyInstance

Use the `redeployAssemblyInstance` command to redeploy an assembly instance. Example 5–33 and Example 5–33 show the `redeployAssemblyInstance` command:

Example 5–33 redeployAssemblyInstance Command (without Waiting for Completion)

```
$ ./abctl help -command redeployAssemblyInstance
$ ./abctl redeployAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin
Request ID: eff86a4c-d064-4794-b1ae-0624a972ab06
Request to redeploy assembly instance has been submitted to deployer.
```

Example 5–34 redeployAssemblyInstance Command (Waiting for Completion)

```
$ ./abctl help -command redeployAssemblyInstance
$ ./abctl redeployAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin -waitForComplete -pollTime 30
Request ID: eff86a4c-d064-4794-b1ae-0624a972ab06
Request to redeploy assembly instance has been submitted to deployer.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.11.4 deleteAssemblyInstance

Use the `deleteAssemblyInstance` command to delete an assembly instance once the assembly instance is in an undeployed state. Example 5–35 shows the `deleteAssemblyInstance` command:

Example 5–35 deleteAssemblyInstance Command

```
$ ./abctl help -command deleteAssemblyInstance
$ ./abctl deleteAssemblyInstance -assemblyInstanceId gdc4_29x5_SMALLOVA_1 -c
cloudAdmin
Assembly instance gdc4_29x5_SMALLOVA_1 has been deleted.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.12 Uploading an Assembly Archive to the Deployer Repository

You can use the `uploadAssemblyArchive` command to upload an assembly archive to Oracle Virtual Assembly Builder Deployer.

Example 5–36 uploadAssemblyArchive Command

```
$ ./abctl help -command uploadAssemblyArchive
$ ./abctl uploadAssemblyArchive -fileName Path -name String [-description String]
-connectionName String
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.13 Deleting an Assembly Archive from the Deployer Repository

The OVAB Admin can use the `deleteAssemblyArchive` command in `abctl` to delete the assembly archive from the Deployer repository. This operation may only be performed if there are no registrations for the assembly archive.

Example 5–37 deleteAssemblyArchive Command (without Waiting for Completion)

```
$ ./abctl help -command deleteAssemblyArchive
$ ./abctl deleteAssemblyArchive -name TheAssembly -version 1 -connectionName
myDeployerConnection
```

Example 5–38 deleteAssemblyArchive Command (Waiting for Completion)

```
$ ./abctl help -command deleteAssemblyArchive
$ ./abctl deleteAssemblyArchive -name TheAssembly -version 1 -connectionName
myDeployerConnection -waitForComplete -pollTime 30
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.14 Export Operations

The following sections describes the available export scenarios:

- Export and import (of the artifacts created by the export) from the Oracle Virtual Assembly Builder Studio Catalog to another Oracle Virtual Assembly Builder Studio Catalog. See the following sections:
  - Section 5.4.14.1, "Export an Appliance or Assembly from a Catalog"

  - Section 5.4.14.2, "Import an Appliance or Assembly to a Catalog"

- Assembly Archive copy. Publishing the assembly archive outside of Oracle Virtual Assembly Builder for someone else to consume. For example, you create an assembly archive and publish it within an enterprise for others to download and deploy.
  - Section 5.4.14.3, "Exporting (Copying) an Assembly Archive"
- Importing an assembly archive from the Deployer repository into the Oracle Virtual Assembly Builder Studio catalog (import command, or Import from Studio).
  - Section 5.4.14.4, "Importing an Assembly Archive to a Catalog"

5.4.14.1 Export an Appliance or Assembly from a Catalog

This section describes how to export an appliance or assembly from a catalog, using Oracle Virtual Assembly Builder Studio, or `abctl`.

To copy an appliance or assembly from one catalog to another, you must use Oracle Virtual Assembly Builder's export and import functionality.

Note: Manual copying of disk files from one catalog to another is not supported and will not work.

5.4.14.1.1 Exporting an Appliance or Assembly from a Catalog Using Oracle Virtual Assembly Builder Studio Access the Export dialog box (Figure 5–17) to export an appliance or assembly from a catalog by selecting **File** > **Export**, or by right-clicking the assembly in the Assemblies navigator and select **Export**.

Enter the following information:

- *Name*: this field pre-populates with the name of the appliance or assembly that you selected for export.
- *Directory*: browse to and select or enter the name of the directory of the location of the export. This directory must be empty and will be created if it does not exist.
- *Metadata Only*: check this checkbox to export only metadata (and not the associated templates or file sets).

Click **OK**.

Figure 5–17 Exporting an Appliance or Assembly from a Catalog

5.4.14.1.2 Exporting an Appliance or Assembly from a Catalog Using abctl Use the `export` command to export an assembly, or assembly metadata. Example 5–39 shows the `export` command for exporting metadata, and associated templates and file sets. Example 5–40 shows exporting metadata only.

Example 5–39 export Command

```
$ ./abctl help -command export
$ ./abctl export -name myOhs -toDir /tmp/myOhs.export/
(some progress messages)
Successfully exported to /tmp/myOhs.export/.
```

Example 5–40 export Command (Metadata Only)

```
$ ./abctl export -name myOhs -to /tmp/myOhs.export/ -metadataOnly
(some progress messages)
Successfully exported to /tmp/myOhs.export/.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command along with a sample output of the command.

5.4.14.2 Import an Appliance or Assembly to a Catalog

This section describes how to import an appliance, assembly, or assembly archive using Oracle Virtual Assembly Builder Studio, or `abctl`.

To copy an appliance or assembly from one catalog to another, you must use Oracle Virtual Assembly Builder's export and import functionality.

5.4.14.2.1 Importing Using Oracle Virtual Assembly Builder Studio Access the Import dialog box (Figure 5–18) to import an appliance or assembly to a Catalog by selecting **File** > **Import**. Enter the following information:

- *Path*: browse to and select or enter the name of the directory of the assembly or appliance which was exported. The path and associated assembly appear in the window.
- *Overwrite*: check this checkbox to specify that any existing metadata and associated file sets and templates are overridden. This is to correct a case of name collision.

Overriding an existing appliance can only be done if the existing appliance can be removed.

Click **OK**.

Figure 5–18 Importing an Assembly Archive

5.4.14.2.2 Importing an Appliance or Assembly Using abctl Use the `import` command to import (into the target catalog) the content of one or more files containing a sparse copy of exported metadata and associated file set and templates.

A new entry is created in the target catalog. If there is a name collision (for example, the import command attempts to create 'mySite', and the catalog already has 'mySite'), the operation will fail.

Example 5–41 shows the `import` command from a directory where you previously ran the export command:

Example 5–41 import Command from a Directory

```
$ ./abctl help -command import
$ ./abctl import -from /tmp/myOhs.export/
Successfully imported myOhs to /example/ab_home/catalog.
```

Example 5–42 shows the `import` command from an assembly archive:

Example 5–42 import Command from an Assembly Archive

```
$ ./abctl help -command import
$ ./abctl import -from /tmp/myOhs.ova
Successfully imported myOhs to /example/ab_home/catalog.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.14.2.3 Importing an External VM Template You can import an external VM Template into a specified catalog, creating an external appliance that can later be added to any assembly for deployment.

This operation can only be performed using `abctl`.

Example 5–43 shows the import external VM template command:

Example 5–43 Import an External VM Template

```
$ ./abctl importExternalTemplate -fromDir /dir/containing/image/file -name
myExternalAppliance
Executing importExternalTemplate command.
Set the root and vnc passwords that will be configured in the imported template.
Enter root password:
Retype root password:
Enter vnc password:
Retype vnc password:
  (some progress messages)
Successfully imported template.
```

5.4.14.3 Exporting (Copying) an Assembly Archive

Assembly archives in the Oracle Virtual Assembly Builder Studio catalog are stored in the `$AB_INSTANCE/ova` directory. You can copy or use FTP to transfer the assembly archives from this directory without the need for any export utility. Typically you will copy the assembly archives to the *Deployments* navigator managed by Deployer.

5.4.14.4 Importing an Assembly Archive to a Catalog

Importing an assembly archive allows you to import an assembly archive template file from a disk location into the local catalog to allow editing of the assembly appliances. Importing creates a metadata structure, with file set and template artifacts that are identical to the original catalog.

To import an assembly archive use the `downloadAssemblyArchive` command in `abctl`:

Example 5–44 downloadAssemblyArchive Command

```
$ ./abctl help -command downloadAssemblyArchive
$ ./abctl downloadAssemblyArchive -name MyAssembly -version 1 -fileName
RenamedAssembly.ova
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.15 Interacting with EM Software Library

Use these operations if you plan to use Enterprise Manager Grid Control Cloud Management Pack to perform deployment operations on assembly archives. In this scenario, you use Oracle Virtual Assembly Builder to create your assembly archive and upload it to the Enterprise Manager Software Library. You use Enterprise Manager Grid Control Cloud Management Pack to deploy the assembly instance.

5.4.15.1 Configuring a Connection to Enterprise Manager Software Library

You can configure a connection to the Enterprise Manager Software Library using the `createEMConnection` command in `abctl`. The connection to Enterprise Manager Software Library is persisted in a connections file.

You must specify the fully qualified hostname of the remote Enterprise Manager machine, for example `myhost.example.com` instead of `myhost`.

Example 5–45 createEMConnection

```
$ ./abctl createEmConnection -connectionURL emMachine:7791 -connectionUser admin
-namedHostCredential hostCredential -remoteUser mySshUser -remoteWorkingDir
/scratch/myovas [-sshPort 23] [-privateKeyFile ~/.ssh/id_rsa]
```

You are prompted for a `connectionPassword`.

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.15.2 Uploading an Assembly

You can upload an assembly archive from Oracle Virtual Assembly Builder Studio Catalog to Enterprise Manager Software Library using the `uploadEMAssemblyArchive` command in `abctl`. The assembly archive must have been created with the assembly.

Example 5–46 Upload an Assembly to Enterprise Manager Software Library

```
$ ./abctl uploadEMAssemblyArchive -name archiveName -description "my assembly
archive"
 Assembly archive upload started
  Assembly archive upload at 10%
  Assembly archive upload at 20%
  Assembly archive upload at 30%
  Assembly archive upload at 40%
  Assembly archive upload at 50%
  Assembly archive upload at 60%
  Assembly archive upload at 70%
  Assembly archive upload at 80%
  Assembly archive upload at 90%
  Assembly archive upload at 100%
  Assembly archive upload complete
  Assembly archive version 0.1 uploaded
  Successfully uploaded the assembly archive mySite to EM Software Library.  Check
the status of the assembly archive with describeEMAssemblyArchives before using.
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.15.3 Describe Assembly Archives in Enterprise Manager Software Library

You can list the assembly archives and their versions present in the Enterprise Manager Software Library using the `describeEMAssemblyArchives` command in `abctl`:

Example 5–47 Describing the Assembly Archives in EM Software Library

```
$ ./abctl describeEMAssemblyArchives [-name nameOfAssemblyArchive]

-----------------------------------------------
Name   | Version | Description | Status
-----------------------------------------------
mySite | 0.3     | mysite3     | READY
       | 0.2     | mysite3     | READY
       | 0.1     | mysite3     | READY
-----------------------------------------------
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.15.4 Deleting an Assembly Archive from Enterprise Manager Software Library

You can delete the specified version of an assembly archive from Enterprise Manager Software Library using the `deleteEMAssemblyArchive` command in `abctl`.

> **Note:** Do not attempt to delete an assembly archive until its status is READY from the `describeEMAssemblyArchives` command. See Example 5–47, "Describing the Assembly Archives in EM Software Library" for example output.

Example 5–48 Delete an Assembly Archive from Enterprise Manager Software Library

```
abctl deleteEMAssemblyArchive -name archiveName -version 1.2
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

5.4.15.5 Downloading an Assembly to Oracle Virtual Assembly Builder Catalog

You can download an Assembly from the Enterprise Manager Software Library to the Oracle Virtual Assembly Builder Studio Catalog using the `download` command in `abctl`.

> **Note:** Do not attempt to download an assembly archive until its status is READY from the `describeEMAssemblyArchives` command. See Example 5–47, "Describing the Assembly Archives in EM Software Library" for example output.

The assembly archive is reverse engineered to have the Oracle Virtual Assembly Builder metadata, file sets and templates created and persisted in the Oracle Virtual Assembly Builder Studio catalog.

Example 5–49 Download an Assembly

```
abctl downloadEMAssemblyArchive -name archiveName -version 1.0 -force -downloadAs
newName
```

For more information see Appendix A, "Command Line Reference", which contains the details of the parameters that can be passed into the command.

A

Command Line Reference

This appendix contains information about the `abctl` commands included in Oracle Virtual Assembly Builder.

A.1 Commands

This section describes the available commands. Commands fall into two categories:

- Commands for creating appliances and assemblies, creating appliance templates and assembly archives, and creating deployment plans.
- Commands for configuring deployment targets, uploading assembly archives to Deployer, creating assembly instances, and deploying, undeploying, starting, and stopping assembly instances and scaling appliance instances. These commands interface with the Oracle Virtual Assembly Builder Deployer Web service.

When Oracle Virtual Assembly Builder Studio is installed, all commands are available. When only Oracle Virtual Assembly Builder Deployer is installed, a subset of the commands are available. Table A–1 indicates which commands are available depending on which Oracle Virtual Assembly Builder components have been installed.

Table A–1 Available Commands by Installation Type

Command Name and Section	Command Availability for Installation Type		
	Installation of Deployer only	Installation of Studio only	Installation of Studio and Deployer
Section A.1.1, "addAssemblyUsers"	Yes	Yes	Yes
Section A.1.2, "addTargetUser"	Yes	Yes	Yes
Section A.1.3, "addToAssembly"	No	Yes	Yes
Section A.1.4, "captureFileSets"	No	Yes	Yes
Section A.1.5, "connectEndpoints"	No	Yes	Yes
Section A.1.6, "createAssembly"	No	Yes	Yes
Section A.1.7, "createAssemblyArchive"	Yes	Yes	Yes
Section A.1.8, "createAssemblyInstance"	Yes	Yes	Yes
Section A.1.9, "createDeployerConnection"	Yes	Yes	Yes

Table A–1 (Cont.) Available Commands by Installation Type

Command Name and Section	Command Availability for Installation Type		
	Installation of Deployer only	Installation of Studio only	Installation of Studio and Deployer
Section A.1.10, "createEMConnection"	No	Yes	Yes
Section A.1.11, "createExternalResources"	No	Yes	Yes
Section A.1.12, "createTags"	Yes	Yes	Yes
Section A.1.13, "createTarget"	Yes	Yes	Yes
Section A.1.14, "createTemplate"	No	Yes	Yes
Section A.1.15, "delete"	No	Yes	Yes
Section A.1.16, "deleteAssemblyArchive"	Yes	Yes	Yes
Section A.1.17, "deleteAssemblyInstance"	Yes	Yes	Yes
Section A.1.18, "deleteDeployerConnection"	Yes	Yes	Yes
Section A.1.19, "deleteEMConnection"	No	Yes	Yes
Section A.1.20, "deleteEMAssemblyArchive"	No	Yes	Yes
Section A.1.21, "deleteFailedAssemblyInstances"	Yes	Yes	Yes
Section A.1.22, "deleteRequests"	Yes	Yes	Yes
Section A.1.23, "deleteTags"	Yes	Yes	Yes
Section A.1.24, "deleteTarget"	Yes	Yes	Yes
Section A.1.25, "deployAssemblyInstance"	Yes	Yes	Yes
Section A.1.26, "describeApplianceInstances"	Yes	Yes	Yes
Section A.1.27, "describeAssemblyArchives"	Yes	Yes	Yes
Section A.1.28, "describeAssemblyInstances"	Yes	Yes	Yes
Section A.1.29, "describeAssemblyUsers"	Yes	Yes	Yes
Section A.1.30, "describeCatalog"	No	Yes	Yes
Section A.1.31, "describeDeployer"	Yes	Yes	Yes
Section A.1.32, "describeDeployerConnections"	No	Yes	Yes
Section A.1.33, "describeDeploymentPlans"	No	Yes	Yes
Section A.1.34, "describeEMAssemblyArchives"	No	Yes	Yes
Section A.1.35, "describeEMConnection"	No	Yes	Yes
Section A.1.36, "describeEndpoints"	No	Yes	Yes
Section A.1.37, "describeRegistrations"	Yes	Yes	Yes
Section A.1.38, "describeRequests"	Yes	Yes	Yes
Section A.1.39, "describeScalingGroups"	Yes	Yes	Yes
Section A.1.40, "describeTags"	Yes	Yes	Yes
Section A.1.41, "describeTargetConfigurations"	Yes	Yes	Yes
Section A.1.42, "describeTargetNames"	Yes	Yes	Yes
Section A.1.43, "describeTargetUsers"	Yes	Yes	Yes

Table A–1 (Cont.) Available Commands by Installation Type

Command Name and Section	Command Availability for Installation Type		
	Installation of Deployer only	Installation of Studio only	Installation of Studio and Deployer
Section A.1.44, "describeTargets"	Yes	Yes	Yes
Section A.1.45, "describeUserTargets"	Yes	Yes	Yes
Section A.1.46, "describeVnets"	Yes	Yes	Yes
Section A.1.47, "downloadAssemblyArchive"	Yes	Yes	Yes
Section A.1.48, "downloadAssemblyMetadata"	Yes	Yes	Yes
Section A.1.49, "downloadEMAssemblyArchive"	No	Yes	Yes
Section A.1.50, "export"	No	Yes	Yes
Section A.1.51, "getDefaultTarget"	Yes	Yes	Yes
Section A.1.52, "getTargetType"	Yes	Yes	Yes
Section A.1.53, "help"	Yes	Yes	Yes
Section A.1.54, "import"	No	Yes	Yes
Section A.1.55, "importExternalTemplate"	No	Yes	Yes
Section A.1.56, "introspectCoherenceWeb"	No	Yes	Yes
Section A.1.57, "introspectForms"	No	Yes	Yes
Section A.1.58, "introspectGenericProd"	No	Yes	Yes
Section A.1.59, "introspectOHS"	No	Yes	Yes
Section A.1.60, "introspectOTD"	No	Yes	Yes
Section A.1.61, "introspectRACDB"	No	Yes	Yes
Section A.1.62, "introspectReports"	No	Yes	Yes
Section A.1.63, "introspectSIDB"	No	Yes	Yes
Section A.1.64, "introspectSOA"	No	Yes	Yes
Section A.1.65, "introspectTuxedo"	No	Yes	Yes
Section A.1.66, "introspectWebCache"	No	Yes	Yes
Section A.1.67, "introspectWLS"	No	Yes	Yes
Section A.1.68, "redeployAssemblyInstance"	No	Yes	Yes
Section A.1.69, "registerAssemblyArchive"	Yes	Yes	Yes
Section A.1.70, "removeAssemblyUsers"	No	Yes	Yes
Section A.1.71, "removeTargetUsers"	Yes	Yes	Yes
Section A.1.72, "restartAssemblyInstance"	Yes	Yes	Yes
Section A.1.73, "scale"	Yes	Yes	Yes
Section A.1.74, "setDefaultTarget"	Yes	Yes	Yes
Section A.1.75, "startAssemblyInstance"	Yes	Yes	Yes
Section A.1.76, "stopAssemblyInstance"	Yes	Yes	Yes
Section A.1.77, "undeployAssemblyInstance"	Yes	Yes	Yes

Table A–1 (Cont.) Available Commands by Installation Type

Command Name and Section	Command Availability for Installation Type		
	Installation of Deployer only	Installation of Studio only	Installation of Studio and Deployer
Section A.1.78, "unregisterAssemblyArchive"	Yes	Yes	Yes
Section A.1.79, "updateAssemblyArchive"	Yes	Yes	Yes
Section A.1.80, "updateTarget"	Yes	Yes	Yes
Section A.1.81, "uploadAssemblyArchive"	Yes	Yes	Yes
Section A.1.82, "uploadAssemblyResources"	Yes	Yes	Yes
Section A.1.83, "uploadEMAssemblyArchive"	No	Yes	Yes
Section A.1.84, "version"	Yes	Yes	Yes

A.1.1 addAssemblyUsers

Details for this command follow.

A.1.1.1 Synopsis

```
$ abctl addAssemblyUsers -assembly String -user String... -connectionName String
```

A.1.1.2 Description

Adds one or more users to an assembly.

A.1.1.3 Options

Table A–2 shows the command options for `addAssemblyUsers`.

Table A–2 addAssemblyUsers options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assembly`	a	true	none	A string representing the name of the assembly.	Specifies the assembly to which to add users
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-user`	u	true	none	A string representing the username(s) to add.	Specifies one or more users to add to an assembly.

A.1.1.4 Examples

Here are some command examples.

A.1.1.4.1 Adding Users to an Assembly

```
$ abctl addAssemblyUsers -assembly MyAssembly -user User1 User2
```

A.1.2 addTargetUser

Details for this command follow.

A.1.2.1 Synopsis

```
$ abctl addTargetUser -user String -target String [-properties String...]
```

```
-connectionName String
```

A.1.2.2 Description

Adds a user to a target.

A.1.2.3 Options

Table A–3 shows the command options for `addTargetUser`.

Table A–3 addTargetUser options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-user`	u	true	none	A string representing the username of the user.	The username of the user.
`-target`	t	true	none	A string representing the target to which the user is being added.	The target to which a user is being added.
`-properties`	p	false	none	A string representing property=value pairs to apply to the user.	The properties to apply to the user.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.2.4 Examples

Here are some command examples.

A.1.2.4.1 Adding Several Users to an Assembly

```
$ abctl addTargetUser -user Username -target Targetname -connectionName
MyDeployerConnection
```

A.1.3 addToAssembly

Details for this command follow.

A.1.3.1 Synopsis

```
$ abctl addToAssembly -name String -into String
```

A.1.3.2 Description

Adds an existing appliance or assembly to another existing assembly.

A.1.3.3 Options

Table A–4 shows the command options for `addToAssembly`.

Table A–4 addToAssembly options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	The name of an existing appliance or atomic assembly.	The name of an existing appliance or atomic assembly to add.
`-into`	i	true	none	The name of a non-atomic assembly.	The name of an existing assembly to populate.

A.1.3.4 Examples

Here are some command examples.

A.1.3.4.1 Adding an Appliance into an Existing Assembly

```
$ abctl addToAssembly -name myAppliance -into myAssembly
```

A.1.4 captureFileSets

Details for this command follow.

A.1.4.1 Synopsis

```
$ abctl captureFileSets -name String [-remoteHost String] [-remoteUser String]
[-sudoUser String] [-remoteWorkingDir Path] [-remoteCleanup]
[-privateKeyFile Path] [-quiet] [-force]
```

A.1.4.2 Description

Creates file sets for the specified appliance or assembly.

A.1.4.3 Options

Table A–6 shows the command options for `captureFileSets`.

Table A–5 captureFileSets options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-remoteHost`	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, the `remoteUser` must be specified as well.
`-remoteUser`	ru	false	none	N/A	Name of the ssh user to use for accessing the remote machine. If set, the `remoteHost` must be specified as well.
`-sudoUser`	su	false	none	User name of sudo user.	Name of the user on the remote machine to sudo as before executing operations. Note that sudoUser is equivalent to *Run as user* in Oracle Virtual Assembly Builder Studio. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.
`-remoteWorkingDir`	rwd	false	`/tmp/abRemote_<remote user name>`	N/A	Path on the remote machine to work out of. If set, the `remoteUser` and `remoteHost` must be specified as well.
`-remoteCleanup`	rc	false	none	N/A	Remote clean up flag. When set, the remote working directory is deleted after the operation. Otherwise the directory is not be modified. If set, the `remoteUser` and `remoteHost` must be specified as well.
`-privateKeyFile`	pkf	false	`~/.ssh/id_rsa`	Location of a private key file.	Private SSH key file on the local machine.

Table A–5 (Cont.) captureFileSets options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-quiet`	q	false	none	N/A	By default, the command shows detailed progress/success messages. If `-quiet` is set, the command turns off verbose mode and shows only one or two progress/success messages.
`-name`	n	false	Derived directory name prefixed by component type name.	Name of an appliance or assembly. Nested appliances or assemblies are referred to with slash ('/'), for example: mySite/myOhs	Specify the name of an appliance or assembly to be captured. For an assembly, only an atomic assembly name can be specified. To capture a non-atomic assembly, its sub-appliances and sub-assemblies must be captured individually.
`-force`	f	false	none	N/A	If specified, existing file sets and any appliance templates created from it will be overwritten. The operation can fail if there is an existing registered appliance template that was created from an existing file set. The flag has no effect if no file set or template exists.

A.1.4.4 Examples

Here are some command examples.

A.1.4.4.1 Capture File Sets

```
% abctl captureFileSets -name myOhs -force
```

A.1.5 connectEndpoints

Details for this command follow.

A.1.5.1 Synopsis

```
$ abctl connectEndpoints -from String -fromOutput String -to String -toInput
String
```

A.1.5.2 Description

Creates a new connection between an output and an input. The protocols of the output and input must match, and the owners of the output and input must be part of the same assembly.

A.1.5.3 Options

Table A–6 shows the command options for `connectEndpoints`.

Table A–6 connectEndpoints options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-from`	f	true	none	Name of appliance or assembly.	The name of an appliance or assembly to connect from.
`-fromOutput`	fout	true	none	Appliance output or assembly output.	Appliance output or assembly output to connect from.
`-to`	t	true	none	Name of appliance or assembly.	The name of an appliance or assembly to connect to.
`-toInput`	tin	true	none	Appliance input or assembly input.	Appliance input or assembly input to connect to.

A.1.5.4 Examples

Here are some command examples.

A.1.5.4.1 connectEndpoints

```
% abctl connectEndpoints -from mySite/myOhs -fromOutput output1 -to mySite/myWls
-toInput default
```

A.1.6 createAssembly

Details for this command follow.

A.1.6.1 Synopsis

```
$ abctl createAssembly -name String [-defaultNetwork String] [-force]
```

A.1.6.2 Description

Creates a new assembly in the catalog if one does not already exist by the specified name.

A.1.6.3 Options

Table A–7 shows the command options for `createAssembly`.

Table A–7 createAssembly options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	Name of the new assembly.	Name of a new assembly to be created.
`-defaultNetwork`	dn	false	none	Name of the default vNet.	If specified, specifies the name to use for the assembly's default vNet.
`-force`	f	false	none	N/A	If `-force` is set, the existing appliance or assembly in the catalog using the same as the newly-created assembly will be overridden.

A.1.6.4 Examples

Here are some command examples.

A.1.6.4.1 Creating an Assembly

```
$ abctl createAssembly -name myAssembly -defaultNetwork intranet
```

A.1.7 createAssemblyArchive

Details for this command follow.

A.1.7.1 Synopsis

```
$ abctl createAssemblyArchive -name String -platform String [-noCompress] [-force]
[-quiet]
```

A.1.7.2 Description

Creates an assembly archive for the named top-level assembly. This command can only be invoked on a top-level assembly. Additionally, all the sub-appliances within the assembly must previously have been templated using the `createTemplate` command.

A.1.7.3 Options

Table A–9 shows the command options for `createAssemblyArchive`.

Table A–8 createAssemblyArchive options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	Name of an appliance or atomic assembly. Nested appliances or assemblies are referred to with slash ('/'), for example: mySite/myOhs.	Name of an appliance or assembly in the catalog. Only the top-level appliance or assembly is allowed.
`-platform`	p	true	none	A string representing any valid platform for which templates have been created for the named assembly.	Target platform for which the assembly archive is built.
`-force`	f	false	false	N/A	If specified, any existing assembly archive will be overridden. If no archive exists, this flag has no effect.
`-quiet`	q	false	none	N/A	By default, the command shows detailed progress/success messages. If `-quiet` is set, the command turns off verbose mode and shows only one or two progress/success messages.

A.1.7.4 Examples

Here are some command examples.

A.1.7.4.1 Creating an Assembly Instance

```
$ abctl createAssemblyArchive -name myWlsAssembly -platform OVM -nocompress
```

A.1.8 createAssemblyInstance

Details for this command follow.

A.1.8.1 Synopsis

```
$ abctl createAssemblyInstance -deploymentPlan Path -name String -version String
```

```
[-target String] -connectionName String
```

A.1.8.2 Description

Creates an assembly instance for an assembly.

A.1.8.3 Options

Table A–9 shows the command options for `createAssemblyInstance`.

Table A–9 createAssemblyInstance options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-deployment Path`	dp	true	none	A file path to the assembly deployment plan on disk.	Specifies a path to a deployment plan file to use for the assembly instance.
`-name`	n	true	none	A string representing the name of the assembly.	The name of the assembly.
`-version`	v	true	none	A string representing the version of the assembly.	Assembly version.
`-target`	t	false	none	A string representing the name of the target.	The name of the target.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer Web Service.	The name of a connection to the Deployer Web Service.

A.1.8.4 Examples

Here are some command examples.

A.1.8.4.1 Creating an Assembly Instance

```
$ abctl createAssemblyInstance -deploymentPlan c:/MyDeploymentPlan.xml -name
MyAssembly -version 1
```

A.1.9 createDeployerConnection

Details for this command follow.

A.1.9.1 Synopsis

```
$ abctl createDeployerConnection -name String -url String [-username String]
[-noReviewCert]
```

A.1.9.2 Description

Creates a new connection between `abctl` and the Deployer. If you specify a connection using the HTTPS protocol, you may be prompted to approve a certificate.

A.1.9.3 Options

Table A–10 shows the command options for `createDeployerConnection`.

Table A–10 createDeployerConnection options

Name	Alias	Req'd	Default Values	Possible Values	Description
-name	n	true	none	A string representing the name of the Deployer connection.	The name of the Deployer connection.
-url	u	true	none	A string representing the URL of the Deployer Web service.	The URL of the Deployer.
-username	un	false	none	A string representing the username.	The username to use to authenticate with the Deployer Web service.
-noReviewCert	nr	false	none	N/A	If set, do not present the HTTPS connection certificate for approval.

A.1.9.4 Examples

Here are some command examples.

A.1.9.4.1 Creating a Deployer Connection

```
$ abctl createDeployerConnection -name WLS1 -url http://localhost:7001
```

A.1.10 createEMConnection

Details for this command follow.

A.1.10.1 Synopsis

```
$ abctl createEmConnection -connectionURL emMachine:port -connectionUser admin
-namedHostCredential hostCredential -remoteUser mySshUser -remoteWorkingDir
myRemoteWorkingDir [-sshPort port] [-privateKeyFile ~/.ssh/id_rsa]
```

A.1.10.2 Description

Creates a connection to an Enterprise Manager Software Library, and persists the connection in a connections file.

You must specify the fully qualified hostname of the remote Enterprise Manager machine, for example `myhost.example.com` instead of `myhost`.

When you perform this command, you are prompted for a connection password.

A.1.10.3 Options

Table A–11 shows the command options for `createEMConnection`.

Table A–11 createEMConnection options

Name	Alias	Req'd	Default Values	Possible Values	Description
-connectionURL	c	true	none	Hostname:port.	URL for connecting to EM Software Library.
-connectionUser	cu	true	none	Valid EM Software Library User.	User for EM Software Library.
-namedHostCredential	n	true	none	Valid Named Host Credential.	Named Host Credential.
-remoteUser	ru	true	none	Valid SSH user.	SSH user for connecting to the machine where the EM Software Library is located.

Table A–11 (Cont.) createEMConnection options

Name	Alias	Req'd	Default Values	Possible Values	Description
-sshPort	s	false	none	Valid SSH port number.	SSH port for EM Software Library machine.
-privateKeyFile	pkf	false	none	~/.ssh/id_rsa, id_rsa	Local private SSH key file used for SSH to the remote EM Software Library machine.
-remoteWorkingDir	rwd	true	none	/scratch,/home/mydir	Valid directory on EM Software Library machine, where assembly archives are uploaded and consumed. Oracle Corporation recommends that you do not use the /tmp directory.

A.1.10.4 Examples

Here are some command examples.

A.1.10.4.1 createEMConnection

```
$ abctl createEmConnection -connectionURL emMachine:7791 -connectionUser admin
-namedHostCredential hostCredential -remoteUser mySshUser -remoteWorkingDir
/scratch/myovas [-sshPort 23] [-privateKeyFile ~/.ssh/id_rsa]
```

A.1.11 createExternalResources

Details for this command follow.

A.1.11.1 Synopsis

```
$ abctl createExternalResources -from String [-fromOutput String] [-name String]
[-recurse]
```

A.1.11.2 Description

Creates external resources for each of an appliance's or assembly's unconnected outputs.

A.1.11.3 Options

Table A–12 shows the command options for createExternalResources.

Table A–12 createExternalResources options

Name	Alias	Req'd	Default Values	Possible Values	Description
-from	f	true	none	Appliance or assembly name.	Name of the appliance or assembly for which you want to external resources.
-fromOutput	fout	false	none	Appliance output or assembly output name.	Name of the appliance or assembly output for which you want to create an external resource.
-name	n	false	none	New external resource name.	Name of the new external resource appliance. This parameter is only applicable when creating an external resource for an individual output using the -fromOutput parameter.
-recurse	r	false	none	N/A	If specified, create external resources for each of an assembly's sub-elements.

A.1.11.4 Examples

Here are some command examples.

A.1.11.4.1 Create External Resources for an Assembly

```
% abctl createExternalResources -from mySite/myWls
```

A.1.11.4.2 Create External Resources for Each of an Assembly's Sub-elements

```
% abctl createExternalResources -from mySite -r
```

A.1.11.4.3 Create an External Resource for the jdbc0 Output

```
% abctl createExternalresources -from mySite/myWls -fromOutput jdbc0 -name my_Ext_
JDBC
```

A.1.12 createTags

Details for this command follow.

A.1.12.1 Synopsis

```
$ abctl createTags -tag String... -resource String... -connectionName String
```

A.1.12.2 Description

Creates one or more tags for a resource.

A.1.12.3 Options

Table A–13 shows the command options for `createTags`.

Table A–13 createTags options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-tag`	t	true	none	A set of name=value pairs specifying the tags.	Specifies one or more tags to tag a resource with.
`-resource`	r	true	none	A string specifying the resource id of the object to tag.	Specifies one or more resources to apply a tag to.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer Web Service.	The name of a connection to the Deployer Web Service.

A.1.12.4 Examples

Here are some command examples.

A.1.12.4.1 Create Tags

```
$ abctl createTags -tag key=value -resource MyResource
```

A.1.13 createTarget

Details for this command follow.

A.1.13.1 Synopsis

```
$ abctl createTarget -name String -type String -connectionName String [-properties
String...] [-default]
```

A.1.13.2 Description

Creates a deployment target. This command is enabled for Oracle VM targets, but disabled for Oracle Exalogic. The single target in an Oracle Exalogic installation is preconfigured and cannot be changed.

The following are required and optional properties for the `createTarget` command. (The asterisks (*) indicate a required property):

- ovm [ovm.url*, ovm.poolName*, ovm.user*, ovm.pwd*, ovm.vmmversion*, ovm.vmOperationTimeout]

A.1.13.3 Oracle VM Configuration

Oracle recommends that you configure your target connections for Oracle VM 3 with TCP instead of HTTP protocol.

To configure with TCP, specify a URL of the form "tcp://their-ovm-host:54321".

A.1.13.4 Options

Table A–14 shows the command options for `createTarget`.

Table A–14 createTarget options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	A string representing the name of the target.	The name of the target.
`-type`	t	true	none	A string representing the type of target. Possible value is `ovm`.	The type of assembly instance target.
`-properties`	p	false	none	A string representing property=value pairs to set on the target.	The properties to set on the target.
`-default`	d	false	false	N/A	If set, indicates that this target is the default target.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.13.5 Examples

Here are some command examples.

A.1.13.5.1 Create Target

```
$ abctl createTarget -name MyTarget -type OVM
```

A.1.14 createTemplate

Details for this command follow.

A.1.14.1 Synopsis

```
$ abctl createTemplate -name String -platform String [-quiet] [-baseImage Path]
[-force]
```

A.1.14.2 Description

Creates an appliance template for a given appliance or assembly.

A.1.14.3 Options

Table A–15 shows the command options for `createTemplate`.

Table A–15 createTemplate options

Name	Alias	Req'd	Default Values	Possible Values	Description
-baseImage	bi	false	If not specified, the command attempts to locate the base image from $<AB_INSTANCE>/templates/baseImage/OVM/OEL. If not found there, it attempts to locate it from $<ORACLE_HOME>/templates/baseImage/OVM/OEL.	Path to a valid base image.	Path to a valid base image used to create an appliance template.
-force	f	false	false	N/A	If -force is set, the existing template(s) for assemblies or appliances will be overridden. If the template does not exist, this flag has no effect.
-name	n	true	none	Name of appliance or assembly in catalog. Nested appliances or assemblies are referred to with a slash ("/"). For example: mySite/myOhs.	Name of an appliance or assembly in the catalog.
-platform	p	true	none	OVM, Exalogic	Target platform for which the appliance template is built.
-quiet	q	false	false	N/A	By default, the command shows detailed progress/success messages. If -quiet is set, the command turns off verbose mode and shows only one or two progress/success messages.

A.1.14.4 Examples

Examples for this command follow.

A.1.14.4.1 No valid base image is found

```
$ abctl createTemplate -name myOhs -platform OVM
Executing createTemplate command.
Error: OAB-7389: Failed to create VM template for myOhs.
Caused by: OAB-20343: Unable to locate a valid default base image.
  Action: Specify a base image location, or place a base image in default
location. Refer to user guide for detail.
```

A.1.14.4.2 Template already exists for given OS type

```
$ abctl createTemplate -name myOhs -platform OVM
Executing createTemplate command.
Error: OAB-7389: Failed to create VM template for myOhs.
Caused by: OAB-20120: Appliance myOhs already has template for OEL.
  Action: Use -force flag to override existing template.
```

A.1.14.4.3 Successful Template Creation

```
$ abctl createTemplate -name myOhs -platform OVM -baseImage
/private/baseImage/OVM/OEL/System.img
Executing createTemplate command.
Set the root and vnc passwords that will be configured in the template.
Enter root password:
Retype root password:
Enter vnc password:
Retype vnc password:
```

```
  Step 1 of 2: Creating template for appliance myOhs started.
    Step 1 of 6: Copying base image to catalog started.
    Step 2 of 6: Copying base image to catalog completed.
    Step 3 of 6: Creating AB image started.
    Step 4 of 6: Creating AB image completed.
    Step 5 of 6: Creating product disk for myOhs_root started.
    Step 6 of 6: Creating product disk for myOhs_root completed.
  Step 2 of 2: Creating template for appliance myOhs completed.
Successfully created template for myOhs.
```

A.1.15 delete

Details for this command follow.

A.1.15.1 Synopsis

```
$ abctl delete [-name] String [-archiveOnly]
```

A.1.15.2 Description

Deletes the appliance or assembly with the given name. Only the top-level appliance or assembly can be deleted. Nested appliances or assemblies cannot be deleted using this command. Also, registered appliances or assemblies cannot be deleted.

A.1.15.3 Options

Table A–16 shows the command options for `delete`.

Table A–16 delete options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	Name of the top-level appliance or assembly.	Name of the top-level appliance or assembly to be deleted.
`-archiveOnly`	o	false	none	N/A	If specified, delete only the assembly archive, leaving the rest of the assembly intact.

A.1.15.4 Examples

Here are some command examples.

A.1.15.4.1 Attempted to delete nested appliance/assembly

```
$ abctl delete -name mySite/myOhs
Executing delete command.
Error: OAB-7672: Unable to delete mySite/myOhs from catalog.
  Cause: Nested appliance or assembly cannot be deleted.
  Action: Use AbStudio (GUI) to delete nested appliances or assemblies.
```

A.1.15.4.2 Successful Delete

```
$ abctl delete -name myOhs
Executing delete command.
Successfully deleted myOhs.
```

A.1.15.4.3 Delete of Only an Archive

```
$ abctl delete -name myOhs -archiveOnly
```

A.1.15.4.4 Delete Failed

```
$ abctl delete -name myOhs
Deleted metadata
Deleted File Sets
Error: Failed to delete templates.
```

A.1.16 deleteAssemblyArchive

Details for this command follow.

A.1.16.1 Synopsis

```
$ abctl deleteAssemblyArchive -name String [-version String] -connectionName
String
```

A.1.16.2 Description

This command deletes an assembly from the Deployer. This operation may only be performed if there are no registrations for the assembly archive.

A.1.16.3 Options

Table A–17 shows the command options for `deleteAssemblyArchive`.

Table A–17 deleteAssemblyArchive options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	A string representing the name of the assembly.	Name of an assembly in the Deployer.
`-version`	v	false	The default is the latest version number assigned by the Deployer.	A string representing the version of the assembly.	Specifies the version of the assembly to delete from the Deployer.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.16.4 Examples

Here are some command examples.

A.1.16.4.1 Deleting an Assembly

```
$ abctl deleteAssemblyArchive -name TheAssemblyArchive -version 1
```

A.1.17 deleteAssemblyInstance

Details for this command follow.

A.1.17.1 Synopsis

```
$ abctl deleteAssemblyInstance -assemblyInstanceId String -connectionName String
```

A.1.17.2 Description

Deletes an assembly instance.

This operation can only be executed when the assembly instance is in an undeployed state.

A.1.17.3 Options

Table A–18 shows the command options for `deleteAssemblyInstance`.

Table A–18 deleteAssemblyInstance options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assemblyInstanceId`	d	true	none	A string representing the `assemblyInstanceId`.	The identifier of an assembly instance to be deleted.
`-connectionName`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.17.4 Examples

Here are some command examples.

A.1.17.4.1 Deleting an Assembly Instance

```
$ abctl deleteAssemblyInstance -assemblyInstanceId MyId
```

A.1.18 deleteDeployerConnection

Details for this command follow.

A.1.18.1 Synopsis

```
$ abctl deleteDeployerConnection [-name] String
```

A.1.18.2 Description

Deletes a connection to the Deployer, and removes the connection from the connections file.

A.1.18.3 Options

Table A–19 shows the command options for `deleteDeployerConnection`.

Table A–19 deleteDeployerConnection options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	A string representing the name of the Deployer connection.	The name of the Deployer connection.

A.1.18.4 Examples

Here are some command examples.

A.1.18.4.1 Deleting a Connection to the Deployer

```
$ abctl deleteDeployerConnection -name WLS1
```

A.1.19 deleteEMConnection

Details for this command follow.

A.1.19.1 Synopsis

```
$ abctl deleteEMConnection
```

A.1.19.2 Description

Deletes a connection to the EM Software Library.

A.1.19.3 Options

None.

A.1.19.4 Examples

Here are some command examples.

A.1.19.4.1 Deleting an EM Connection

```
$ abctl deleteEMConnection
```

A.1.20 deleteEMAssemblyArchive

Details for this command follow.

A.1.20.1 Synopsis

```
$ abctl deleteEMAssemblyArchive -name String -version String
```

A.1.20.2 Description

Deletes the specified version of an assembly archive from the Enterprise Manager Software Library.

A.1.20.3 Options

Table A–20 shows the command options for `deleteEMAssemblyArchive`.

Table A–20 deleteEMAssemblyArchive options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	Name of assembly archive	Name of assembly archive to delete from the EM Software Library.
`-version`	v	true	none	1.0, 1.2, 2.0, etc.	Version of an assembly archive to delete from the EM Software Library.

A.1.20.4 Examples

Here is a command example.

A.1.20.4.1 Delete EM Assembly Archive

```
% abctl deleteEMAssemblyArchive -name archiveName -version 1.2
```

A.1.21 deleteFailedAssemblyInstances

Details for this command follow.

A.1.21.1 Synopsis

```
$ abctl deleteFailedApplianceInstances -applianceId String -applianceInstanceIds
```

```
String... -connectionName String
```

A.1.21.2 Description

Deletes one or more appliances instances in the failed state.

A.1.21.3 Options

Table A–21 shows the command options for `deleteFailedAssemblyInstances`.

Table A–21 deleteFailedAssemblyInstances options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-applianceId`	a	true	none	A string representing the ID of the appliance.	The ID of the appliance.
`-applianceInstanceIds`	ai	true	none	A string representing the IDs of the appliances.	The IDs of the appliance instances in a failed state to delete.
`-connectionName`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.21.4 Examples

Here is a command example.

A.1.21.4.1 Delete Failed Assembly Instances

```
% abctl deleteFailedApplianceInstances -applianceId MyId -applianceInstanceIds
MyInstanceId1
```

A.1.22 deleteRequests

Details for this command follow.

A.1.22.1 Synopsis

```
$ abctl deleteRequests [-requestId String...] -connectionName String
```

A.1.22.2 Description

Deletes one or more previously completed requests.

A.1.22.3 Options

Table A–67 shows the command options for `deleteRequests`.

Table A–22 deleteRequests options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connectionName`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-requestId`	d	false	none	A string representing the `requestId`.	The `requestId` of a previously completed request.

A.1.22.4 Examples

Here are some command examples.

A.1.22.4.1 Delete Requests

```
$ abctl deleteRequests
```

A.1.23 deleteTags

Details for this command follow.

A.1.23.1 Synopsis

```
$ abctl deleteTags -tag String... -resource String... -connectionName String
```

A.1.23.2 Description

This command deletes one or more tags for a resource.

A.1.23.3 Options

Table A–23 shows the command options for `deleteTags`.

Table A–23 deleteTags options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-resource`	r	true	none	A string specifying the resource id of the object to remove the tag from.	Specifies one or more resources to remove a tag from.
`-tag`	t	true	none	A set of name=value pairs specifying the tags.	Specifies one or more tags to remove from a resource.

A.1.23.4 Examples

Here are some command examples.

A.1.23.4.1 Deleting Tags

```
$ abctl deleteTags -tag foo -resource MyResource
```

A.1.24 deleteTarget

Details for this command follow.

A.1.24.1 Synopsis

```
$ abctl deleteTarget -name String -connectionName String
```

A.1.24.2 Description

This command deletes a target and all configuration information. If this target was a default for a user or all users, then that default is unset.

A.1.24.3 Options

Table A–24 shows the command options for `deleteTarget`.

Table A–24 deleteTarget options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	A string representing the name of the target.	The name of the target.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.24.4 Examples

Here are some command examples.

A.1.24.4.1 Deleting a Target

```
$ abctl deleteTarget -name MyTarget
```

A.1.25 deployAssemblyInstance

Details for this command follow.

A.1.25.1 Synopsis

```
$ abctl deployAssemblyInstance -assemblyInstanceId String -connectionName String
[-waitForComplete] [-pollTime String]
```

A.1.25.2 Description

This command deploys an assembly archive.

A.1.25.3 Options

Table A–25 shows the command options for `deployAssemblyInstance`.

Table A–25 deployAssemblyInstance options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assemblyInstanceId`	a	true	none	A string representing the `assemblyInstanceId`.	The identifier of an assembly instance to be deployed.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-waitForComplete`	w	false	no	N/A	Specifies whether to wait for an asynchronous operation to complete successfully.
`-pollTime`	pt	false	5	A string representing the number of seconds.	Specifies the amount of time to wait for an asynchronous operation to complete successfully.

A.1.25.4 Examples

Here are some command examples.

A.1.25.4.1 Deploying an Assembly Instance

```
$ abctl deployAssemblyInstance -assemblyInstanceId MyId
```

A.1.26 describeApplianceInstances

Details for this command follow.

A.1.26.1 Synopsis

```
$ abctl describeApplianceInstances [-assemblyInstanceId String...]
[-applianceInstanceId String...] [-applianceIndex String...] -connectionName
String
```

A.1.26.2 Description

Describes one or more deployed instances of an assembly.

A.1.26.3 Options

Table A–26 shows the command options for `describeApplianceInstances`.

Table A–26 describeApplianceInstances options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assemblyInstanceId`	as	false	none	A string representing the assemblyInstanceId.	One or more `assemblyInstanceIds`.
`-applianceInstanceId`	ap	false	none	A string representing the `applianceId`.	One or more `applianceIds`.
`-applianceIndex`	ai	false	none	A string representing the `instanceId`.	One or more `instanceIds`.
`-connectionName`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.26.4 Examples

Here are some command examples.

A.1.26.4.1 Describe Appliance Instances

```
$ abctl describeApplianceInstances
```

A.1.27 describeAssemblyArchives

Details for this command follow.

A.1.27.1 Synopsis

```
$ abctl describeAssemblyArchives [-assembly String...] -connectionName String
```

A.1.27.2 Description

Describes one or more assemblies in the Deployer.

A.1.27.3 Options

Table A–27 shows the command options for `describeAssemblyArchives`.

Table A–27 describeAssemblyArchives options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assembly`	a	false	none	A list of the assemblies to describe.	Specifies the assemblies to describe.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.27.4 Examples

Here are some command examples.

A.1.27.4.1 Describe Assembly Archives

```
$ abctl describeAssemblyArchives -assembly TheAssembly
```

A.1.28 describeAssemblyInstances

Details for this command follow.

A.1.28.1 Synopsis

```
$ abctl describeAssemblyInstances [-assemblyInstanceId String...] -connectionName
String
```

A.1.28.2 Description

Describes one or more assembly instances.

A.1.28.3 Options

Table A–28 shows the command options for `describeAssemblyInstances`.

Table A–28 describeAssemblyInstances options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assemblyInstanceId`	a	false	none	A comma-separated list of assembly instance IDs.	Identifiers of one or more assembly instances to be described.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.28.4 Examples

Here are some command examples.

A.1.28.4.1 Describe Assembly Instances

```
$ abctl describeAssemblyInstances
```

A.1.29 describeAssemblyUsers

Details for this command follow.

A.1.29.1 Synopsis

```
$ abctl describeAssemblyUsers -assembly String -connectionName String
```

A.1.29.2 Description

This command describes one or more users of an assembly.

A.1.29.3 Options

Table A–28 shows the command options for `describeAssemblyUsers`.

Table A–29 describeAssemblyUsers options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assembly`	a	false	none	A string representing the name of the assembly.	Specifies the assembly whose users will be described.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.29.4 Examples

Here are some command examples.

A.1.29.4.1 Describe Assembly Users

```
$ abctl describeAssemblyUsers -assembly MyAssembly
```

A.1.30 describeCatalog

Details for this command follow.

A.1.30.1 Synopsis

```
$ abctl describeCatalog [[-name] String] [-long]
```

A.1.30.2 Description

Lists appliances and assemblies in the catalog.

A.1.30.3 Options

Table A–30 shows the command options for `describeCatalog`.

Table A–30 describeCatalog options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-long`	l	false	N/A	N/A	Lists information with maximum detail. Included for compatibility only.
`-name`	n	false	None.	Name of an appliance or assembly. Nested appliances or assemblies are referred to with slash ('/'), for example: `mySite/myOhs`.	If not specified, all appliances and assemblies in the catalog are displayed. If the name of an assembly is specified, its sub-appliances and sub-assemblies are listed in addition to the assembly itself. If the name of an appliance is specified, only that appliance is listed.

A.1.30.4 Examples

Here are some command examples.

A.1.30.4.1 Describe Catalog in Long Format

```
$ abctl describeCatalog -name myAssembly -long
```

A.1.31 describeDeployer

Details for this command follow.

A.1.31.1 Synopsis

```
$ abctl describeDeployer -connectionName String
```

A.1.31.2 Description

Describes a Deployer instance.

A.1.31.3 Options

Table A–31 shows the command options for `describeDeployer`.

Table A–31 describeDeployer options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.31.4 Examples

Here are some command examples.

A.1.31.4.1 Describe Deployer

```
$ abctl describeDeployer
```

A.1.32 describeDeployerConnections

Details for this command follow.

A.1.32.1 Synopsis

```
$ abctl describeDeployerConnections
```

A.1.32.2 Description

Describes the configured Deployer connections.

A.1.32.3 Options

None.

A.1.32.4 Examples

Here are some command examples.

A.1.32.4.1 Describe Deployer Connections

```
$ abctl describeDeployerConnections
```

A.1.33 describeDeploymentPlans

Details for this command follow.

A.1.33.1 Synopsis

```
$ abctl describeDeploymentPlans [-name String] [-plan String] [-long]
```

A.1.33.2 Description

Describes the available deployment plans.

A.1.33.3 Options

Table A–31 shows the command options for `describeDeploymentPlans`.

Table A–32 describeDeploymentPlan options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	false	.*	Any regular expression.	Name of an assembly or assemblies, specified as a regular expression.
`-plan`	p	false	.*	Any regular expression.	Name of a plan or plans. It is specified as a regular expression.
`-long`	l	false	none	N/A	Flag to indicate if the long version of information is required.

A.1.33.4 Examples

Here are some command examples.

A.1.33.4.1 Describe Deployment Plans

```
$ abctl describeDeploymentPlans -name myAssembly -plan myPlan -long
```

A.1.34 describeEMAssemblyArchives

Details for this command follow.

A.1.34.1 Synopsis

```
$ abctl describeEMAssemblyArchives [-name nameOfAssemblyArchive]
```

A.1.34.2 Description

Describes assembly archives in the EM Software Library.

A.1.34.3 Options

Table A–31 shows the command options for `describeEMAssemblyArchives`.

Table A–33 describeEMAssemblyArchives options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	false	.*	Existing assembly archive names.	Name of an assembly or assemblies, specified as a regular expression.

A.1.34.4 Examples

Here are some command examples.

A.1.34.4.1 Describe EM Assembly Archives

```
$ abctl describeEMAssemblyArchives [-name nameOfAssemblyArchive]
```

A.1.35 describeEMConnection

Details for this command follow.

A.1.35.1 Synopsis

```
$ abctl describeEMConnection
```

A.1.35.2 Description

Describes the configured EM Software Library connection.

A.1.35.3 Options

None.

A.1.35.4 Examples

Here are some command examples.

A.1.35.4.1 Describe EM Connection

```
$ abctl describeEMConnection
```

A.1.36 describeEndpoints

Details for this command follow.

A.1.36.1 Synopsis

```
$ abctl describeEndpoints -name String [-recurse]
```

A.1.36.2 Description

Lists the endpoints of an appliance or assembly. Specify -recurse to list endpoints of an assembly's sub-appliances and sub-assemblies.

A.1.36.3 Options

Table A–34 shows the command options for `describeEndpoints`.

Table A–34 describeEndpoints options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	None.	Name of an appliance or assembly.	Name of an appliance or assembly whose endpoints will be listed.
`-recurse`	r	false	N/A	N/A	If specified, list the endpoints for an assembly's sub-elements instead of endpoints for the assembly itself.

A.1.36.4 Examples

Here are some command examples.

A.1.36.4.1 Describe Endpoints

```
$ abctl describeEndpoints -name mySite/myWls -r
```

A.1.37 describeRegistrations

Details for this command follow.

A.1.37.1 Synopsis

```
$ abctl describeRegistrations [-assembly String] [-version String] -connectionName
String
```

A.1.37.2 Description

This command one or more assembly registrations.

A.1.37.3 Options

Table A–35 shows the command options for `describeRegistrations`.

Table A–35 describeRegistrations options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assembly`	a	false	none	A string representing the name of the assembly.	The name of an assembly.
`-version`	v	true	none	A string representing the version of the assembly.	Assembly version.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.37.4 Examples

Here are some command examples.

A.1.37.4.1 Describe Registrations

```
$ abctl describeRegistrations -assembly MyAssembly -version 1
```

A.1.38 describeRequests

Details for this command follow.

A.1.38.1 Synopsis

```
$ abctl describeRequests [-requestId String...] -connectionName String
```

A.1.38.2 Description

This command describes one or more previously issued synchronous requests.

A.1.38.3 Options

Table A–36 shows the command options for `describeRequests`.

Table A–36 describeRequests options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-requestId`	r	false	none	A string representing the `requestId`.	The `requestId` of a previously issued synchronous request.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.38.4 Examples

Here are some command examples.

A.1.38.4.1 Describe Requests

```
$ abctl describeRequests
```

A.1.39 describeScalingGroups

Details for this command follow.

A.1.39.1 Synopsis

```
$ abctl describeScalingGroups [-assemblyInstanceId String...] [-scalingGroupId
String...] -connectionName String
```

A.1.39.2 Description

Describes one or more scaling groups.

A.1.39.3 Options

Table A–37 shows the command options for `describeScalingGroups`.

Table A–37 describeScalingGroups options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assemblyInstanceId`	a	false	none	A string representing the `assemblyInstanceId`.	The identifier of a previously created assembly instance.
`-scalingGroupId`	s	true	none	A string representing the `scalingGroupId`.	The identifier of a previously created scaling group.
`-connectionName`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.39.4 Examples

Here are some command examples.

A.1.39.4.1 Describe Scaling Groups

```
$ abctl describeScalingGroups
```

A.1.40 describeTags

Details for this command follow.

A.1.40.1 Synopsis

```
$ abctl describeTags -resource String [-tag String...] -connectionName String
```

A.1.40.2 Description

This command describes one or more tags associated with artifacts maintained by the Deployer.

A.1.40.3 Options

Table A–38 shows the command options for `describeTags`.

Table A–38 describeTags options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-resource`	-r	true	none	A string specifying the resource id of the resource.	Specifies the resource for which to get tag information.
`-tag`	-t	false	none	A string representing the name of the tag.	Specifies one or more tags for which to get the values.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.40.4 Examples

Here are some command examples.

A.1.40.4.1 Describe Tags

```
$ abctl describeTags -tag foo -resource MyResource
```

A.1.41 describeTargetConfigurations

Details for this command follow.

A.1.41.1 Synopsis

```
$ abctl describeTargetConfigurations [-target String...] -connectionName String
```

A.1.41.2 Description

This command describes one or more target configurations.

A.1.41.3 Options

Table A–39 shows the command options for `describeDeployer`.

Table A–39 describeDeployer options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-target`	t	false	none	A string representing the name of the target.	The name of one or more targets.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.41.4 Examples

Here are some command examples.

A.1.41.4.1 Describe Target Configurations

```
$ abctl describeTargetConfigurations -target MyTarget
```

A.1.42 describeTargetNames

Details for this command follow.

A.1.42.1 Synopsis

```
$ abctl describeTargetNames -type String -connectionName String
```

A.1.42.2 Description

Describes one or more types of assembly instance targets.

A.1.42.3 Options

Table A–40 shows the command options for `describeTargetNames`.

Table A–40 describeTargetNames options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-type`	t	true	none	A string representing the type of target. Possible values are `exalogic` or `ovm`.	The type of the target.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.42.4 Examples

Here are some command examples.

A.1.42.4.1 Describe Target Names

```
$ abctl describeTargetNames
```

A.1.43 describeTargetUsers

Details for this command follow.

A.1.43.1 Synopsis

```
$ abctl describeTargetUsers -target String -connectionName String
```

A.1.43.2 Description

Describes one or more users of assembly instance targets.

A.1.43.3 Options

Table A–41 shows the command options for `describeTargetUsers`.

Table A–41 describeTargetUsers options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-target`	t	true	none	A string representing the name of the target.	The name of the target.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.43.4 Examples

Here are some command examples.

A.1.43.4.1 Describe Target Users

```
$ abctl describeTargetUsers -target MyTarget
```

A.1.44 describeTargets

Details for this command follow.

A.1.44.1 Synopsis

```
$ abctl describeTargets [-target String...] -connectionName String
```

A.1.44.2 Description

This command describes runtime information for one or more deployment targets.

A.1.44.3 Options

Table A–42 shows the command options for `describeTargets`.

Table A–42 describeTargets options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-target`	t	false	none	A string representing the name of the target.	The name of one or more targets.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.44.4 Examples

Here are some command examples.

A.1.44.4.1 Describe Targets

```
$ abctl describeTargets
```

A.1.45 describeUserTargets

Details for this command follow.

A.1.45.1 Synopsis

```
$ abctl describeUserTargets -user String -connectionName String
```

A.1.45.2 Description

This command describes one or more types of deployment targets.

A.1.45.3 Options

Table A–43 shows the command options for `describeUserTargets`.

Table A–43 describeUserTargets options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-user`	u	true	none	A string representing the username of the user. Possible values are `exalogic` or `ovm`.	The username of the user.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.45.4 Examples

Here are some command examples.

A.1.45.4.1 Describe User Targets

```
$ abctl describeUserTargets -user MyUser
```

A.1.46 describeVnets

Details for this command follow.

A.1.46.1 Synopsis

```
$ abctl describeVnets -target String [-id String...] -connectionName String
```

A.1.46.2 Description

This command describes one or more networks.

A.1.46.3 Options

Table A–43 shows the command options for `describeVnets`.

Table A–44 describeVnets options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-id`	i	false	none	A string representing the network ID.	The network IDs to describe.
`-target`	t	true	none	A string representing the target name.	The name of a target whose networks will be described.

A.1.46.4 Examples

Here are some command examples.

A.1.46.4.1 Describe Vnets

```
$ abctl describeVnets
```

A.1.47 downloadAssemblyArchive

Details for this command follow.

A.1.47.1 Synopsis

```
$ abctl downloadAssemblyArchive -name String -version String [-fileName Path]
-connectionName String
```

A.1.47.2 Description

This command downloads an assembly archive from the Deployer repository.

A.1.47.3 Options

Table A–45 shows the command options for `downloadAssemblyArchive`.

Table A–45 downloadAssemblyArchive options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	A string representing the name of the assembly.	The name of the assembly archive.
`-version`	v	true	none	A string representing the version of the assembly.	The version of the assembly archive.
`-fileName`	r	true	none	A string representing the new name and/or location of the assembly.	The new name of the assembly archive.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.47.4 Examples

Here are some command examples.

A.1.47.4.1 Download Assembly Archive

```
$ abctl downloadAssemblyArchive -name MyAssembly -version 1 RenamedAssembly.ova
```

A.1.48 downloadAssemblyMetadata

Details for this command follow.

A.1.48.1 Synopsis

```
$ abctl downloadAssemblyMetadata -name String -version String [-fileName Path]
[-generatePlan] -connectionName String
```

A.1.48.2 Description

Downloads assembly metadata descriptor from the Deployer. This allows you to determine what is in the assembly without downloading the entire archive.

A.1.48.3 Options

Table A–46 shows the command options for `downloadAssemblyMetadata`.

Table A–46 downloadOVF options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-fileName`	fn	false	none	An absolute or relative path to a file where the assembly metadata will be written.	The new name of the OVF. The file name where the assembly metadata will be written. If omitted, the metadata will be saved as <assembly name>.ovf in the current working directory. You may specify an absolute or relative path to a file.

Table A–46 (Cont.) downloadOVF options

Name	Alias	Req'd	Default Values	Possible Values	Description
-generatePlan	g	false	none	N/A	If set, generate a default deployment plan. The plan will be generated in the same location where the downloaded metadata is saved.
-name	n	true	none	A string representing the name of the assembly.	The name of the assembly for which a metadata descriptor will be downloaded.
-version	v	true	none	A string representing the version of the assembly.	Assembly version.

A.1.48.4 Examples

Here are some command examples.

A.1.48.4.1 Download Assembly Metadata

```
$ abctl downloadAssemblyMetadata -name MyAssembly -version 1
```

A.1.49 downloadEMAssemblyArchive

Details for this command follow.

A.1.49.1 Synopsis

```
$ abctl downloadEMAssemblyArchive -name String -version String [-downloadAs
String] [-force]
```

A.1.49.2 Description

Downloads an assembly archive from the Enterprise Manager Software Library and imports it into the Oracle Virtual Assembly Builder Studio catalog. The assembly archive is reverse engineered to have the Oracle Virtual Assembly Builder metadata, file sets and templates created and persisted in the catalog.

By default, the download fails if an assembly with the same name already exists in the catalog. The `-downloadAs` option can be used to download an assembly with a different name.

A.1.49.3 Options

Table A–47 shows the command options for `downloadEMAssemblyArchive`.

Table A–47 downloadEMAssemblyArchive options

Name	Alias	Req'd	Default Values	Possible Values	Description
-name	n	true	none	Assembly archive name.	Name of assembly archive to download.
-version	v	true	none	A string representing the version of the assembly archive.	Specifies the version of the assembly archive to download from Enterprise Manager Software Library.
-force	f	false	false	N/A	If set, overwrites an existing assembly in the catalog that has the same name as the imported assembly.
-downloadAs		false	none	Assembly archive name.	Name to assign to a downloaded assembly inside the catalog.

A.1.49.4 Examples

Here is a command example.

A.1.49.4.1 downloadEMAssemblyArchive

```
% abctl downloadEMAssemblyArchive -name archiveName -version 1.0 -force
-downloadAs newName
```

A.1.50 export

Details for this command follow.

A.1.50.1 Synopsis

```
$ abctl export -name String -toDir Path [-quiet] [-metadataOnly]
```

A.1.50.2 Description

Exports an appliance or assembly to disk so that it can later be imported to another catalog.

A.1.50.3 Options

Table A–48 shows the command options for `export`.

Table A–48 export options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-metadataOnly`	m	false	none	N/A	Indicates that only the metadata portion of the appliance or assembly will be exported.
`-name`	n	true	none	Top level appliance or assembly in the catalog. Nested appliances cannot be exported.	Name of a top level appliance or assembly in the catalog.
`-quiet`	q	false	none	N/A	By default, the command shows detailed progress/success messages. If `-quiet` is set, the command turns off verbose mode and shows only one or two progress/success messages.
`-toDir`	td	true	none	A path to a directory. The directory must be empty. A non-existing directory will be created.	Path to the directory to which a specified appliance or assembly will be exported. If a relative path is given, it will be relative to AB_INSTANCE.

A.1.50.4 Examples

Here are some command examples.

A.1.50.4.1 Regular export

```
$ abctl export -name mySite -toDir /tmp/mySite.export
Executing export to /tmp/mySite.export.
  Step 1 of 4: Copying from source to dest.
    Copying: 100% of   52MB completed.
    Copying: 100% of  690MB completed.
    Copying: 100% of   86MB completed.
    Copying: 100% of   405B completed.
```

```
    Copying: 100% of  188MB completed.
    Copying: 100% of  1024B completed.
  Step 2 of 4: Copying from source to dest completed.
  Step 3 of 4: Archiving temporary catalog.
    Zipping: 100% of  138MB completed.
    Copying: 100% of  690MB completed.
    Copying: 100% of  188MB completed.
  Step 4 of 4: Archiving temporary catalog completed.
Successfully exported to /tmp/mySite.export.
```

A.1.50.4.2 Export with -metadataOnly flag

```
$ abctl export -name mySite -toDir /tmp/mySite.export -metadataOnly
Executing export to /tmp/mySite.export.
  Step 1 of 2: Archiving temporary catalog.
    Zipping: 100% of  6163B completed.
  Step 2 of 2: Archiving temporary catalog completed.
Successfully exported to /tmp/mySite.export.
```

A.1.50.4.3 Export to a non-empty directory

```
$ abctl export -name myOhs -toDir /tmp/non-empty-dir
Executing export to /tmp/non-empty-dir.
Error: OAB-7443: Failed to export myWls to /tmp/non-empty-dir.
Caused by: OAB-09509: Directory is not empty at /tmp/non-empty-dir.
  Action: Clean up the directory, or choose an empty directory.
```

A.1.51 getDefaultTarget

Details for this command follow.

A.1.51.1 Synopsis

```
$ abctl getDefaultTarget -connectionName String
```

A.1.51.2 Description

This command returns the default target.

A.1.51.3 Options

Table A–49 shows the command options for `getDefaultTarget`.

Table A–49 getDefaultTarget options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.51.4 Examples

Here are some command examples.

A.1.51.4.1 Get Default Target

```
$ abctl getDefaultTarget
```

A.1.52 getTargetType

Details for this command follow.

A.1.52.1 Synopsis

```
$ abctl getTargetType -name String -connectionName String
```

A.1.52.2 Description

This command returns the type of the target.

A.1.52.3 Options

Table A–50 shows the command options for `getTargetType`.

Table A–50 getTargetType options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	A string representing the name of the target	The name of the target.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.52.4 Examples

Here are some command examples.

A.1.52.4.1 Get Target Type

```
$ abctl getTargetType -name MyTarget
```

A.1.53 help

Details for this command follow.

A.1.53.1 Synopsis

```
help [[-command] string] [-usage]
```

A.1.53.2 Description

Prints a brief help message or more detailed help for a specified command.

A.1.53.3 Options

Table A–51 shows the command options for `help`.

Table A–51 help options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-command`	c	false	none	Any command of this utility.	Specifies the command for which Help should be printed.
`-usage`	u	false	none	N/A	Print only an option summary of the specified command.

A.1.53.4 Examples

Here is an example.

A.1.53.4.1 Print help, help for introspectWLS command and option summary for import command

```
abctl help, abctl help -command introspectWLS,
abctl help -usage -command import
```

A.1.54 import

Details for this command follow.

A.1.54.1 Synopsis

```
$ abctl import -from Path [-quiet] [-importAs String] [-force]
```

A.1.54.2 Description

Imports an appliance from a specified directory or an assembly from either a specified directory or a specified assembly archive.

A.1.54.3 Options

Table A–51 shows the command options for `import`.

Table A–52 import options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-from`	fr	true	none	An assembly archive file or a directory containing an exported appliance or assembly.	Path to a valid export location or assembly archive file.
`-quiet`	q	false	none	N/A	By default, the command shows detailed progress/success messages. If `-quiet` is set, the command turns off verbose mode and shows only one or two progress/success messages.
`-importAs`	ia	false	none	A unique name among top-level appliances or assemblies in a catalog.	If specified, imported appliancie or assembly will be saved with the given name in the catalog.
`-force`	f	false	false	N/A	If specified, existing top-level appliance or assembly in catalog using the same name as imported appliance or assembly will be overwritten.

A.1.54.4 Examples

Here are some command examples.

A.1.54.4.1 Import Assembly Archive

```
$ abctl import -from /tmp/mySite.ova -importAs myNewSite
```

A.1.54.4.2 Import from an Export Location

```
$ abctl import -fromDir /tmp/myWls.export -importAs wls_1
```

A.1.55 importExternalTemplate

Details for this command follow.

A.1.55.1 Synopsis

```
$ abctl importExternalTemplate -fromDir Path -name String -platform String
[-quiet] [-skipValidation] [-force]
```

A.1.55.2 Description

Imports an existing virtual machine template located at the specified directory into the catalog as an external appliance.

A.1.55.3 Options

Table A–53 shows the command options for `importExternalTemplate`.

Table A–53 importExternalTemplate options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-quiet`	q	false	none	N/A	By default, the command shows detailed progress/success messages. If `-quiet` is set, the command turns off verbose mode and shows only one or two progress/success messages.
`-fromDir`	fd	true	none	Directory containing the VM template.	Path to the directory containing the VM template.
`-name`	n	true	none	A name that will be unique among top-level appliances or assemblies in a catalog.	If set, the imported template will be saved as the given name in the target catalog.
`-platform`	p	true	false	A string representing any valid platform supported by the product.	Target platform for the imported external appliance.
`-skipValidation`	s	false	false	N/A	If specified, platform-specific validation will not be performed before importing the template.
`-force`	f	false	false	N/A	If specified, existing top-level appliance or assembly in catalog using the same name as imported external appliance will be overwritten.

A.1.55.4 Examples

Here are some command examples.

A.1.55.4.1 Import External Template

```
$ abctl importExternalTemplate -fromDir /my/template -name myAppliance
```

A.1.56 introspectCoherenceWeb

Details for this command follow.

A.1.56.1 Synopsis

```
$ abctl introspectCoherenceWeb -wlsHome Path -domainRoot Path -adminUser String
[-name String] [-force] [-skipFileSetCapture] [-remoteHost String] [-remoteUser
String] [-sudoUser String] [-remoteWorkingDir Path] [-remoteCleanup]
[-privateKeyFile Path]
```

A.1.56.2 Description

This command is an alias for `introspectWLS`. It examines the configuration of an installed WebLogic Server domain to determine what file sets must be captured and what configuration must be changed at deployment. All collected data is stored in the catalog upon successful completion.

A.1.56.3 Options

Table A–54 shows the command options for `introspectCoherenceWeb`.

Table A–54 introspectCoherenceWeb options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-adminUser`	au	true	none	N/A	Administrative name for the WebLogic Server domain.
`-domainRoot`	dr	true	none	N/A	Full path to the WebLogic Server domain root.
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name.	Any name not already used within the catalog.	Specifies a name by which the introspection output is stored.
`-privateKeyFile`	pkf	false	none	N/A	Private SSH key file on the local machine.
`-remoteCleanup`	rc	false	none	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, the remoteUser and remoteHost must be specified as well.
`-remoteHost`	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, the remoteUser must be specified as well.
`-remoteUser`	ru	false	none	N/A	Name of the ssh user to use for accessing the remote machine. If set, the remoteHost must be specified as well.
`-remoteWorkingDir`	rwd	false	`/tmp/abRemote_<remote user name>`	N/A	Path on the remote machine to work out of. If set, the remoteUser and remoteHost must be specified as well.
`-skipFileSetCapture`	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
`-sudoUser`	su	false	none	User name of sudo user.	Specifies a sudo user. When specified, remote capturing of file sets or introspection will substitute the user (sudo) as the sudo user before running the remote Assembly Builder. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.
`-wlsHome`	wh	true	none	N/A	Full path to the WebLogic Server Home, generally `<middleware home>/wlserver_10.3`

A.1.56.4 Examples

Here is a command example.

A.1.56.4.1 Basic Introspection of a Coherence Appliance

This is a basic introspection of a Coherence appliance to a specific catalog, using a capture name of myIntrospection.

```
$ abctl introspectCoherenceWeb -name myIntrospection <Coherence options>
```

A.1.57 introspectForms

Details for this command follow.

A.1.57.1 Synopsis

```
introspectForms -wlsHome Path -domainRoot Path -adminUser String [-soaGlobalCP
Path] [-name String] [-force] [-skipFileSetCapture] [-remoteHost String]
[-remoteUser String] [-sudoUser String] [-remoteWorkingDir Path] [-remoteCleanup]
[-privateKeyFile Path]
```

A.1.57.2 Description

This command is an alias for `introspectWLS`. Examines the configuration of an installed Oracle WebLogic Server domain to determine what file sets needs to be captured and what configuration needs to be changed at deployment.

All collected data is stored in the catalog upon successful completion.

A.1.57.3 Options

Table A–55 shows the command options for `introspectForms`.

Table A–55 introspectForms options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-adminUser`	au	true	none	N/A	Administrative name for the Oracle WebLogic Server domain.
`-domainRoot`	dr	true	none	N/A	Full path to the Oracle WebLogic Server domain root.
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name.	Any name not already used within the catalog.	Specifies a name by which the introspection output is stored.
`-privateKeyFile`	pkf	false	none	N/A	Private SSH key file on the local machine.
`-remoteCleanup`	rc	false	false	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, `remoteUser` and `remoteHost` must be specified as well.
`-remoteHost`	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, `remoteUser` must be specified as well.
`-remoteUser`	ru	false	none	N/A	Name of the SSH user to use for accessing the remote machine. If set, `remoteHost` must be specified as well.
`-remoteWorkingDir`	rwd	false	`/tmp/abRemote_<remote user name>`	N/A	Path on the remote machine to work out of. If set, `remoteUser` and `remoteHost` must be specified as well.
`-soaGlobalCP`	sgcp	false	none	Location of optional global configuration plan.	The absolute path to an optional global configuration plan.

Table A–55 (Cont.) introspectForms options

Name	Alias	Req'd	Default Values	Possible Values	Description
-skipFileSetCapture	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
-sudoUser	su	false	none	User name of sudo user.	Specifies a sudo user. When specified, remote capturing of file sets or introspection will substitute the user (sudo) as the sudo user before running the remote Assembly Builder. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.
-wlsHome	wh	true	none	WLS Home	Full path to the Oracle WebLogic Server home (usually, <middleware home>/wlserver_10.3).

A.1.57.4 Examples

Here are some command examples.

A.1.57.4.1 Basic Introspection

This is a basic introspection of a Forms appliance.

```
abctl introspectForms <Forms options>
```

A.1.57.4.2 Introspection with a specific capture name

An introspection of component "Forms" saved with an appliance/assembly name of "myIntrospection":

```
% abctl introspectForms -name myIntrospection <Forms options>
```

A.1.58 introspectGenericProd

Details for this command follow.

A.1.58.1 Synopsis

```
$ abctl introspectGenericProd -productRoots String [-propertyFile Path]
[-scriptRootDir Path] [-name String] [-force] [-skipFileSetCapture] [-remoteHost
String] [-remoteUser String] [-sudoUser String] [-remoteWorkingDir Path]
[-remoteCleanup] [-privateKeyFile Path]
```

A.1.58.2 Description

Captures products generically by taking as input the set of product directories to capture, a set of properties that can be modified by the user, and a set of scripts to run on the appliance instance during deployment operations.

A.1.58.3 Options

Table A–56 shows the command options for introspectGenericProd.

Table A–56 introspectGenericProd options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name.	Any name not already used within the catalog.	Specifies a name by which the introspection output is stored.
`-privateKeyFile`	pkf	false	none	N/A	Private SSH key file on the local machine.
`-productRoots`	pr	true	none	The colon-separated list of product directories to capture.	A list of one or more colon-separated paths. Each path must be a directory that exists. All files and directories within each specified directory are captured as file sets. All specified paths are available at the same locations on the appliance instance during deployment.

Table A–56 (Cont.) introspectGenericProd options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-propertyFile`	pf	true	none	Properties file containing properties to add to the appliance.	If the `propertyFile` parameter is specified then it must point to a file that exists and is readable. A property file must be a text file containing a list of name/value pairs. Each property in the property file will be added as a user property into the appliance. Like any other user property, these properties can be edited in the generated appliance and can be overridden in deployment plans. During deployment the properties will be written back out to a file, including the values as edited within the appliance or overridden in the deployment plan. The regenerated properties file will be made available to all scripts during their execution through an environment variable named '`$AB_USERPROPS_FILE`'. These properties are intended for eventual consumption by the scripts captured through the '`scriptRootDir`' parameter. For this reason, the property names and values must be in a format that can be sourced by a shell script. Each line in a property file must consist of zero or more lines where each line must be a property declaration, a comment, or a blank line. Each property declaration must be equivalent to a variable declaration (name=value) where the name can be converted to an environment variable. Property declarations must be contained on a single line. Ending a line with '\' will not result in line continuation. Comments and blank lines are discarded at dehydration and will not be reproduced when the file is regenerated at reconfiguration. All properties will be marked as 'required' in the appliance metadata. Property declarations without any assigned value (nothing after '=') will be set to null in the appliance metadata, requiring that the user assign a value to that property prior to deployment. Whitespace is not permitted anywhere to the left of '=' in a property declaration. Whitespace to the right of '=' is assumed to be part of the intended value and will be preserved (resulting in failure if the value is sourced). Quotes around property values will be preserved and will be visible to scripts as part of the value. When editing a property value, it is the responsibility of the user to add/remove/preserve quotes as necessary according to the rules of shell interpretation.
`-remoteCleanup`	rc	false	none	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, the remoteUser and remoteHost must be specified as well.
`-remoteHost`	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, the remoteUser must be specified as well.

Table A–56 (Cont.) introspectGenericProd options

Name	Alias	Req'd	Default Values	Possible Values	Description
-remoteUser	ru	false	none	N/A	Name of the ssh user to use for accessing the remote machine. If set, the remoteHost must be specified as well.
-remoteWorkingDir	rwd	false	/tmp/abRemote_<remote user name>	N/A	Path on the remote machine to work out of. If set, the remoteUser and remoteHost must be specified as well.
-scriptRootDir	srd	false	none	The top level directory containing the script subdirectories.	The script root directory is the top level directory containing the script subdirectories. If the specified directory does not exist or is not readable then an error will be returned and an appliance will not be created. Scripts must be placed within the root script directory under the following well-known subdirectories: config.d/, start.d/, ping.d/, stop.d/. Scripts under each subdirectory will be captured during introspection and stored with the appliance. During deployment the appropriate set of scripts according to the requested operation will be executed sequentially. The script root directory need not contain all well-known subdirectories and well-known subdirectories that do exist may be empty. All scripts are executed as the root user to provide the flexibility of performing operations requiring root privileges or switching to another user as necessary. The path to a properties file containing the variables specified at introspection will be made available during script execution through the '$AB_USERPROPS_FILE' environment variable. This file can be sourced by the script. All scripts must exit with a zero exit status upon success. Any script exiting with a non-zero exit status will result in the failure of the operation.
-skipFileSetCapture	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
-sudoUser	su	false	none	User name of sudo user.	Specifies a sudo user. When specified, remote capturing of file sets or introspection will substitute the user (sudo) as the sudo user before running the remote Assembly Builder. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.

A.1.58.4 Examples

Here is a command example.

A.1.58.4.1 Basic Introspection of Appliance "GenericProd"

This is a basic introspection of appliance Coherence to a specific catalog, using a capture name of myIntrospection.

```
$ abctl introspectGenericProd <GenericProd options>
```

A.1.58.4.2 Basic Introspection of Appliance "GenericProd"

This is an introspection of a generic appliance saved with an appliance/assembly name of "myIntrospection".

```
$ abctl introspectGenericProd -name myIntrospection <GenericProd options>
```

A.1.59 introspectOHS

Details for this command follow.

A.1.59.1 Synopsis

```
introspectOHS -oracleInstance Path -componentName String [-name string] [-force]
[-skipFileSetCapture] [-remoteHost String] [-remoteUser String] [-sudoUser String]
[-remoteWorkingDir Path] [-remoteCleanup] [-privateKeyFile Path]
```

A.1.59.2 Description

Examines the configuration of an installed OHS component to determine what file sets must be captured and what configuration must be changed at deployment. All collected data is stored in the catalog upon successful completion.

A.1.59.3 Options

Table A–57 shows the command options for `introspectOHS`.

Table A–57 introspectOHS options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-componentName`	cn	true	none	N/A	The name of the Oracle HTTP Server appliance to introspect (for example: `ohs1`).
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name	Any name not previously used within the catalog	Specifies a name by which the introspection output is stored.
`-oracleInstance`	oi	true	none	N/A	The absolute path of the ORACLE_INSTANCE to introspect.
`-privateKeyFile`	pkf	false	none	Location of a private key file.	Private SSH key file on the local machine.
`-remoteCleanup`	rc	false	false	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, `remoteUser` and `remoteHost` must be specified as well.
`-remoteHost`	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, `remoteUser` must be specified as well.
`-remoteUser`	ru	false	none	N/A	Name of the SSH user to use for accessing the remote machine. If set, `remoteHost` must be specified as well.
`-remoteWorkingDir`	rwd	false	`/tmp/abRemote`	N/A	Path on the remote machine to work out of. If set, `remoteUser` and `remoteHost` must be specified as well.
`-skipFileSetCapture`	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
`-sudoUser`	su	false	none	User name of sudo user.	Name of the user on the remote machine to sudo as before executing operations. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.

A.1.59.4 Examples

Here are some command examples.

A.1.59.4.1 Successful Introspection

```
% abctl introspectOHS -name myOHS -oracleInstance /ora/inst1 -componentName ohs1
Launching introspection of appliance 'OHS' ...
  Step 1 of 5: OHS introspection starting
    Step 1 of 4: OHS Httpd Configuration parsed
    Step 2 of 4: OHS Httpd configuration transformed
    Step 3 of 4: OHS Httpd configuration processed
    Step 4 of 4: OHS Httpd configuration written
  Step 2 of 5: HTTPD processing completed
    Step 1 of 3: OHS OPMN configuration parsed
    Step 2 of 3: OHS OPMN configuration processed
    Step 3 of 3: OHS OPMN configuration writtend
  Step 3 of 5: OPMN XML processing completed
    Step 1 of 2: OHS opmnctl script parsed
    Step 2 of 2: Appliance updated with ORACLE_HOME
  Step 4 of 5: OPMNCTL processing completed
  Step 5 of 5: OHS introspection complete
Task is done: DehydrateJob completed
Introspection complete
Storing result in catalog: '/Oracle/IntrospectionCatalog' ...
Introspection stored as 'myohs' in the catalog
%
```

A.1.59.4.2 Failed Introspection bad -oracleInstance value

```
% abctl introspectOHS -oracleInstance /ora/dontexist -componentName foobar
Launching introspection of appliance 'OHS' ...
  Step 1 of 5: OHS task starting
Task is done: DehydrateJob failed with error: The specified Oracle Instance does
not exist.
Error: Introspection failed
Caused by: The specified Oracle Instance does not exist.
%
```

A.1.60 introspectOTD

Details for this command follow.

A.1.60.1 Synopsis

```
introspectOTD -oracleHome Path -oracleInstance Path -configName String [-name
String] [-force] [-skipFileSetCapture] [-remoteHost String] [-remoteUser String]
[-sudoUser String] [-remoteWorkingDir Path] [-remoteCleanup] [-privateKeyFile
Path]
```

A.1.60.2 Description

Examines the configuration of an installed Oracle Traffic Director configuration to determine what file sets need to be captured and what configuration needs to be changed at deployment. All collected data is stored in the catalog upon successful completion. Note that Oracle Traffic Director administration server will not be introspected and will be recreated from scratch during reconfiguration.

A.1.60.3 Options

Table A–58 shows the command options for `introspectOTD`.

Table A–58 introspectOTD options

Name	Alias	Req'd	Default Values	Possible Values	Description
-configName	cn	true	none	Name of the Oracle Traffic Director configuration.	Specifies the name of an Oracle Traffic Director configuration which needs to be introspected.
-force	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
-name	n	false	Derived directory name prefixed by component type name	Any name not previously used within the catalog	Specifies a name by which the introspection output is stored.
-oracleHome	oh	true	none	The directory where Oracle Traffic Director is installed.	Specify the absolute path to the directory where Oracle Traffic Director is installed.
-oracleInstance	oi	true	none	N/A	The absolute path of the ORACLE_INSTANCE to introspect.
-privateKeyFile	pkf	false	none	Location of a private key file.	Private SSH key file on the local machine.
-remoteCleanup	rc	false	false	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, remoteUser and remoteHost must be specified as well.
-remoteHost	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, remoteUser must be specified as well.
-remoteUser	ru	false	none	N/A	Name of the SSH user to use for accessing the remote machine. If set, remoteHost must be specified as well.
-remoteWorkingDir	rwd	false	/tmp/abRemote	N/A	Path on the remote machine to work out of. If set, remoteUser and remoteHost must be specified as well.
-skipFileSetCapture	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
-sudoUser	su	false	none	User name of sudo user.	Name of the user on the remote machine to sudo as before executing operations. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.

A.1.60.4 Examples

Here are some command examples.

A.1.60.4.1 Basic Introspection

```
% abctl introspectOTD <OTD options>
```

A.1.60.4.2 Introspection of appliance "OTD" saved with the name "myIntrospection"

```
% abctl introspectOTD -name myIntrospection <OTD options>
```

A.1.61 introspectRACDB

Details for this command follow.

A.1.61.1 Synopsis

```
introspectRACDB -crsHome Path -dbHome Path [-globalDbName String] [-sysDBAUserName
String] -shutdownDBOK String [-asmHome Path] [-name String] [-force]
[-skipFileSetCapture] [-remoteHost String] [-remoteUser String] [-sudoUser String]
[-remoteWorkingDir Path] [-remoteCleanup] [-privateKeyFile Path]
```

A.1.61.2 Description

Examines CRS and RAC Database configuration and captures metadata.

A.1.61.3 Options

Table A–59 shows the command options for `introspectRACDB`.

Table A–59 introspectRACDB options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-asmHome`	ch	false	none	N/A	This parameter is required if ASM is used as the storage type and it is installed in a separate Oracle Home.
`-crsHome`	ch	true	none	N/A	The ORACLE_HOME of the Oracle CRS to be introspected.
`-dbHome`	dh	true	none	N/A	The ORACLE_HOME of the Oracle RDBMS to be introspected.
`-globalDbName`	dun	false	value specified for -oracleSid	N/A	The global database name of the Oracle RDBMS to be introspected.
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name.	Any name not already used within the catalog.	Specifies a name by which the introspection output is stored
`-privateKeyFile`	pkf	false	none	N/A	Private SSH key file on the local machine.
`-remoteCleanup`	rc	false	none	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, `remoteUser` and `remoteHost` must be specified as well.
`-remoteHost`	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, `remoteUser` must be specified as well.
`-remoteUser`	ru	false	none	N/A	Name of the ssh user to use for accessing the remote machine. If set, `remoteHost` must be specified as well.
`-remoteWorkingDir`	rwd	false	`/tmp/abRemote_<remote user name>`	N/A	Path on the remote machine to work out of. If set, `remoteUser` and `remoteHost` must be specified as well.

Table A–59 (Cont.) introspectRACDB options

Name	Alias	Req'd	Default Values	Possible Values	Description
-shutdownDBOK	sdbok	true	none	N/A	This flag needs to be passed to approve the database reboot.
-skipFileSetCapture	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
-sysDBAUserName	sdbaun	false	none	N/A	Database account with SYSDBA privileges. This parameter is required only if OS authentication is disabled for the current database.
-sudoUser	su	false	none	User name of sudo user.	Specifies a sudo user. When specified, remote capture of file sets or introspection will substitute the user (sudo) as the sudo user before running the remote Assembly Builder. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.

A.1.61.4 Examples

Here are some command examples.

A.1.61.4.1 Basic Introspection

This is a basic introspection of a single-instance DB appliance.

```
abctl introspectRACDB <DB options>
```

A.1.61.4.2 Introspection into a specific catalog with a specific capture name

```
% abctl introspectRACDB -name myIntrospection <DB options>
```

A.1.62 introspectReports

Details for this command follow.

A.1.62.1 Synopsis

```
introspectReports -wlsHome Path -domainRoot Path -adminUser String [-soaGlobalCP
Path] [-name String] [-force] [-skipFileSetCapture] [-remoteHost String]
[-remoteUser String] [-sudoUser String] [-remoteWorkingDir Path] [-remoteCleanup]
[-privateKeyFile Path]
```

A.1.62.2 Description

This command is an alias for `introspectWLS`. Examines the configuration of an installed Oracle WebLogic Server domain to determine what file sets needs to be captured and what configuration needs to be changed at deployment.

All collected data is stored in the catalog upon successful completion.

A.1.62.3 Options

Table A–62 shows the command options for `introspectReports`.

Table A–60 introspectReports options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-adminUser`	au	true	none	N/A	Administrative name for the Oracle WebLogic Server domain.
`-domainRoot`	dr	true	none	N/A	Full path to the Oracle WebLogic Server domain root.
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name.	Any name not already used within the catalog.	Specifies a name by which the introspection output is stored.
`-privateKeyFile`	pkf	false	none	N/A	Private SSH key file on the local machine.
`-remoteCleanup`	rc	false	false	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, `remoteUser` and `remoteHost` must be specified as well.
`-remoteHost`	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, `remoteUser` must be specified as well.
`-remoteUser`	ru	false	none	N/A	Name of the SSH user to use for accessing the remote machine. If set, `remoteHost` must be specified as well.
`-remoteWorkingDir`	rwd	false	`/tmp/abRemote_<remote user name>`	N/A	Path on the remote machine to work out of. If set, `remoteUser` and `remoteHost` must be specified as well.
`-soaGlobalCP`	sgcp	false	none	Location of optional global configuration plan.	The absolute path to an optional global configuration plan.
`-skipFileSetCapture`	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
`-sudoUser`	su	false	none	User name of sudo user.	Specifies a sudo user. When specified, remote capturing of file sets or introspection will substitute the user (sudo) as the sudo user before running the remote Assembly Builder. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.
`-wlsHome`	wh	true	none	WLS Home	Full path to the Oracle WebLogic Server home (usually, `<middleware home>/wlserver_10.3`).

A.1.62.4 Examples

Here are some command examples.

A.1.62.4.1 Basic Introspection

This is a basic introspection of a Reports appliance.

```
abctl introspectReports <Reports options>
```

A.1.62.4.2 Introspection with a specific capture name

An introspection of component "Reports" saved with an appliance/assembly name of "myIntrospection":

```
% abctl introspectReports -name myIntrospection <Reports options>
```

A.1.63 introspectSIDB

Details for this command follow.

A.1.63.1 Synopsis

```
introspectSIDB -dbHome Path -oracleSid String [-name String] [-force] [-noing]
[-remoteHost String] [-remoteUser String] [-remoteWorkingDir Path] -shutdownDBOK
String [-remoteCleanup] [-dataFileDir Path] [-flashRecoveryDir Path] [-sudoUser]
```

A.1.63.2 Description

Examines single-instance Oracle database (releases 10.2, 11.1, 11.2) configuration and captures metadata.

A.1.63.3 Options

Table A–61 shows the command options for `introspectSIDB`.

Table A–61 introspectSIDB options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-dataFileDir`	dfd	false	DB 10.2 release: `<Parent directory of $ORACLE_HOME>/oradata` DB 11.1 and 11.2 release: `$ORACLE_BASE/oradata`	N/A	The full path of the database files. This parameter is required if your database file directory is different from the default.
`-dbHome`	dh	true	none	N/A	The ORACLE_HOME of the Oracle RDBMS to be introspected.
`-flashRecoveryDir`	frd	false	DB 10.2 release: `<Parent directory of $ORACLE_HOME>/flash_recovery_area` DB 11.1 release: `$ORACLE_BASE/flash_recovery_area` DB 11.2 release: `$ORACLE_BASE/recovery_area`	N/A	The full path of the database flash recovery files. This parameter is required if your recovery area is different from the default. If you do not have a recovery area, you can ignore this parameter.
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name.	Any name not already used within the catalog.	Specifies a name by which the introspection output is stored
`-oracleSid`	os	true	none	N/A	The SID of the Oracle RDBMS to be introspected.
`-privateKeyFile`	pkf	false	none	N/A	Private SSH key file on the local machine.
`-remoteCleanup`	rc	false	none	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, `remoteUser` and `remoteHost` must be specified as well.

Table A–61 (Cont.) introspectSIDB options

Name	Alias	Req'd	Default Values	Possible Values	Description
-remoteHost	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, remoteUser must be specified as well.
-remoteUser	ru	false	none	N/A	Name of the ssh user to use for accessing the remote machine. If set, remoteHost must be specified as well.
-remoteWorkingDir	rwd	false	/tmp/abRemote_<remote user name>	N/A	Path on the remote machine to work out of. If set, remoteUser and remoteHost must be specified as well.
-shutdownDBOK	sdbok	true	none	N/A	This flag needs to be passed to approve the database reboot.
-skipFileSetCapture	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
-sudoUser	su	false	none	User name of sudo user.	Specifies a sudo user. When specified, remote capture of file sets or introspection will substitute the user (sudo) as the sudo user before running the remote Assembly Builder. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.

A.1.63.4 Examples

Here are some command examples.

A.1.63.4.1 Basic Introspection

This is a basic introspection of a single-instance DB appliance.

```
abctl introspectSIDB <DB options>
```

A.1.63.4.2 Introspection into a specific catalog with a specific capture name

```
% abctl introspectSIDB -name myIntrospection <DB options>
```

A.1.64 introspectSOA

Details for this command follow.

A.1.64.1 Synopsis

```
introspectSOA -wlsHome Path -domainRoot Path -adminUser String [-soaGlobalCP Path]
[-name String] [-force] [-skipFileSetCapture] [-remoteHost String] [-remoteUser
String] [-sudoUser String] [-remoteWorkingDir Path] [-remoteCleanup]
[-privateKeyFile Path]
```

A.1.64.2 Description

This command is an alias for `introspectWLS`. Examines the configuration of an installed Oracle WebLogic Server domain to determine what file sets need to be captured and what configuration needs to be changed at deployment.

All collected data is stored in the catalog upon successful completion.

A.1.64.3 Options

Table A–62 shows the command options for `introspectSOA`.

Table A–62 introspectSOA options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-adminUser`	au	true	none	N/A	Administrative name for the Oracle WebLogic Server domain.
`-domainRoot`	dr	true	none	N/A	Full path to the Oracle WebLogic Server domain root.
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name.	Any name not already used within the catalog.	Specifies a name by which the introspection output is stored.
`-privateKeyFile`	pkf	false	none	N/A	Private SSH key file on the local machine.
`-remoteCleanup`	rc	false	false	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, `remoteUser` and `remoteHost` must be specified as well.
`-remoteHost`	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, `remoteUser` must be specified as well.
`-remoteUser`	ru	false	none	N/A	Name of the SSH user to use for accessing the remote machine. If set, `remoteHost` must be specified as well.
`-remoteWorkingDir`	rwd	false	`/tmp/abRemote_<remote user name>`	N/A	Path on the remote machine to work out of. If set, `remoteUser` and `remoteHost` must be specified as well.
`-soaGlobalCP`	sgcp	false	none	Location of optional global configuration plan.	The absolute path to an optional global configuration plan.
`-skipFileSetCapture`	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
`-sudoUser`	su	false	none	User name of sudo user.	Specifies a sudo user. When specified, remote capturing of file sets or introspection will substitute the user (sudo) as the sudo user before running the remote Assembly Builder. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.
`-wlsHome`	wh	true	none	WLS Home	Full path to the Oracle WebLogic Server home (usually, `<middleware home>/wlserver_10.3`).

A.1.64.4 Examples

Here are some command examples.

A.1.64.4.1 Basic Introspection

This is a basic introspection of a single-instance DB appliance.

```
abctl introspectSOA <SOA options>
```

A.1.64.4.2 Introspection with a specific capture name

```
% abctl introspectSOA -name myIntrospection <SOA options>
```

A.1.65 introspectTuxedo

Details for this command follow.

A.1.65.1 Synopsis

```
introspectTuxedo -TUXDIR Path -TUXCONFIG Path [-environmentScript Path]
[-oracleClientDir Path] [-tnsNamesLocation Path] [-artCICSAppHome Path]
[-artBatchSecurityProfile Path] [-name String] [-force] [-skipFileSetCapture]
[-remoteHost String] [-remoteUser String] [-sudoUser String] [-remoteWorkingDir
Path] [-remoteCleanup] [-privateKeyFile Path]
```

A.1.65.2 Description

Examines a single or multiple-machine Oracle Tuxedo domain, and the Oracle Home Directory that it resides on (including add-ons).

A.1.65.3 Options

Table A–63 shows the command options for `introspectTuxedo`.

Table A–63 introspectTuxedo options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-artCICSBatchHome`	acicsah	false	none	N/A	The absolute path to the ART CICS Application Home.
`-artBatchSecurityProfile`	absp	false	none	N/A	The absolute path to the security_profile for ART Batch.
`-environmentScript`	es	false	none	The absolute path to the environment script of the application to introspect	The absolute path to the script that sets the environment of the Tuxedo application to introspect.
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name.	Any name not already used within the catalog.	Specifies a name by which the introspection output is stored.
`-oracleClientDir`	ocd	false	none	The absolute path to the Oracle Database Client software.	The absolute path to the location where the Oracle Database Client software is installed.
`-privateKeyFile`	pkf	false	none	N/A	Private SSH key file on the local machine.
`-remoteCleanup`	rc	false	false	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, `remoteUser` and `remoteHost` must be specified as well.

Table A–63 (Cont.) introspectTuxedo options

Name	Alias	Req'd	Default Values	Possible Values	Description
-remoteHost	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, remoteUser must be specified as well.
-remoteUser	ru	false	none	N/A	Name of the SSH user to use for accessing the remote machine. If set, remoteHost must be specified as well.
-remoteWorkingDir	rwd	false	/tmp/abRemote_<remote user name>	N/A	Path on the remote machine to work out of. If set, remoteUser and remoteHost must be specified as well.
-soaGlobalCP	sgcp	false	none	Location of optional global configuration plan.	The absolute path to an optional global configuration plan.
-skipFileSetCapture	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
-sudoUser	su	false	none	User name of sudo user.	Specifies a sudo user. When specified, remote capturing of file sets or introspection will substitute the user (sudo) as the sudo user before running the remote Assembly Builder. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.
-tnsNamesLocation	tnl	false	none	The absolute path to the TNSNAMES.ora file.	The absolute path to the location of the TNSNAMES.ora file.
-TUXDIR	tuxdir	true	none	N/A	The absolute path to the TUXDIR to introspect.
-TUXCONFIG	tuxconfig	true	none	N/A	The absolute path to the TUXCONFIG file of the application to introspect.

A.1.65.4 Examples

Here are some command examples.

A.1.65.4.1 Basic Introspection

This is a basic introspection of a single-instance DB appliance.

```
abctl introspectTuxedo <Tuxedo options>
```

A.1.65.4.2 Introspection with a specific capture name

```
% abctl introspectTuxedo -name myIntrospection <Tuxedo options>
```

A.1.66 introspectWebCache

Details for this command follow.

A.1.66.1 Synopsis

```
$ abctl introspectWebCache -oracleInstance Path -componentName String [-name
String] [-force] [-skipFileSetCapture] [-remoteHost String] [-remoteUser String]
[-sudoUser String] [-remoteWorkingDir Path] [-remoteCleanup] [-privateKeyFile
Path]
```

A.1.66.2 Description

Examines the configuration of an installed Web Cache component and records what must be captured as file sets, and what must be configured during deployment. All collected data is stored in the catalog upon successful completion.

A.1.66.3 Options

Table A–64 shows the command options for `introspectWebCache`.

Table A–64 introspectWebCache options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-componentName`	cn	true	none	N/A	The name of the Web Cache component to introspect.
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name.	Any name not already used within the catalog.	Specifies a name by which the introspection output is stored.
`-oracleInstance`	oi	true	none	N/A	The absolute path of the ORACLE_INSTANCE to introspect.
`-privateKeyFile`	pkf	false	none	N/A	Private SSH key file on the local machine.
`-remoteCleanup`	rc	false	false	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, `remoteUser` and `remoteHost` must be specified as well.
`-remoteHost`	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, `remoteUser` must be specified as well.
`-remoteUser`	ru	false	none	N/A	Name of the SSH user to use for accessing the remote machine. If set, `remoteHost` must be specified as well.
`-remoteWorkingDir`	rwd	false	`/tmp/abRemote_<remote user name>`	N/A	Path on the remote machine to work out of. If set, `remoteUser` and `remoteHost` must be specified as well.
`-skipFileSetCapture`	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
`-sudoUser`	su	false	none	User name of sudo user.	Specifies a sudo user. When specified, remote capturing of file sets or introspection will substitute the user (sudo) as the sudo user before running the remote Assembly Builder. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.

A.1.66.4 Examples

Here are some command examples.

A.1.66.4.1 Basic introspection of Web Cache appliance to the default catalog and allowing introspection to choose a default capture name within the catalog

```
$  introspectWebCache <WebCache options>
```

A.1.66.4.2 Introspection of Web Cache appliance put into a specific catalog under a capture name of "web cache"

```
abctl introspectWebCache -name webcache -oracleInstance
/oracle/instances/instance1 -componentName webcache1
```

```
Launching introspection of appliance 'WebCache' ...
Step 1 of 9: Webcache task starting
Step 9 of 9: Webcache task complete
Task is done: DehydrateJob completed
Introspection complete
Storing result in catalog ...
backup needed
Introspection stored as 'webcache' in the catalog
```

A.1.66.4.3 Introspection with incorrect component Name

```
abctl introspectWebCache -name webcache -oracleInstance
 /bea/Oracle_WT1/instances/instance1/ -componentName webcache2
Launching introspection of appliance 'WebCache' ...
Task is done: DehydrateJob failed with error: Unable to find file: /bea/Oracle_
WT1/instances/instance1/config/WebCache/webcache2/webcache.xml
Error: Introspection failed
Caused by: Unable to find file: /bea/Oracle_
WT1/instances/instance1/config/WebCache/webcache2/webcache.xml
```

A.1.67 introspectWLS

Details for this command follow.

A.1.67.1 Synopsis

```
introspectWLS -wlsHome Path -domainRoot Path -adminUser String [-soaGlobalCP Path]
[-name String] [-force] [-skipFileSetCapture] [-remoteHost String] [-remoteUser
String] [-sudoUser String] [-remoteWorkingDir Path] [-remoteCleanup]
[-privateKeyFile Path]
```

A.1.67.2 Description

Examines the configuration of an installed WebLogic Server component to determine what file sets must be captured and what configuration must be changed at deployment. All collected data is stored in the catalog upon successful completion.

A.1.67.3 Extensions

The `CoherenceWeb`, `SOACoherence`, and SOA extensions are available as alias commands.

A.1.67.4 CoherenceWeb Extension Description

Inspects and captures Coherence cluster and cache server configuration that is defined within a WebLogic domain configuration. This extension supports the out-of-process deployment topology in which cache servers run in their own processes.

A.1.67.5 SOACoherence Extension Description

Inspects and captures Coherence configuration specified within SOA managed server start arguments to enable SOA cluster high availability.

A.1.67.6 SOA Extension Description

Oracle SOA platform plug-in. Examines the configuration of an installed Oracle WebLogic Server domain to determine what file sets need to be captured and what configuration needs to be changed at deployment.

A.1.67.7 Options

Table A–65 shows the command options for `introspectWLS`.

Table A–65 introspectWLS options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-adminUser`	au	true	none	N/A	Administrative name for the WebLogic Server domain.
`-domainRoot`	dr	true	none	N/A	Full path to the WebLogic Server domain root.
`-force`	f	false	none	N/A	Overwrite any introspection in the catalog that exists with the same name.
`-name`	n	false	Derived directory name prefixed by component type name.	Any name not already used within the catalog.	Specifies a name by which the introspection output is stored.
`-privateKeyFile`	pkf	false	none	N/A	Private SSH key file on the local machine.
`-remoteCleanup`	rc	false	false	N/A	Remote clean up flag. When set, the remote working directory will be deleted after the operation. Otherwise the directory will not be modified. If set, `remoteUser` and `remoteHost` must be specified as well.
`-remoteHost`	rh	false	none	N/A	Host name or IP address and optional SSH port of the remote machine. If set, `remoteUser` must be specified as well.
`-remoteUser`	ru	false	none	N/A	Name of the SSH user to use for accessing the remote machine. If set, `remoteHost` must be specified as well.
`-remoteWorkingDir`	rwd	false	`/tmp/abRemote_<remote user name>`	N/A	Path on the remote machine to work out of. If set, `remoteUser` and `remoteHost` must be specified as well.
`-skipFileSetCapture`	sf	false	none	N/A	If specified, file sets are not captured for the component during introspection.
`-soaGlobalCP`	sgcp	false	none	Location of optional global configuration plan.	The absolute path to an optional global configuration plan.
`-sudoUser`	su	false	none	User name of sudo user.	Specifies a sudo user. When specified, remote capturing of file sets or introspection will substitute the user (sudo) as the sudo user before running the remote Assembly Builder. If sudoUser is specified, you cannot use the privateKeyFile. That is, sudoUser can only be used when you provide a password.
`-wlsHome`	wh	true	none	WLS Home	Full path to the Oracle WebLogic Server home (usually, `<middleware home>/wlserver_10.3`).

A.1.67.8 Examples

Here are some command examples.

A.1.67.8.1 Successful Introspection: local execution with use of all options

```
% abctl introspectWLS -name myWlsCapture
-wlsHome /ora/mw/wlserver_10.3 -domainRoot /ora/mw/user_projects/domains/MyDomain
-adminUser weblogic
Launching introspection of appliance 'WLS' ...
  Step 1 of 3: WLS dehydration starting. Due to domain template creation this may
take some time
    Step 1 of 15: WlsAssemblyBuilder has started creating the AssemblyBuilder
```

```
      Step 1 of 2: Capturing Node Manager configuration.
      Step 2 of 2: Node Manager capture complete.
    Step 12 of 15: Processor: 10 completed
    Step 15 of 15: WlsAssemblyBuilder has completed the AssemblyBuilder
  Step 2 of 3: WLS Assembly is completed
  Step 3 of 3: WLS dehydration completed
Task is done: DehydrateJob completed
Introspection complete
Storing result in catalog: ...
Introspection stored as 'myWlsCapture' in the catalog
%
```

A.1.67.8.2 Successful Introspection: local execution with all defaults and short names

```
% abctl introspectWLS -adminUser weblogic -wh /ora/mw/wlserver_10.3
-dr /ora/mw/user_projects/domains/MyDomain
Launching introspection of appliance 'WLS' ...
  Step 1 of 3: WLS dehydration starting. Due to domain template creation this may
take some time
    Step 1 of 15: WlsAssemblyBuilder has started creating the AssemblyBuilder
      Step 1 of 2: Capturing Node Manager configuration.
      Step 2 of 2: Node Manager capture complete.
    Step 12 of 15: Processor: 10 completed
    Step 15 of 15: WlsAssemblyBuilder has completed the AssemblyBuilder
  Step 2 of 3: WLS Assembly is completed
  Step 3 of 3: WLS dehydration completed
Task is done: DehydrateJob completed
Introspection complete
Storing result in catalog: '/ora/ab/catalog' ...
Introspection stored as 'WLS-1256089687424' in the catalog
%
```

A.1.67.8.3 Missing -wlsHome Parameter

```
% abctl introspectWLS -domainRoot
/ora/mw/user_projects/domains/MyDomain
Error: missing required parameter 'wlsHome'

Command usage:

introspectWLS [-name string]
   [-remoteHost string] [-remotePort numeric] [-remoteUser string]
   [-remoteWorkingDir path] -wlsHome path -domainRoot path

Try 'abctl help -command introspectWLS' for detailed help of the command.

%
```

A.1.67.8.4 Bad -domainRoot path

```
$ abctl introspectWLS -adminUser weblogic -wlsHome
/scratch/aime1/Oracle/Middleware/wlserver_10.3/ -domainRoot /tmp/foobar -name test
Enter 'Admin Password':
Launching introspection of appliance 'WLS' ...
 Step 1 of 3: Started WLS dehydration (expect delays during domain template
creation)..
Task is done: Dehydration failed with error: The domainRoot specified does not
exist..
Error: OAB-7105: Introspection failed.
Caused by: OAB-50005: The domainRoot specified does not exist.
```

A.1.68 redeployAssemblyInstance

Details for this command follow.

A.1.68.1 Synopsis

```
$ abctl redeployAssemblyInstance -assemblyInstanceId String -connectionName String
[-waitForComplete] [-pollTime String]
```

A.1.68.2 Description

Redeploys an assembly instance, and is equivalent to performing an `undeployAssemblyInstance` followed by a `deployAssemblyInstance`.

A.1.68.3 Options

Table A–66 shows the command options for `redeployAssemblyInstance`.

Table A–66 redeployAssemblyInstance options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assemblyInstanceId`	a	true	none	A string representing the `assemblyInstanceId`.	The identifier of an assembly instance to redeploy.
`-connectionName`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-pollTime`	pt	false	5	A string representing the number of seconds.	Specifies the amount of time to wait for an asynchronous operation to complete successfully.
`-waitForComplete`	w	false	no	N/A	Specifies whether to wait for an asynchronous operation to complete successfully.

A.1.68.4 Examples

Here are some command examples.

A.1.68.4.1 Redeploy Deployment

```
$ abctl redeployAssemblyInstance -assemblyInstanceId MyId
```

A.1.69 registerAssemblyArchive

Details for this command follow.

A.1.69.1 Synopsis

```
$ abctl registerAssemblyArchive -name String [-version String] [-target String]
-connectionName String [-waitForComplete] [-pollTime String]
```

A.1.69.2 Description

Registers an assembly archive in the Deployer.

A.1.69.3 Options

Table A–67 shows the command options for `registerAssemblyArchive`.

Table A–67 registerAssemblyArchive options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-name`	n	true	none	A string representing the name of the assembly archive.	The name of the assembly archive.
`-pollTime`	pt	false	5	A string representing the number of seconds.	Specifies the amount of time to wait for an asynchronous operation to complete successfully.
`-target`	t	false	none	A string representing the name of the target.	The name of the target.
`-version`	v	false	none	A string representing the version of the assembly archive.	The version of the assembly archive.
`-waitForComplete`	w	false	no	N/A	Specifies whether to wait for an asynchronous operation to complete successfully.

A.1.69.4 Examples

Here are some command examples.

A.1.69.4.1 Register Assembly Archive

```
$ abctl registerAssemblyArchive -connectionName MyDeployerConnection -name
TheAssembly -version 1
```

A.1.70 removeAssemblyUsers

Details for this command follow.

A.1.70.1 Synopsis

```
$ abctl removeAssemblyUsers -assembly String -user String... -connectionName
String
```

A.1.70.2 Description

Removes one or more users from an assembly.

A.1.70.3 Options

Table A–69 shows the command options for `removeAssemblyUsers`.

Table A–68 getDefaultTarget options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assembly`	a	false	none	A string representing the name of the assembly archive.	Specifies the assembly to remove users from.
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-user`	u	true	none	A string representing the usernames of the users to remove from an assembly archive.	The usernames of the users to remove from an assembly archive.

A.1.70.4 Examples

Here are some command examples.

A.1.70.4.1 Remove Assembly Users

```
$ abctl removeAssemblyUsers -assembly MyAssembly -user User1 User2
```

A.1.71 removeTargetUsers

Details for this command follow.

A.1.71.1 Synopsis

```
$ abctl removeTargetUsers -user String... -target String -connectionName String
```

A.1.71.2 Description

This command removes a user from the target.

A.1.71.3 Options

Table A–69 shows the command options for `removeTargetUsers`.

Table A–69 getDefaultTarget options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-target`	t	true	none	A string representing the target to add the user to.	The target to add the user to.
`-user`	u	true	none	A string representing the usernames of the users to remove from the target	The usernames of the users to remove from the target.

A.1.71.4 Examples

Here are some command examples.

A.1.71.4.1 Remove Target Users

```
$ abctl removeTargetUsers -user Username -target Targetname
```

A.1.72 restartAssemblyInstance

Details for this command follow.

A.1.72.1 Synopsis

```
$ abctl restartAssemblyInstance -assemblyInstanceId String -connectionName String
[-waitForComplete] [-pollTime String]
```

A.1.72.2 Description

This command restarts an assembly instance, and is equivalent to performing a `stopAssemblyInstance` followed by a `startAssemblyInstance`.

A.1.72.3 Options

Table A–70 shows the command options for `restartAssemblyInstance`.

Table A–70 restartAssemblyInstance options

Name	Alias	Req'd	Default Values	Possible Values	Description
-assemblyInstanceId	a	true	none	A string representing the assemblyInstanceId.	The identifier of an assembly instance to be restarted.
-connectionName	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
-pollTime	t	false	5	A string representing the number of seconds.	Specifies the amount of time to wait for an asynchronous operation to complete successfully.
-waitForComplete	w	false	no	N/A	Specifies whether to wait for an asynchronous operation to complete successfully.

A.1.72.4 Examples

Here are some command examples.

A.1.72.4.1 Restart Deployment

```
$ abctl restartAssemblyInstance -assemblyInstanceId MyId
```

A.1.73 scale

Details for this command follow.

A.1.73.1 Synopsis

```
$ abctl scale -scalingGroupId String -target String -connectionName String
[-waitForComplete] [-pollTime String]
```

A.1.73.2 Description

Scales a scaling group to a new size.

A.1.73.3 Options

Table A–71 shows the command options for `scale`.

Table A–71 scale options

Name	Alias	Req'd	Default Values	Possible Values	Description
-connectionName	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
-pollTime	pt	false	5	A string representing the number of seconds.	Specifies the amount of time to wait for an asynchronous operation to complete successfully.
-scalingGroupId	s	true	none	A string representing the scalingGroupId.	The scalingGroupId of a scalingGroup.
-target	t	true	none	A string representing the new target.	The new value to scale to.
-waitForComplete	w	false	no	N/A	Specifies whether to wait for an asynchronous operation to complete successfully.

A.1.73.4 Examples

Here are some command examples.

A.1.73.4.1 Scale a Scaling Group

```
$ abctl scale -scalingGroupId FOO -target 4
```

A.1.74 setDefaultTarget

Details for this command follow.

A.1.74.1 Synopsis

```
$ abctl setDefaultTarget -name String -connectionName String
```

A.1.74.2 Description

This command sets a target as the default.

A.1.74.3 Options

Table A–72 shows the command options for `setDefaultTarget`.

Table A–72 setDefaultTarget options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-name`	n	true	none	A string representing the name of the target.	`The name of the target.`

A.1.74.4 Examples

Here are some command examples.

A.1.74.4.1 Set Default Target

```
$ abctl setDefaultTarget -name MyTarget
```

A.1.75 startAssemblyInstance

Details for this command follow.

A.1.75.1 Synopsis

```
$ abctl startAssemblyInstance -assemblyInstanceId String -connectionName String
[-waitForComplete] [-pollTime String]
```

A.1.75.2 Description

Starts an assembly instance.

A.1.75.3 Options

Table A–73 shows the command options for `startAssemblyInstance`.

Table A–73 startAssemblyInstance options

Name	Alias	Req'd	Default Values	Possible Values	Description
-assemblyInstanceId	a	true	none	A string representing the assemblyInstanceId.	The identifier of an assembly instance to be started.
-connectionName	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
-pollTime	pt	false	5	A string representing the number of seconds.	Specifies the amount of time to wait for an asynchronous operation to complete successfully.
-waitForComplete	w	false	no	N/A	Specifies whether to wait for an asynchronous operation to complete successfully.

A.1.75.4 Examples

Here are some command examples.

A.1.75.4.1 Start Assembly Instance

```
$ abctl startAssemblyInstance
```

A.1.76 stopAssemblyInstance

Details for this command follow.

A.1.76.1 Synopsis

```
$ abctl stopAssemblyInstance -assemblyInstanceId String [-force] -connectionName
String [-waitForComplete] [-pollTime String]
```

A.1.76.2 Description

This command stops a deployment for an assembly instance.

A.1.76.3 Options

Table A–74 shows the command options for `stopAssemblyInstance`.

Table A–74 stopAssemblyInstance options

Name	Alias	Req'd	Default Values	Possible Values	Description
-assemblyInstanceId	a	true	none	A string representing the assemblyInstanceId.	The identifier of an assembly instance to be stopped.
-connectionName	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
-force	f	false	none	True/false.	Flag to indicate if local cleanup should be done even if the resource manager is not available.
-pollTime	pt	false	5	A string representing the number of seconds.	Specifies the amount of time to wait for an asynchronous operation to complete successfully.
-waitForComplete	w	false	no	N/A	Specifies whether to wait for an asynchronous operation to complete successfully.

A.1.76.4 Examples

Here are some command examples.

A.1.76.4.1 Stop Assembly Instance

```
$ abctl stopAssemblyInstance
```

A.1.77 undeployAssemblyInstance

Details for this command follow.

A.1.77.1 Synopsis

```
$ abctl undeployAssemblyInstance -assemblyInstanceId String -connectionName String
[-waitForComplete] [-pollTime String]
```

A.1.77.2 Description

Undeploys an assembly instance.

A.1.77.3 Options

Table A–75 shows the command options for `undeployAssemblyInstance`.

Table A–75 undeployAssemblyInstance options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-assemblyInstanceId`	d	true	none	A string representing the `assemblyInstanceId`.	The identifier of an assembly instance to undeploy.
`-connectionName`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-pollTime`	pt	false	5	A string representing the number of seconds.	Specifies the amount of time to wait for an asynchronous operation to complete successfully.
`-waitForComplete`	w	false	no	N/A	Specifies whether to wait for an asynchronous operation to complete successfully.

A.1.77.4 Examples

Here are some command examples.

A.1.77.4.1 Undeploy Assembly Instance

```
$ abctl undeployAssemblyInstance -assemblyInstanceId MyId
```

A.1.78 unregisterAssemblyArchive

Details for this command follow.

A.1.78.1 Synopsis

```
$ abctl unregisterAssemblyArchive -name String [-version String] [-target String]
-connectionName String [-waitForComplete] [-pollTime String]
```

A.1.78.2 Description

This command unregisters an assembly from the Deployer.

A.1.78.3 Options

Table A–76 shows the command options for `unregisterAssemblyArchive`.

Table A–76 unregisterAssemblyArchive options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-name`	n	true	none	A string representing the name of the assembly archive.	The name of the assembly archive.
`-pollTime`	pt	false	5	A string representing the number of seconds.	Specifies the amount of time to wait for an asynchronous operation to complete successfully.
`-target`	t	false	none	A string representing the name of the target.	The name of the target.
`-version`	v	false	none	A string representing the version of the assembly archive.	The version of the assembly archive.
`-waitForComplete`	w	false	no	N/A	Specifies whether to wait for an asynchronous operation to complete successfully.

A.1.78.4 Examples

Here are some command examples.

A.1.78.4.1 Unregister Assembly Archive

```
$ abctl unregisterAssemblyArchive -name TheAssembly -version 1
```

A.1.79 updateAssemblyArchive

Details for this command follow.

A.1.79.1 Synopsis

```
$ abctl updateAssemblyArchive -name String -version String -description String
-connectionName String
```

A.1.79.2 Description

This command updates the description (attributes) of an assembly archive in the Deployer.

A.1.79.3 Options

Table A–77 shows the command options for `updateAssemblyArchive`.

Table A–77 updateAssemblyArchive options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-description`	d	true	none	A string representing the description of the assembly archive.	The description of the assembly archive.
`-name`	n	true	none	A string representing the name of the assembly archive.	The name of the assembly archive.
`-version`	v	true	none	A string representing the version of the assembly archive.	The version of the assembly archive.

A.1.79.4 Examples

Here are some command examples.

A.1.79.4.1 Update Assembly Archive

```
$ abctl updateAssemblyArchive -name MyAssembly -version 1 -description
NewDescription
```

A.1.80 updateTarget

Details for this command follow.

A.1.80.1 Synopsis

```
$ abctl updateTarget -name String [-properties String...] -connectionName String
```

A.1.80.2 Description

Updates one or more property values. This command is enabled for Oracle VM targets, but disabled for Oracle Exalogic. The single target in an Oracle Exalogic installation is preconfigured and cannot be changed.

A.1.80.3 Options

Table A–78 shows the command options for `updateTarget`.

Table A–78 updateTarget options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connection Name`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-name`	n	true	none	A string representing the name of the target.	The name of the target.
`-properties`	p	false	none	A string representing the properties and values to update.	A set of key=value pairs representing the property and its new value.

A.1.80.4 Examples

Here are some command examples.

A.1.80.4.1 Update Target

```
$ abctl updateTarget -name MyTarget -properties prop=newvalue
```

A.1.81 uploadAssemblyArchive

Details for this command follow.

A.1.81.1 Synopsis

```
$ abctl uploadAssemblyArchive -fileName Path -name String [-description String]
-connectionName String
```

A.1.81.2 Description

Uploads an assembly archive to Oracle Virtual Assembly Builder Deployer.

A.1.81.3 Options

Table A–79 shows the command options for `uploadAssemblyArchive`.

Table A–79 uploadAssemblyArchive options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-connectionName`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.
`-fileName`	fn	true	none	A string representing the file path to the assembly archive on disk.	Uploads an assembly archive to the Deployer.
`-name`	n	true	none	A string representing the name of the assembly archive.	The name of the assembly archive.
`-description`	d	false	none	A string representing the description of the assembly archive.	The description of the assembly archive.

A.1.81.4 Examples

Here are some command examples.

A.1.81.4.1 Upload Assembly Archive

```
$ abctl uploadAssemblyArchive -fileName=c:/mySite.ova -name TheAssembly -version 1
```

A.1.82 uploadAssemblyResources

Details for this command follow.

A.1.82.1 Synopsis

```
$ abctl uploadAssemblyResources -fileName Path -assemblyName String -version
String -connectionName String [-append]
```

A.1.82.2 Description

Uploads an assembly resources file to associate with a specific version of an assembly. The resource zip file is uploaded and extracted into the repository.

The `uploadAssemblyResources` command is controlled by a security policy. A resources file may or may not contain scripts. If the resource file does not contain scripts, a user on the assembly access list can run the command. If the resource file does contain scripts, only the Cloud Admin user is allowed to run the command, to prevent a malicious attack.

When including scripts in the resources files, the lifecycle names that are supported are: `pre-deploy`, `post-deploy`, `deployer-pre-app-config`, `deployer-post-app-config`, `deployer-pre-vm-start`, `deployer-post-vm-start`, `deployer-pre-vm-stop`, `deployer-post-vm-stop`, `pre-undeploy`, `post-undeploy`. You can create corresponding script folder names.

The following is a sample resource zip file containing scripts:

```
unzip ../myResources.zip
Archive:  ../myResources.zip
    creating: disks/
   inflating: disks/test1.iso
   inflating: disks/test2.iso
    creating: scripts.d/
```

```
 creating: scripts.d/pre-deploy.d/
inflating: scripts.d/pre-deploy.d/00script.sh
inflating: scripts.d/pre-deploy.d/01script.sh
 creating: scripts.d/post-deploy.d/
inflating: scripts.d/post-deploy.d/00script.sh
inflating: scripts.d/post-deploy.d/01script.sh
 creating: scripts.d/deployer-pre-vm-stop.d/
inflating: scripts.d/deployer-pre-vm-stop.d/00script.sh
inflating: scripts.d/deployer-pre-vm-stop.d/01script.sh
 creating: scripts.d/post-undeploy.d/
inflating: scripts.d/post-undeploy.d/00script.sh
inflating: scripts.d/post-undeploy.d/01script.sh
```

A.1.82.3 Options

Table A–80 shows the command options for `uploadAssemblyResources`.

Table A–80 uploadAssemblyResources options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-fileName`	fn	true	none	A string representing the file path to the assembly resources files on disk.	Uploads an assembly resources file to the Deployer.
`-assemblyName`	n	true	none	A string representing the name of the assembly.	The name of the assembly.
`-version`	v	false	none	A string representing the assembly version.	The assembly version.
`-append`	a	false	none	A flag, that if set, appends the assembly resources file upload.	If set, the assembly resources file upload is appended.
`-connectionName`	c	true	none	A string representing the name of the connection to the Deployer.	The name of a connection to the Deployer.

A.1.82.4 Examples

Here are some command examples.

A.1.82.4.1 Upload Assembly Resources File

```
$ abctl uploadAssemblyResources -assemblyName myAssembly -version 1 -fileName
resources.zip -connectionName myConnection

Upload File Size: 2,708
100% Complete
Assembly resources zip has been uploaded to associate with assembly myAssembly,
version 1.
```

A.1.83 uploadEMAssemblyArchive

Details for this command follow.

A.1.83.1 Synopsis

```
$ abctl uploadEMAssemblyArchive -name String -[description String]
```

A.1.83.2 Description

Uploads an assembly archive to the Enterprise Manager Software Library. The assembly can only be a top-level assembly, and the assembly archive must be created for the assembly.

A.1.83.3 Options

Table A–81 shows the command options for `uploadEMAssemblyArchive`.

Table A–81 uploadEMAssemblyArchive options

Name	Alias	Req'd	Default Values	Possible Values	Description
`-name`	n	true	none	Top level appliance or assembly in the catalog.	Name of a top level appliance or assembly in the catalog.
`-description`	d	false	none	Textual description.	A description of the assembly and assembly archive.

A.1.83.4 Examples

Here is a command example.

A.1.83.4.1 uploadEMAssemblyArchive

```
% abctl uploadEMAssemblyArchive -name archiveName -description "my assembly
archive"
```

A.1.84 version

Details for this command follow.

A.1.84.1 Synopsis

```
$ abctl version
```

A.1.84.2 Description

Displays the Oracle Virtual Assembly Builder specification version.

A.1.84.3 Example

$ abctl version

A.2 Help

Details for this command follow.

A.2.1 Synopsis

```
$ abctl help [-command String] [-category] [-usage]
```

A.2.2 Description

With no parameters, the `help` command displays a list of help categories that you can query to obtain the list of available commands along with the brief description of each command.

When the -all option is specified, the help command lists all available commands with a short description of each command. When a command name is specified, then detailed Help about the specified command is printed. When a command name is specified and the -usage parameter is specified, just the synopsis (argument usage) is printed.

When the -category option is specified, the help command lists the commands available under that category. Commands may appear under more than one category.

A.2.3 Options

Table A–82 shows the command options for help.

Table A–82 help options

Name	Alias	Req'd	Default Values	Possible Values	Description
-command	c	false	none	Name of a command.	Name of one of the commands listed when help is invoked without arguments. If specified, then detailed help information about the specified command is provided.
-usage	u	false	none		Not valid when the -command parameter is not specified. When this flag is specified only synopsis details are provided about the corresponding specified command.
-category	c	false	none		When this flag is specified, the help command lists the commands available under the specified category.
-quiet	q	false	none	N/A	By default, the command shows detailed progress/success messages. If -quiet is set, the command turns off verbose mode and shows only one or two progress/success messages.

A.2.4 Examples

Here are some command examples.

A.2.4.1 No Arguments

```
$ abctl help

Usage: abctl command [options]

Command                 Description
-------------------------------------------------------------------------------
abctl help all              List all commands.
abctl help catalog          Manage appliances and assemblies.
abctl help deployer         Setup and manage OVAB Deployer.
abctl help deployments      Deploy and manage assembly deployments.
abctl help EM               Manage assemblies in EM Software Library.
abctl help general          General help commands.
abctl help introspection    Capture product installations.
abctl help targets          Configure and manage deployment targets.

Try "abctl help -command cmd_name" for detailed help of a specific command.
```

A.2.4.2 Specifying Help on a Category of Commands

```
$ abctl help catalog

Usage: abctl command [options]

Command                Description
-------------------------------------------------------------------------------
addToAssembly           Adds an appliance or assembly to an assembly.
connectEndpoints        Create a new connection between two endpoints.
createAssembly          Creates a new empty assembly.
createAssemblyArchive   Creates an assembly archive.
createExternalResources Creates and connects external resources for
                         unconnected outputs.
createTemplate          Creates an appliance template.
delete                  Deletes an appliance or assembly.
describeCatalog         Lists appliances and assemblies in the catalog.
describeEndpoints       Lists endpoints of an appliance or assembly.
export                  Exports an appliance or assembly to disk.
import                  Imports an appliance or assembly from disk.
importExternalTemplate  Imports a VM template as an external appliance.

Try "abctl help <command>" for detailed help of a specific command.
```

A.2.4.3 Help with a -command parameter specified

```
$ abctl help -command captureFileSets

NAME

captureFileSets

SYNOPSIS

captureFileSets -name String [-remoteHost String] [-remoteUser String] \
[-sudoUser String] [-remoteWorkingDir Path] [-remoteCleanup] \ [-privateKeyFile
Path] [-quiet] [-force]

DESCRIPTION

Creates file sets for specified appliance or assembly.

OPTIONS

Name: remoteHost
Aliases: rh
Type: String
Required: false
Value description: String
Default value:
Possible values:
Description: Host name or IP address and optional SSH port of the remote
    machine.  If specified, the remoteUser must be specified as well.

Name: remoteUser
Aliases: ru
Type: String
```

```
Required: false
Value description: String
Default value:
Possible values:
Description: Name of the SSH user to use for accessing the remote machine.  If
    specified, the remoteHost must be specified as well.

...

EXAMPLES

abctl captureFileSets -name myOhs -force
```

A.2.4.4 Help with a -command parameter specified and -usage flag specified

```
$ abctl help -command captureFileSets -usage

Command usage:

captureFileSets -name String [-remoteHost String] [-remoteUser String] \
[-sudoUser String] [-remoteWorkingDir Path] [-remoteCleanup] \ [-privateKeyFile
Path] [-quiet] [-force]

Try 'abctl help -command captureFileSets' for detailed help of the command.
```

B

Oracle Virtual Assembly Builder Introspection Plug-ins

This appendix describes the plug-ins for appliances that Oracle Virtual Assembly Builder can introspect:

B.1 Oracle WebLogic Server Plug-in

The Oracle WebLogic Server introspection plug-in examines a single Oracle WebLogic Server domain and the Oracle Middleware Home it resides in. The domain specified and its Middleware Home are captured.

B.1.1 Versions Supported

This plug-in supports versions 11gR1 11.1.1.2 and later.

B.1.2 Oracle WebLogic Server Introspection Parameters

Table B–1 lists the introspection parameters for Oracle WebLogic Server:

Table B–1 Oracle WebLogic Server Plug-in Introspection Parameters

Parameter	Description
domainRoot	The fully qualified path to the domain you want to introspect. This should be the directory that contains the 'config' directory.

Table B–1 (Cont.) Oracle WebLogic Server Plug-in Introspection Parameters

Parameter	Description
wlsHome	The fully qualified path to the WLS Home server directory. For example, `/u01/oracle/middleware/wlserver_10.3`.
adminUser	The administrative user for the WLS domain.
adminPassword	The password for the administrative user specified for the adminUser property.

B.1.3 Reference System Prerequisites

The AdminServer for the domain must be running and introspection must target the host where the AdminServer is running.

B.1.4 Requirements

The following requirements apply to Oracle WebLogic Server:

B.1.4.1 Oracle WebLogic Server Domain Requirements

You must ensure that any Oracle WebLogic Server domain being introspected is configured to be editable. This allows edits to be performed successfully during deployment. For more information on configuring your Oracle WebLogic Server, see your product documentation.

B.1.4.1.1 LDAP Provider Requirements Oracle recommends the use of Oracle Internet Directory as the LDAP provider on the reference system, not file-based LDAP. File-based LDAP cannot work properly in a deployed system due to synchronization issues.

B.1.4.2 Requirement for Remote User Specified for Remote Introspection of Oracle WebLogic Server

The remote user specified for remote introspection of Oracle WebLogic Server must be able to access files created by the user that owns the Oracle WebLogic Server process. When possible it is recommended that the remote user specified be the same as the user who owns the Oracle WebLogic Server process.

B.1.4.3 Requirements for SSL Certificate and Hostname Validation

You must use only a demo certificate, with hostname validation turned off.

B.1.4.4 Requirement to Update Applications Accessing Web Services

For each application that accesses a Web service hosted on the Oracle WebLogic Server reference system, you must update the application to access the Web service WSDL on the new Oracle VM host, and then redeploy the application through Oracle WebLogic Server administration tools, such as Admin Console or wlst, to the Oracle VM Oracle WebLogic Server environment.

B.1.4.5 Requirement Not to Create Templates on Individual Servers

You must not create a template on an individual server in Oracle WebLogic Server. Such templates cannot be deployed because they lack certain deployment artifacts (the domain template jar in content, and data at the assembly level).

B.1.4.6 Requirement to Specify Admin URL When Managed Server Not Running

If you want to perform manual starts from the context of the Guest-OS, you must manually modify the StartManagedServer.sh script to provide the correct Admin Server URL (Admin Server hostname). This is required to provide the default admin URL the correct value (the machine name of the Admin Server is not known at the time of template creation).

You can still start or stop the server through the node manager in Admin Console.

B.1.5 Resulting Artifact Type

An atomic assembly which contains an appliance for the AdminServer and appliances for any clusters found and any stand-alone (non-clustered) managed servers found. One appliance is created for a cluster regardless of the number of managed servers in that cluster. The Oracle WebLogic Server plug-in presumes that every managed server in a cluster is configured identically. The number of servers in the cluster is saved as 'scale out' information in the appliance metadata, as are the names of the servers in the cluster.

Note: An atomic assembly cannot be edited to add or remove appliances. To wire other appliances to an atomic Oracle WebLogic Server assembly a non-atomic assembly must be created and the Oracle WebLogic Server assembly must be added to the non-atomic assembly.

B.1.6 Wiring

Inputs will be created on the Oracle WebLogic Server assembly for all the channels the servers in the domain are listening on. Typically Oracle HTTP Server outputs would be connected to the Oracle WebLogic Server inputs.

Outputs will be created on the Oracle WebLogic Server assembly for the following types of configuration found:

- JDBC
- LDAP
- JMS Message Bridges
- Foreign JMS

These outputs must all be connected to either an external resource or to an appliance before deployment. The description on the output and the protocol supported by the output will give hints about the type of appliance to connect the output to.

B.1.7 Wiring Properties

All input endpoints have two editable properties - `port` and `description`, and one non editable property - a list of `protocols`. The `protocols` indicate what sort of outputs can be connected to the input.

All output endpoints have one editable property - `description`, and two non-editable properties - `protocol` and `singleton`. The protocol indicates what sort of input can be connected to the output. `Singleton` indicates what sort of appliance the output can be connected to. If `singleton` is true, the output can only be connected to an input on an appliance that has a scalability absolute max value of 1.

The following properties are specific to Oracle WebLogic Server endpoints:

Table B–2 describes common Oracle WebLogic Server appliance input system properties:

Table B–2 Common Oracle WebLogic Server Appliance Input System Properties

Name	Type	Req'd	Default	Description
originalBindAddresses	String	false	none	The original address of the system that was introspected.
originalDefaultHostname	String	false	none	The original hostname of the system that was introspected. (for example, "example.com").

Table B–3 describes common Oracle WebLogic Server appliance input user properties:

Table B–3 Common Oracle WebLogic Server Appliance Input User Properties

Name	Type	Req'd	Default	Description
keepLocalHost	Boolean	false	none	If this input was originally bound to localhost explicitly, this property will exist and be set to true. Connections should not be made to this input if this property exists and its value is not overridden to false.
readymetric-naming-password	String	false	none	The password to use for the connection made to the server when doing the ready metric check.
readymetric-naming-protocol	String	false	none	Optional protocol you can specify for naming connections used for the ready metric check (for example, "iiop").
readymetric-naming-user	String	false	none	The user to use for the connection made to the server when doing the ready metric check (for example, "weblogic").
readymetric-server-protocol	String	false	none	The protocol to use for the connection made to the server when doing the ready metric check (for example, "iiop").

Table B–4 describes Admin Server appliance input system properties:

Table B–4 Admin Server Appliance Input System Properties

Name	Type	Req'd	Default	Description
admin-password	String	true	none	The admin user's password.
admin-username	String	true	none	The admin user name for connecting to the Admin server (for example, "weblogic").

Table B–5 through Table B–8 describes Admin Server appliance output user properties for JDBC, foreign JMS, JMS message bridge, and LDAP.

Table B–5 describes Admin Server appliance output user and system properties for JDBC. The `password` and `username` properties are user properties, and `original-url` is a system property.

Table B–5 Admin Server Appliance Output Properties: JDBC

Name	Type	Req'd	Default	Description
password	String	false	<empty>	The password for the user needed for the data source connection.
username	String	false	none	The user needed for the data source connection. The value will be the original user for the data source connection.
original-url	String	false	none	The original JDBC URL from the introspected Oracle WebLogic Server domain. (for example, "jdbc:oracle:thin:@adc2100927.example.com:1521:orcl").

Table B–6 describes Admin Server appliance output user properties for foreign JMS:

Table B–6 Admin Server Appliance Output Properties: Foreign JMS

Name	Type	Req'd	Default	Description
original-connection-url	String	false	none	The original URL for the foreign JMS server.

Table B–7 describes Admin Server appliance output system properties for JMS message bridge:

Table B–7 Admin Server Appliance Output Properties: JMS Message Bridge

Name	Type	Req'd	Default	Description
original-url	String	false	none	The original URL for the JMS messaging bridge server.
original-username	String	false	none	The original username for the JMS messaging bridge server.
original-password	String	false	none	The original password for the JMS messaging bridge server, encrypted.

Table B–8 describes Admin Server appliance output system properties for LDAP:

Table B–8 Admin Server Appliance Output Properties: LDAP

Name	Type	Req'd	Default	Description
original-name	String	false	none	The original name for the LDAP security provider.
original-host	String	false	none	The original host for the LDAP security provider.
original-port	String	false	none	The original port for the LDAP security provider.
original-user	String	false	none	The original user for the LDAP security provider.

B.1.8 Oracle WebLogic Server Appliance Properties

This section discusses the following properties for assemblies with an Oracle WebLogic Server appliance. Those properties include assembly-level properties,

properties on the inputs and outputs of each application, and properties of the appliances themselves. This section contains the following subsections:

- Section B.1.8.1, "Assembly-Level System Properties"
- Section B.1.8.2, "Properties Common to Admin and Managed Server Appliances"
- Section B.1.8.3, "Admin Server Appliance Properties"

B.1.8.1 Assembly-Level System Properties

Table B–9 describes assembly-level system properties:

Table B–9 Assembly-level System Properties

Name	Type	Req'd	Default	Description
admin-password	String	true	none	The admin user password for the domain.
admin-username	String	true	none	The admin user for the domain (for example, "weblogic").
admsvr-jmx-input	String	true	none	Indicates what input on the AdminServer appliance should be used when making JMX connections (for example, "Default").
admsvr-jmx-protocol	String	true	none	The protocol to use when making a JMX connection to the Admin Server (for example, "iiop").
domain-name	String	false	none	The domain name of the domain that was introspected (for example, "test_ domain").
usesOracleHomes	boolean	true	none	Indicates that this is not a core Oracle Oracle WebLogic Server installation and as such has an OracleHome associated with it. This will be true for SOA and WebCenter domains. Allowable values are true and false.

B.1.8.2 Properties Common to Admin and Managed Server Appliances

The following information describes properties common to admin and managed server appliances.

Table B–10 describes common Oracle WebLogic Server appliance system properties:

Table B–10 Common Oracle WebLogic Server Appliance System Properties

Name	Type	Req'd	Default	Description
capture.hostname	String	true	none	The host name where the introspection was performed (for example, “example.com”).
capture.is64bit	boolean	true	none	Indicates if the system where introspection was performed is a 64-bit system.
capture.osarch	String	true	none	The architecture of the system that was introspected (for example, “i386”).
capture.osname	String	true	none	The operating system name of the system that was introspected (for example, “Linux”).

Table B–10 (Cont.) Common Oracle WebLogic Server Appliance System Properties

Name	Type	Req'd	Default	Description
capture.time	String	true	none	The time the introspection was performed (for example, "1269628142430").
domain-name	String	false	none	The domain name of the system that was introspected.
admin-input-name	String	false	none	The name of the input for administrative traffic on the admin server.
admin-input-protocol	String	false	none	The protocol to use when connecting to the admin server (for example, "iiop").
isAdminserver	String	false	none	True for the admin server, false otherwise.
NodeManagerType	String	true	none	The type of node manager machine definition to create (for example, "SSL").
server-names	String	false	none	A list of server names for the appliance (for example, "AdminServer"). For a cluster appliance there will most likely be more than one server name in the list.

Table B–11 describes common Oracle WebLogic Server appliance user properties:

Table B–11 Common Oracle WebLogic Server Appliance User Properties

Name	Type	Req'd	Default	Description
NodeManagerPort	Integer	true	5556	The port the node manager should listen on (for example, 5556). This will only be present if node manager was found to be configured on the reference system.
readymetric-attribute-compare-type	String	false	EQUALS	The comparison to make between the readymetric-attribute's value and the value specified for the property `readymetric-attribute-value`. Valid values are EQUALS, LESSER_THAN, GREATER_THAN, LESSER_THAN_OR_EQUAL, and GREATER_THAN_OR_EQUAL.
readymetric-attribute-name	String	false	State	The MBean attribute to check.
readymetric-attribute-type	String	false	STRING	The type of the MBean attribute. Valid values (but specific to the attribute being examined) are STRING, INTEGER, SHORT, LONG, DOUBLE, FLOAT, and BOOLEAN.
readymetric-attribute-value	String	false	RUNNING	The value the property `readymetric-attribute-name` must have for the check to be considered successful.

Table B–11 (Cont.) Common Oracle WebLogic Server Appliance User Properties

Name	Type	Req'd	Default	Description
readymetric-instance-name-0	String	false	com.bea:Name=AdminServer,Type=ServerRuntime	The instance name to use for the JMX ready metric check.
readymetric-max-wait-period	String	false	600	The maximum time in seconds to wait for a successful ready metric check.
readymetric-naming-input	String	false	none	The input to use for the ready metric check (for example, "Default").
readymetric-polling-period	String	false	none	The time between connection attempts, in seconds, for the ready metric check.
readymetric-server-input	String	false	none	The input to use for the ready metric check (for example, "Default").
readymetric-trust-store-0	String	false	none	The location of the trust store to use if the ready metric check is using an SSL enabled port.
readymetric-type	String	false	JMX	The type of ready metric to use for the appliance.
readymetric-verify	String	false	true	If this property is set to true the ready metric check will be performed. Otherwise it will be skipped.
useTemplate	String	false	OEL	Specifies the template type to use by default when creating a template for the appliance.

B.1.8.3 Admin Server Appliance Properties

Table B–12 describes Admin Server appliance system properties:

Table B–12 Admin Server Appliance System Properties

Name	Type	Req'd	Default	Description
admin-input-name	String	false	none	The input to use for connecting to the Admin server admin-input-protocol (for example, "Default").
admin-input-protocol	String	false	none	The protocol to use for connecting to the Admin server (for example, "http").

Table B–13 describes Admin Server appliance user properties:

Table B–13 Admin Server Appliance User Properties

Name	Type	Req'd	Default	Description
<cluster name>-cluster-address	String	false	<empty>	The cluster address for the cluster named by the first part of the property name.
<cluster name>-frontend-host	String	false	<empty>	The front-end host for the cluster named by the first part of the property name.
<cluster name>-frontend-http-port	String	false	<empty>	The non-secure front-end port for the cluster named by the first part of the property name.

Table B–13 (Cont.) Admin Server Appliance User Properties

Name	Type	Req'd	Default	Description
<cluster name>-frontend-https-port	String	false	<empty>	The secure front-end port for the cluster named by the first part of the property name.

B.1.9 Extensions of the Plug-in

See Section B.2, "Oracle Coherence*Web Extension" and Section B.3, "Oracle Forms and Reports Extensions".

B.1.10 Supported Template Types

The supported template type is Oracle Enterprise Linux (OEL).

B.2 Oracle Coherence*Web Extension

The Oracle Coherence*Web introspection extension extends the functionality of the WLS Introspector. It examines the configuration of Coherence cache clusters and servers configured as part of an Oracle WebLogic Server domain.

B.2.1 Versions Supported

The plug-in extension works with Oracle WebLogic Server 11gR1 version 11.1.1.4.0, which includes Coherence 3.6.

B.2.2 Oracle Coherence*Web Introspection Parameters

There are no additional parameters required beyond those needed by Oracle WebLogic Server.

B.2.3 Reference System Prerequisites

There are no additional prerequisites beyond those defined by Oracle WebLogic Server.

B.2.4 Requirements

Oracle Coherence*Web has the following requirements:

B.2.4.1 Deployment Model Requirement

The plug-in extension requires you to use an out-of-process deployment model for Oracle Coherence*Web, in which storage-enabled cache servers are executed as separate processes rather than running within Oracle WebLogic Server.

B.2.4.2 Requirement to Manually Update Custom Cluster Configuration Files

The plug-in extension examines Oracle Coherence*Web configuration defined through the Oracle WebLogic Server console and Oracle WebLogic Server mBeans (including WLST). It does not examine or modify custom cluster configuration files such as tangosol-coherence-override.xml. Custom cluster configuration files are passed through to the deployed environment, but no configuration changes are made to those files to reflect the deployed environment.

After deployment, ensure that you make appropriate manual configuration changes to any custom cluster configuration files.

B.2.5 Resulting Artifact Type

For each Coherence cluster that is defined in an introspected Oracle WebLogic Server domain, the plug-in extension creates a new appliance within the atomic Oracle WebLogic Server assembly.

B.2.6 Wiring

No wiring can be performed for Coherence cluster appliances. Each cluster appliance has a fixed, pre-defined connection to the domain's AdminServer, which is used at rehydration time to modify the cluster's configuration.

B.2.7 Wiring

None.

B.2.8 Oracle Coherence*Web Appliance Properties

Each Oracle Coherence*Web cluster appliances has the following system and user properties:

Table B–14 describes Oracle Coherence*Web cluster appliance system properties:

Table B–14 Oracle Coherence*Web Appliance System Properties

Name	Type	Req'd	Default	Description
cache-servers	String	false	none	A list of the cache servers that are part of the cluster.
targets	String	false	none	A list of WLS managed servers that are part of the cluster.
<cacheserver>.node-manager-type	String	false	none	For each cache server in the above list, there is a property indicating the node manager type.
well-known-addresses	String	false	none	A list of well-known-addresses defined for the cluster. If no well-known-address are defined for this cluster (meaning it uses multicast), then this property will not be present.
<wellknownaddress>.server	String	false	none	For each of the well-known-addresses in the above list, there is a property indicating which cache server the well known address maps to (based on matching listen address and port information).

Table B–15 describes Oracle Coherence*Web cluster appliance user properties:

Table B–15 Oracle Coherence*Web Appliance User Properties

Name	Type	Req'd	Default	Description
<cacheserver>.node-manager-port	String	false	none	For each of the cache servers in the cluster, the node manager port is listed and may be modified by the user.

*Table B–15 (Cont.) Oracle Coherence*Web Appliance User Properties*

Name	Type	Req'd	Default	Description
<cacheserver>.unicast-listen-port	String	false	none	For each of the cache servers in the cluster, the unicast listen port of that server is listed and may be modified by the user.
multicast-listen-address	String	false	none	The cluster-wide multicast listen address. If one or more well-known-addresses are listed (meaning the cluster uses unicast for cluster discovery), then this multicast property will not be present.
multicast-listen-port	String	false	none	The cluster-wide multicast listen port. If one or more well-known-addresses are listed (meaning the cluster uses unicast for cluster discovery), then this multicast property will not be present.
unicast-listen-port	String	false	none	The default unicast listen port for the cluster. This value is used by any cache servers that do not have a unicast listen port defined, as well as by any WLS managed servers that join the cluster.
<wellknownaddress>.server	String	true	none	If any of the defined well known addresses could not be correlated with a cache server (based on matching listen address and port information), they will be listed here, and the user is responsible for specifying a cache server name to be used as the well known address. This property is mandatory, meaning it must be specified either as an appliance property or via a deployment plan.

B.2.9 Supported Template Types

The supported template type is Oracle Enterprise Linux (OEL).

B.3 Oracle Forms and Reports Extensions

The Oracle Forms and Reports introspection extensions extend the functionality of the Oracle WebLogic Server introspection plug-in. These examine Forms and Reports applications and configuration residing in the Forms WebLogic servers, Reports WebLogic servers and Oracle Instance.

B.3.1 Versions Supported

The extensions support introspecting only in the following scenarios:

- Oracle Forms and Reports 11*g* Release 2.
- Only Oracle Access Manager 11*g* Release 1 (11.1.1.5) as the Identity Management/Access Control Server with WebGate as the access client is supported when Forms and Reports applications are to be protected by single sign-on.
- Oracle Access Manager with `mod_osso` as the access client or Oracle Single Sign-On Server with `mod_osso` as the access client is not supported.

B.3.2 Introspection Parameters

There are no additional parameters required beyond those needed by Oracle WebLogic Server.

B.3.3 Reference System Prerequisites

In addition to the reference system prerequisites mentioned for the Oracle WebLogic Server plug-in, create the following empty files on the reference system.

- `$ORACLE_HOME/precomp/public/bnddsc.for`
- `$ORACLE_HOME/precomp/public/oraca.for`
- `$ORACLE_HOME/precomp/public/seldsc.for`
- `$ORACLE_HOME/precomp/public/sqlca.for`

> **Note:** `$ORACLE_HOME` refers to the Forms and Report Oracle Home.

B.3.3.1 Adding Partner Application Registation Utility (Web Tier on a Separate Node)

If the Web tier is set up on a separate node for the Forms and Reports installation on the reference system, copy the partner app registration utility (rreg-toolkit.jar) from the Forms and Reports installation located in the `ORACLE_HOME/oam/server/rreg/client` directory to the `ORACLE_HOME/oam/server/rreg/client` directory on the Web tier before running the introspection on the Web tier node.

B.3.4 Requirements

In addition to the requirements mentioned for Oracle WebLogic Server Plug-in, following requirements must be met:

B.3.4.1 Managed Servers Requirement

Forms and Reports managed servers must be up and running before starting the introspection.

B.3.4.2 Supported Topologies

The Forms and Reports extensions support:

- Only Forms and Reports managed servers (standalone or part of the cluster) that are on the same machine as the Admin Server will be examined and captured.
- In case of an Expand Cluster configuration scenario, on the reference system if there are multiple managed servers in the cluster_forms cluster, the configuration from the WLS_FORMS managed server will be replicated to all the Forms managed servers in the virtual deployed environment. Similarly, the configuration from the WLS_REPORTS managed server will be replicated to all the Reports managed servers in the virtual deployed environment.

B.3.4.3 Unsupported Topology

The Forms and Reports extensions do not support:

- Introspection of remote Forms and Reports managed servers: servers that are created on a different machine than the Admin Server machine.

- Forms and Reports managed servers created through the Remote Extend Domain configuration scenario: the case where the domain pre-exists and the Forms and/or Reports managed servers are added later through the Extend Domain configuration scenario on a different machine than the Admin Server machine.

B.3.4.4 Requirement to Support Scale Out of Deployed Assembly

After deploying the assembly, you can add new managed servers to Forms and Reports WLS clusters through the "scale" operation. But a cluster can only be scaled out up to the "max" scalability property which is limited to the number of managed servers that were present in the cluster in the reference system at introspection time. To account for future scale out, in the reference system, you should temporarily add additional managed servers to the Forms cluster (cluster_forms) and Reports cluster (cluster_reports) using the WLS Admin Console or WLST before that WLS domain is introspected.

These additional managed servers need not be assigned Machines nor do they need to be up and running in the reference system. Once the assembly is created you can remove these temporary managed servers from the reference system. You can control the actual number of Forms and Reports managed servers that have to be deployed using the "target" scalability property of the cluster_forms WLS appliance. Refer to Table B–44, " Scalability Properties of the Instance Appliance", for details on the scalability properties.

B.3.4.5 Oracle HTTP Server to Reports Cluster Configuration Requirement

In the reference system, if the Reports managed server(s) is part of a WebLogic cluster (cluster_reports) and it's front ended by an Oracle HTTP Server, make sure WebLogicCluster directive is used to for Oracle HTTP Server to Reports managed server(s) routing. By default, it's configured with WebLogicHost and WebLogicPort directives. Modify the following file to add an entry for WebLogicCluster directive and comment out WebLogicHost and WebLogicPort directives:

`$ORACLE_INSTANCE/config/OHS/<ohs_name>/moduleconf/reports_ohs.conf`

For example:

```
#mod_weblogic related entry
#<IfModule mod_weblogic.c>
<Location /reports>
SetHandler weblogic-handler

# Add this line:
WebLogicCluster machine1.domain:port1,machine2.domain:port2

# Comment following two entries
# WebLogicHost machine1.domain
# WebLogicPort port1
</Location>
#</IfModule>
```

B.3.4.6 tnsnames.ora

The `tnsnames.ora` file on the reference system is included in any assemblies that are created, and any databases (and host machines for them) which are referred to in the file are also referred to in the `tnsnames.ora` that is deployed as part of the deployed assemblies. Thus, generally the `tnsnames.ora` file should be empty (or be removed) from the reference system before the assemblies are created, particularly in cases

where the assemblies will be shipped to third parties (since those databases and machines will not exist in the new environment). In those cases, users should add their required database entries to the file on the Forms and Reports virtual nodes after deployment.

B.3.5 Resulting Artifact Type

For each Forms and Reports clusters and standalone (non-clustered) managed servers in the introspected WebLogic Server domain, the Forms and Reports extensions create a new appliance within the atomic Oracle WebLogic Server Assembly.

B.3.6 Wiring

The appliances are wired as follows:

B.3.6.1 Oracle HTTP Server Appliance to Forms and Reports WLS Appliances Wiring

If Oracle HTTP Server is configured on the reference system, one or more appliance outputs with the prefix "wls-" are created on the Oracle HTTP Server appliance. Connect the appliance output with the description property "loc=/forms" to the Forms appliance in the atomic WLS assembly and similarly connect the appliance output with the description property "loc=/reports" to the Reports appliance in the atomic WLS assembly.

B.3.6.2 Forms and Reports WLS Appliances to Oracle Internet Directory External Resource Wiring

On the reference system, if the Forms and Reports managed servers are configured with the Oracle Internet Directory server to support running Forms and Reports applications with single sign on protection, appliance outputs named "OIDConnection" are created on the Forms and Reports appliances.

Create an External Resource for this appliance output and enter the properties described in Section B.3.7, "Wiring Properties". If Forms and Reports are part of the same assembly, use the same External Resource to represent the Oracle Internet Directory Server.

Also refer to Oracle HTTP Server - OAM Server wiring as described in Section B.6.7.1, "Oracle Access Manager Admin Server".

B.3.6.3 Reports Appliance to Oracle Database Wiring

If a Reports WLS cluster was configured in the reference system, an appliance output named "job_repos_db" is created on the Reports cluster appliance (cluster_reports). This output should be connected to an Oracle Database appliance or an External Resource representing an Oracle Database that contains required Reports job database schemas (see Oracle Reports documentation for details about such schemas) and enter the properties defined in Section B.3.7, "Wiring Properties".

B.3.7 Wiring Properties

The following properties should be set for the External Resource (refer to Section B.3.6.2, "Forms and Reports WLS Appliances to Oracle Internet Directory External Resource Wiring") representing the Oracle Internet Directory server.

Table B–16 Oracle Internet Directory External Resource Properties

Name	Type	Req'd	Default	Description
UseOID	Boolean	true	true	This property determines whether or not to configure Forms-OID wiring in the virtual deployment. This property is set to *true* if the reference system was setup with Forms-OID/Reports-OID configuration. Setting this property to *false* disables Forms-OID/Reports-OID configuration in the virtual deployments. When this property is set to *true*, set the rest of the properties listed in this table.
Host	String	true	none	OID Server host name.
Port	Integer	true	none	OID Server port number.
OID_UserDn	Boolean	true	none	OID Server Administrator user name (typically `orcladmin`).
OID_Password	String	true	none	OID Server Administrator password.
sslConnection	Boolean	true	false	This property indicates whether to establish an SSL connection with the OID Server. When this property is set to *true*, you should provide the OID Server's SSL port number in the Port property.

Note: When Single-Sign-On (SSO) protection is to be enabled for the default Forms and Reports applications in the deployed environment, add the values **/reports/rwservlet/*,/forms/frmservlet?*oamMode=true*** to the `webgate.protectedResourcesList` user property on the OHS Appliance as described in Table B–25.

Note: If the `UseOID` property described in Table B–16 is set to false, that is, you chose not to enable Sign-Sign-On protection on the Forms and Reports applications in the deployed instance, then make sure that the values **/reports/rwservlet/*,/forms/frmservlet?*oamMode=true*** are removed from to the `webgate.protectedResourcesList` user property on the OHS Appliance, as described in Table B–25.

Applicable only for a Reports appliance, the following properties should be set for the Oracle Database appliance or an External Resource representing an Oracle Database that contains required Reports job database schemas.

Table B–17 Oracle Database Appliance Properties

Name	Type	Req'd	Default	Description
global-db-name	String	true	none	The Oracle System ID (SID) of the Oracle Database. Property defined at input endpoint.

Table B–17 (Cont.) Oracle Database Appliance Properties

Name	Type	Req'd	Default	Description
port	Integer	true	none	Oracle Database listener port number. Property defined at input endpoint.

Table B–18 Oracle Database External Resource Properties

Name	Type	Req'd	Default	Description
hostname	String	true	none	Oracle Database host name. Property defined at appliance level.
global-db-name	String	true	none	The Oracle System ID (SID) of the Oracle Database. Property defined at input endpoint.
port	String	true	none	Oracle Database listener port number. Property defined at input endpoint.

Table B–19 Reports Appliance Output Properties: job_repos_db

Name	Type	Req'd	Default	Description
username	String	true	none	The user needed for the database connection.
password	String	true	<empty>	The password for the user needed for the database connection.
address-name	String	true	none	The database address/service name that goes in the `tnsnames.ora` file (for example, `myProdDb`).

B.3.8 Oracle Forms and Reports Appliance Properties

Same as for Oracle WebLogic Server appliances.

(Reports only) If the reference system is enabled for a Reports cluster, specify the following property in your deployment plan:

```
wlsAssembly -->FileSets -> reportsClusterSharedDir
```

This is the shared NFS directory to which the VM should mount, and the location of the shared directory for the Oracle Reports shared cache.

B.4 Oracle Service Bus Support

Support for Oracle Service is provided via Oracle WebLogic Server plug-in. There is no separate plug-in for it. Oracle WebLogic Server plug-in can be used to introspect a Oracle WebLogic Server domain where Oracle Service Bus is configured.

B.4.1 Versions Supported

This plug-in supports version 11gR1 11.1.1.6.

B.4.2 Oracle Service Bus Introspection Parameters

There are no additional parameters required beyond those needed by Oracle WebLogic Server.

B.4.3 Reference System Prerequisites

There are no additional prerequisites beyond those defined by Oracle WebLogic Server.

B.4.4 Requirements

The following requirements apply to an Oracle WebLogic Server domain containing Oracle Service Bus.

B.4.4.1 Supported Domains

Supported OSB domains are where there is single server, one managed server with Oracle Service Bus and SOA or separate Oracle Service Bus and SOA clusters.

B.4.4.2 Requirement for Configuration Archiving

Turn on configuration archiving since domain security configurations will have to be modified to facilitate the introspection, it is advisable to turn on the configuration backup in WebLogic Server. Refer to the WebLogic Server documentation on configuration backup.

B.4.4.3 Export and Optionally Delete the OSB Artifacts from the Reference Domain

It is recommended that the OSB artifacts be exported before introspection. Optionally they can be deleted.

B.4.4.4 Delete Temporary Files from the Domain Directory

Before introspection and capturing file sets of the domain, all temporary files under the domain directory can be cleaned up. OVAB uses WebLogic pack utility to archive the domain directory from the reference domain. Files containing '\' in their names cannot be processed by the pack utility. They have to be manually removed before introspection.

B.4.4.5 Post Assembly Deployment Requirements

The following are requirements after deployment.

B.4.4.5.1 Import the OSB Artifacts

After the successful deployment of the OSB assembly instance, you can import the OSB artifacts that were exported and optionally deleted before the assembly creation began.

Create a new session for configuration changes and import the OSB artifacts that were saved earlier. After the successful import of the OSB artifacts, the session can be activated.

B.4.4.5.2 Apply Customizations

Some of the endpoint URLs may have to be customized in the deployed VMs as the network configuration is different. You can create a customization file for the assembly instance and after making necessary changes, execute the customizations.

Create a new session for configuration changes and apply the customizations using the OSB console. After execution of the customizations, the change session can be activated.

B.4.5 Resulting Artifact Type

Same as in Oracle WebLogic Server plug-in.

B.4.6 Wiring

Same as in Oracle WebLogic Server plug-in.

B.4.7 Wiring Properties

Same as in Oracle WebLogic Server plug-in.

B.4.8 Oracle Service Bus Appliance Properties

Same as in Oracle WebLogic Server plug-in.

B.4.9 Supported Template Types

The supported template type is Oracle Enterprise Linux (OEL).

B.5 Oracle SOA Plug-in

The Oracle SOA introspection plug-in introspects an existing SOA WebLogic Server deployment, capturing all the configuration in the SOA domain as well as the container configuration and deployed composites in MDS.

The SOA plug-in enables running configuration plans for deployed composites in the target environment.

B.5.1 Versions Supported

This plug-in supports version 11gR1 11.1.1.6.

B.5.2 Oracle SOA Service Engine Configuration

The SOA plug-in uses a property mappings files to map configuration elements to bean properties. For each property in a property mappings file, a user property is created on the admin appliance. These mappings files are contained under `/config` in the `soa-plugin.jar`. Each mappings file is associated with the corresponding service engine configuration file by a naming convention. Each mappings file adds "-mappings" to the configuration file name just before ".xml". For example, for the service engine configuration file "bpel-config.xml", the associated property mappings file is "bpel-config-mappings.xml".

Each property mappings file has the following structure.

Example B–1 Property Mappings File used by SOA Plug-in

```
<property-mappings>
  <prefix>soa.bpel</prefix>

<persistence-bean>oracle.soa.management.config.bpel.BPELConfig</persistence-bean>

  <property name="my-property">
    <bean property="myProperty" type="int"/>
  </property>
  <property name="my-nested-property">
    <bean property="nestedObject.myProperty" type="string"/>
```

```
  </property>
  ...
</property-mappings>
```

Table B–20 describes elements in the property mappings files:

Table B–20 SOA WebLogic Server: Property Mappings File Elements

Parameter	Description
prefix	Prefix to add to each property in the file.
persistence-bean	Bean used to set/get property values. The bean can use any mechanism for reading/writing to the service engine configuration file. All persistence beans currently use JAXB for persistence.
property	Contains information about the property.
property.name	The name of the property which will be visible to the end user. This name will be prefixed by the above mentioned prefix.
bean	Contains information about the bean property that this user property is mapped to.
bean.property	The name of the property in the bean. This name is used to determine setter/getter names. For a property with location="myProperty", the getter name is `getMyProperty` and the setter is `setMyProperty`. For boolean types, "isMyProperty" and "hasMyProperty" are also tried. If the property in the bean is on a nested object, that can be specified using the notation location="nestedObject.myProperty". In this case, "getNestedObject()" would be called on the persistence bean and on the result "getMyProperty()" would be called.
bean.type	The type of the property in the bean. Must be one of {string,int,long,float,double,boolean}

B.5.3 Oracle SOA Application Configuration

Application-scoped configuration is configuration scoped to a composite application. The SOA plug-in exposes application configuration as user properties which you can update for the target environment when creating an assembly. Application configuration supports the configuration exposed through SOA configuration plans.

The following types of application configuration are supported:

- Any composite, service component, reference and binding properties in the SOA composite application file (composite.xml)
- Attribute values for bindings (for example, the location for binding.ws)
- `schemaLocation` attribute of an import in a WSDL file
- `location` attribute of an include in a WSDL file
- `schemaLocation` attribute of an include, import, and redefine in an XSD file
- Any properties in JCA adapter files
- Modify and add policy references for the following:
  - Service component
  - Service and reference binding components

See *Oracle SOA Configuration Plan* for more information on supported configuration.

During introspection the SOA plug-in checks each composite for a configuration plan. If no configuration plan exists, a default configuration plan is generated for the composite.

The plug-in parses the configuration plan and exposes each element in the configuration plan as a user property in Oracle Virtual Assembly Builder. The configuration plan is an XML document containing hierarchical data.

B.5.3.1 Configuration Plan User Properties

User Property keys are encoded using a "dot" notation where levels of the XML hierarchy are separated by a period ('.') and are referred to as path tokens or tokens. Elements that have an attribute, such as *name*, that uniquely identify the element are included in the token and are separated from the element type by an underscore ('_').

The portion of the token following the '_' is referred to as the "token id". Each token type (such as composite, or service) may have rules for what constitutes the token id. All configuration plan properties start with `soa.cp` to make them easily identifiable, as shown in Example B–2.

Example B–2 Configuration Plan used by SOA Plug-in

```
<composite name="HelloWorldProject">
  <service name="helloworldbpel_client_ep">
    <binding type="ws">
      <attribute name="port">
        <replace>http://xmlns.oracle.com/myPort</replace>
      </attribute>
    </binding>
  </service>
</composite>
```

The associated user property is:

```
soa.cp.composite_ HelloWorldProject_1-1.service_ helloworldbpel_client_ep.binding_
ws.attribute_port
```

In this XML code example there is one configuration element, the attribute "port". The composite, service, binding and attribute tokens each have a token id. For the composite token id, it is the name attribute and composite version. For the binding element it is the binding type.

The following are a list of configuration plan elements and their associated path tokens, including the token id rules:

Table B–21 describes elements in the configuration plan:

Table B–21 SOA WebLogic Server: Configuration Plan Elements

Element	Path Token
Composite	composite_<composite name>_<composite version with dash ('-') replacing period ('.')>
Service	service_<service name>
Reference	reference_<reference name>
Import	import_<generated id>
Property	property_<property name>
Component	component_<component name>

Table B–21 (Cont.) SOA WebLogic Server: Configuration Plan Elements

Element	Path Token
Attribute	attribute_<attribute name>
Binding	binding_<binding type>
SearchReplace	There is no token for SearchReplace. See `Search` and `Replace` tokens.
Search	search_<generated id>
Replace	replace_<generated id>
PolicyReference	PolicyReference_<generated id>
Callback	callback (The configuration plan currently only allows one per element. There is no token id.)
OverrideProperty	OverrideProperty_<override property name>
WsdlAndSchema	wsdlAndSchema_<wsdlAndSchema name>

B.5.3.2 External References

Composite configurations may contain references to external resources. Examples of this are Web service reference bindings where a location is specified, and imports. You can update these external references for the target environment.

Only "search" properties are parsed for external references. For search properties that contain an external reference, the `replace` value will be the search value with the protocol://host:port portion of the URL re-placed with a token in the form of <#Output::output-name#>. This correlates that user property with the appropriate assembly output.

When composites are deployed using a configuration plan, the configuration artifacts in the composite are modified based on the configuration plan. Because of this, if a configuration plan is used to deploy a composite to a staging environment and then later the staging environment is introspected, any external references in the configuration plan will have already been replaced in the composite. As a result, none of the external references in the configuration plan will be replaced during reconfiguration because the search value no longer exists in the composite configuration. To handle this case, each output has a user property called `soa.reference.aliases`. This alias property allows for aliases to be specified for the output. By default, the location portion of the replace value is set as an alias. Additional aliases may be added manually using a ',' or ' ' separator between aliases. In addition to the original search value, each of the aliases will also be replaced during reconfiguration.

Example B–3 External Reference in a Configuration Plan used by SOA Plug-in

```
<composite name="compositeA">
  <reference name="reference1">
    <binding type="ws">
      <attribute name="location">
        <searchReplace>
          <search>http://my.host.com:8080/some/path/info</search>
          <replace>https://my.new-host.com:8081/some/path/info</replace>
        </searchReplace>
    </attribute>
    </binding>
  </reference>
</composite
```

The associated user properties are:

- Search Property:
  - **name**=soa.cp.composite_compositeA_1-0.reference_reference1.binding_ws.attribute_location.search
  - **value**=http://my.host.com:8080/some/path/info
- Replace Property:
  - **name**=soa.cp.composite_compositeA_1-0.reference_reference1.binding_ws.attribute_location.replace
  - **value**=<#Output::my_host_com:8080#>/some/path/info

In the example, the original `replace` value is not used. Instead, the original search value is used with the protocol, host, and port being replaced with a token that correlates this search/replace with an external reference output. At reconfiguraiton, this token is replaced with the protocol, host, and port obtained from the external resource that is connected to the external reference output of the name "my_host_com:8080". The replacement token should not be modified, but the path information following the replacement token can be updated if needed.

If another external reference that matches "http://my.host.com:8080" is contained in a configuration plan of any composite deployed to the domain, the same output is used.

In the following configuration plan, no new output is added to the assembly:

Example B–4 Configuration Plan Where No New Output is Added to the Assembly

```
<composite name="compositeB">
  <import>
    <searchReplace>
      <search>http://my.host.com:8080</search>
      <replace/>
    </searchReplace>
  </import>
</composite>
```

The associated user properties are:

- Search Property:
  - **name**=soa.cp.composite_compositeB_1-0.import.search
  - **value**=http://my.host.com:8080
- Replace Property:
  - **name**=soa.cp.composite_compositeB_1-0.import.replace
  - **value**=<#Output::my_host_com:8080#>

B.5.4 Reference System Prerequisites

The SOA plug-in has runtime dependencies on product jars in the SOA/WebLogic Server distribution. These dependencies are not part of the plug-in's initial classpath. You must load these product dependencies dynamically from the product distribution.

These dependencies fall into two functional categories:

- Fabric dependencies include the capturing of file sets of composites and the deployment of composites, and are contained in `dynlib/soa-plugin-fabric.jar`.

- MDS dependencies include all interactions with MDS, and are contained in `dynlib/soa-plugin-mds.jar`.

B.5.5 Resulting Artifact Type

The introspection produces a SOA assembly consisting at minimum of an admin server and a managed server, but most likely consisting of an admin server, and a cluster of managed SOA servers. Each of these servers, or cluster, is represented by an appliance.

B.5.6 Wiring

Composite configurations may contain references to external resources. Examples of this are Web service reference bindings where a location is specified, and imports. These external references are exposed so that they can be updated for the target environment.

For each distinct external reference an output is added to the associated assembly. You must wire each output to an input of an external resource using Oracle Virtual Assembly Builder Studio. A reference is distinct from another reference if it differs in protocol, host/IP address or port.

B.5.7 Wiring Properties

See Section B.5.3.2, "External References".

B.5.8 Oracle SOA Appliance Properties

Oracle SOA appliances have the following properties:

- User properties (see Section B.5.2, "Oracle SOA Service Engine Configuration" and Section B.5.3, "Oracle SOA Application Configuration") and
- System properties (Table B–22).

Table B–22 Oracle SOA: System Properties

Name	Type	Req'd	Default	Description
SOA_MW_HOME	String	false	none	The Middleware Home directory.
SOA_HOME	String	false	none	The SOA Oracle Home (for example, "AS11gR1SOA").
SOA_JDK_HOME	String	false	none	The JDK Home.
IS_SOA_SERVER	String	false	none	Identifies SOA Managed Servers.

B.5.9 Extensions of the Plug-in

None.

B.5.10 Supported Template Types

The supported template type is Oracle Enterprise Linux (OEL).

B.6 Oracle HTTP Server Introspector Plug-in

The Oracle HTTP Server introspection plug-in examines a single Oracle HTTP Server appliance from an Oracle Web Tier instance. Oracle HTTP Server and the Oracle

Process Manager and Notification Server that manages it are captured, as well as the Oracle Access Manager 11g admin server and any server instances.

B.6.1 Versions Supported

This plug-in supports versions 11gR1 11.1.1.5 and 11gR1 11.1.1.6.

B.6.2 Oracle HTTP Server Introspection Parameters

Table B–23 lists the introspection parameters for Oracle HTTP Server:

Table B–23 Oracle HTTP Server Plug-in Introspection Parameters

Parameter	Description
oracleInstance	The fully qualified path to the Oracle Instance that contains the Oracle HTTP Server appliance to be introspected.
componentName	The name of the Oracle HTTP Server appliance within the Oracle Instance specified. For example, 'ohs1'.

B.6.3 Reference System Prerequisites

The Oracle HTTP Server plug-in supports Oracle Access Manager 11g WebGate for single-sign on if you meet the following requirements:

- You install WebGate 11g in the same `MW_HOME` as Oracle HTTP Server, with the `ORACLE_INSTANCE` of WebGate 11g the same as the `ORACLE_INSTANCE` of Oracle HTTP Server.
- You configure Oracle HTTP Server with Oracle Access Manager 11g Webgate and register the Webgate 11g agent with the Oracle Access Manager server.
- The RREG toolkit, `rreg-toolkit.jar`, is available on the reference system under `OHS_MW_HOME/Oracle_FRHome1/oam/server/rreg/client`.

  The RREG toolkit is located in the `oam` directory inside the Oracle home after Oracle Forms and Reports is installed, and is used to register the partner application with Oracle Access Manager 11g during reconfiguration. The version of the RREG toolkit must match the version of Oracle Access Manager 11g. If the RREG toolkit of the correct version is not available under this location, then the introspection fails.

  If the Web tier is set up on a separate node for the Forms and Reports installation on the reference system, ensure that you meet the requirements described in Section B.3.3.1, "Adding Partner Application Registation Utility (Web Tier on a Separate Node)" prior to introspection.

  For more information on the RREG toolkit, see "Configuring Oracle Forms and Reports with Oracle Access Manager in Secure Mode" in *Oracle® Fusion Middleware Installation Guide for Oracle Forms and Reports*.

B.6.4 Resulting Artifact Type

A single scalable appliance.

B.6.5 Requirements

Make sure Oracle Access Manager admin server and the instances are up and running.

B.6.6 Wiring

Inputs are created on the Oracle HTTP Server appliance for each Listen or Port directive found in the configuration. The protocol of an Oracle HTTP Server input is set to *http* unless the Listen directive is found inside a VirtualHost directive and has *SSLEngine on* directive set, then it has the protocol set to *https*. Typically Web Cache outputs are connected to Oracle HTTP Server inputs.

Outputs on the Oracle HTTP Server appliance are created based on various directives related to Oracle WebLogic Server in the Oracle HTTP Server configuration. The outputs indicate which inputs on an Oracle WebLogic Server assembly to connect to through the output 'description'.

One output endpoint is created on the Oracle HTTP Server appliance to represent the Oracle Access Manager admin server connection.

B.6.7 Wiring Properties

All input endpoints have two editable properties - `port` and `description`, and one non editable property - a list of `protocols`. The `protocols` indicate what sort of outputs can be connected to the input.

All output endpoints have one editable property - `description`, and two non-editable properties - `protocol` and `singleton`. The `protocol` indicates what sort of input can be connected to the output. `Singleton` indicates what sort of appliance the output can be connected to. If `singleton` is true, the output can only be connected to an input on an appliance that has a scalability absolute max value of 1.

B.6.7.1 Oracle Access Manager Admin Server

One output endpoint is created on the Oracle HTTP Server appliance to represent the Oracle Access Manager admin server connection. The output exposes the following non-editable properties:

Table B–24 Output for Oracle Access Manager Admin Server Connection

Name	Type	Default	Description
`name`	String	OAMRegistration	Name of the Oracle Access Manager admin server.
`protocols`	String	Discovered at introspection.	The security protocol. Can be *simple*, *open*, or *cert*.

An External Resource should be created to represent the Oracle Access Manager admin server. The external resource exposes the `admin-username`, `admin-password`, `port` and `protocols` as input endpoint properties.

B.6.8 Oracle HTTP Server Appliance Properties

Oracle HTTP Server appliances have user properties (Table B–25) and system properties (Table B–26). Properties with the `webgate` prefix are required for partner application registration during reconfiguration.

Note: There are a number of other properties that you can use in partner application registration during reconfiguration. For example, `maxConnections`, `maxSessionTime`, `failOverThreshold`, etc. None of these properties are exposed by the Oracle HTTP Server plug-in as appliance properties.

You can change these properties after reconfiguration. For a list of supported properties, see Registering Partners (Agents and Applications) Remotely in *Oracle® Fusion Middleware Administrator's Guide for Oracle Access Manager with Oracle Security Token Service* .

Table B–25 Oracle HTTP Server: User Properties

Name	Type	Req'd	Default	Description
`userDirective`	String	false	none	Indicates whether the user directive exists in the configuration files.
`groupDirective`	String	false	none	Indicates whether the group directive exists in the configuration files.
`readymetric.timeout`	Integer	false	300	Sets the timeout duration, in seconds.
`readymetric.pollingPeriod`	Integer	false	5	Sets the polling period, in seconds.
`webgate.security`	String	true	Discovered at introspection.	Indicates the level of communication transport security between the Agent and the Oracle Access Manager Server (this must match the level specified for the Oracle Access Manager Server). Possible values: *simple*, *open* or *cert*. If the value of webgate.security user property specified is *cert*, you must install the required certificates on the deployed VM and establish the trust channel between WebGate and Oracle Access Manager server manually after reconfiguration. You do not need to install the certificates if the same (reference system) Oracle Access Manager server is used for partner application registration during reconfiguration, because the certificates required for the communication between WebGate and Oracle Access Manager server are, by default, copied from the reference system to the deployed VM.
`webgate.agentPassword`	String	false	N/A	An optional, unique password for this Webgate, which can be assigned during this registration process.
`webgate.agentKeyPassword`	String	false	N/A	Required by the Oracle Access Manager Server to generate `password.xml`. Required only when the webgate.security value is "cert".

Table B–25 (Cont.) Oracle HTTP Server: User Properties

Name	Type	Req'd	Default	Description
webgate.protectedResourcesList	String	true	/,/.../*	Specifies the resource URLs that you want the Oracle Access Manager Agent to protect with some authentication scheme. The resource URLs must be relative paths to the agentBaseUrl. Specify as a comma-separated list. For example, in an environment where Oracle HTTP Server is a front-end to Oracle Forms and Reports: /reports/rwservlet/*,/forms/frmservlet?*oamMode=true
webgate.publicResourcesList	String	false	/public/index.html	Specifies the resource URLs that you want to keep public (not protected by the Oracle Access Manager Agent). The resource URLs must be relative paths to the agentBaseUrl. For instance, you may want to specify the Home page or the Welcome page of your application. Specify as a comma-separated list.
webgate.excludedResourcesList	String	false	/excluded/index.html	Specifies the HTTP type resource URLs that you want to keep public (not protected by the Oracle Access Manager Agent). The resource URLs must be relative paths to the agentBaseUrl. For instance, you may want to specify the Home page or the Welcome page of your application. Only HTTP resource types can be excluded. Typically security insensitive files like Images (*.jpg, *.png) that do not require Authentication, Authorization, Response processing, Session management, and Auditing. Excluded resources cannot be added to any user-defined policy in the console. Specify as a comma-separated list.

Note: The agentBaseURL is generated automatically during reconfiguration based on the deployed VM hostname and Oracle HTTP Server instance port.

Table B–26 Oracle HTTP Server: System Properties

Name	Type	Req'd	Default	Description
ORACLE_INSTANCE	String	false	none	The path the user specified as the Oracle instance.
COMPONENT_TYPE	String	false	none	The type of the appliance being introspected.
COMPONENT_NAME	String	false	none	The name of the appliance being introspected.
ORACLE_HOME	String	false	none	The path to the Oracle home related to this Oracle instance.

Table B–26 (Cont.) Oracle HTTP Server: System Properties

Name	Type	Req'd	Default	Description
FMW_HOME	String	false	none	The path to the Fusion Middleware home related to this Oracle instance.
JAVA_HOME	String	false	none	The path to the Java home used by this Oracle instance.
oraInstLocDir	String	false	none	The directory used by Oracle Universal Installer for installation files.

B.6.9 Extensions of the Plug-in

None.

B.6.10 Supported Template Types

The supported template type is Oracle Enterprise Linux (OEL).

B.7 Oracle Web Cache Plug-in

The Oracle Web Cache introspection plug-in examines a single Oracle Web Cache appliance from an Oracle Web tier instance. Both Oracle Web Cache and the Oracle Process Manager and Notification Server that manages it are captured.

B.7.1 Versions Supported

This plug-in supports versions 11gR1 11.1.1.5 and 11gR1 11.1.1.6.

B.7.2 Oracle Web Cache Introspection Parameters

Table B–27 lists the introspection parameters for Oracle Web Cache Server:

Table B–27 Oracle Web Cache Plug-in Introspection Parameters

Parameter	Description
oracleInstance	The fully qualified path to the Oracle Instance that contains the Oracle HTTP Server appliance to be introspected.
componentName	The name of the Oracle HTTP Server component within the Oracle Instance specified. For example, 'ohs1'.

B.7.3 Reference System Prerequisites

The Oracle Web Cache introspection plug-in does not support configurations with multiple network interface cards (NICs). If the Web Cache configuration binds to more than one NIC, introspection will fail. To avoid this failure, before introspection set all IP addresses in the `<LISTEN>` elements to "ANY".

B.7.4 Requirements

The following requirements apply to Oracle Web Cache:

B.7.4.1 Requirement to Update Virtual Host Map Properties

Whenever you make a port change, you must update your virtual host map (VHM) ports by manually updating the properties associated with the VHMs.

B.7.5 Resulting Artifact Type

A single scalable appliance.

B.7.6 Wiring

Inputs will be created on the Web Cache appliance for each `<LISTEN>` element found in webcache.xml.

Outputs on the Oracle Web Cache appliance indicate how they should be connected to an Oracle HTTP Server appliance via the output 'description'. The outputs are created based on various directives in the Oracle Web Cache configuration and the description can be used to determine which input on the Oracle HTTP Server appliance to connect the Oracle Web Cache output to.

B.7.7 Wiring Properties

All input endpoints have two editable properties - `port` and `description`, and one non editable property - a list of `protocols`. The `protocols` indicate what sort of outputs can be connected to the input.

All output endpoints have one editable property - `description`, and two non-editable properties - `protocol` and `singleton`. The protocol indicates what sort of input can be connected to the output. `Singleton` indicates what sort of appliance the output can be connected to. If `singleton` is true, the output can only be connected to an input on an appliance that has a scalability absolute max value of 1.

The following properties are specific to Oracle Web Cache endpoints:

A single output is created for each virtual host mapping. Each output contains the properties described in Table B–28:

Table B–28 Oracle Web Cache: Output Properties

Name	Type	Req'd	Default	Description
relatedOriginServers	String	false	OEL	A comma separated list of all of the host definition names for the given virtual host map. For example, 'host1,host2,host3'
vhm-siteX-HOST	String	false	none	The value of the host property of the virtual host map.
PORT	String	false	none	The value of the port property of the virtual host map.

B.7.8 Oracle Web Cache Appliance Properties

Table B–29 describes Oracle Web Cache appliance user properties.

Table B–29 Oracle Web Cache: User Properties

Name	Type	Req'd	Default	Description
adminPassword	String	false	none	The password to use for the MONITORING password. If not specified, the system property originalAdminPassword will be used.

Table B–29 (Cont.) Oracle Web Cache: User Properties

Name	Type	Req'd	Default	Description
statisticsPassword	String	false	none	The password to use for the INVALIDATION password. If not specified, the system property originalStatisticsPassword will be used.
`readymetric.timeout`	Integer	false	300	Sets the timeout duration, in seconds.
`readymetric.pollingPeriod`	Integer	false	5	Sets the polling period, in seconds.
siteX-HOST	String	false	<read from file>	The host name for the site definition.
siteX-PORT	String	false	<read from file>	The port value for the site definition.

Table B–30 describes Oracle Web Cache appliance system properties.

Table B–30 Oracle Web Cache: System Properties

Name	Type	Req'd	Default	Description
oracleInstance	String	false	none	The path the user specified as the Oracle instance.
componentType	String	false	none	The type of the appliance being introspected.
componentName	String	false	none	The name of the appliance being introspected.
oracleHome	String	false	none	The path to the Oracle home related to this Oracle instance.
javaHome	String	false	none	The path to the Java home used by this Oracle instance.
originalAdminPassword	String	false	See the description column.	This is the password hash as it exists for this Oracle Web Cache instance. The deployed system uses this value unless you specifically set the value of the 'adminPassword' user property. The default value is the hashed password from the existing Oracle Web Cache configuration for the 'MONITORING' password hash.
originalStatisticsPassword	String	false	A hashed value.	This is the value of the password hash from the existing Oracle Web Cache configuration for the 'INVALIDATION' password hash.
oraInstLocDir	String	false	none	The directory used by Oracle Universal Installer for installation files.

B.7.9 Extensions of the Plug-in

None.

B.7.10 Supported Template Types

The supported template type is Oracle Enterprise Linux (OEL).

B.8 Oracle Database (SIDB) Plug-in

The single-instance Oracle Database introspection plug-in examines a single-instance Oracle Database appliance and captures its metadata.

B.8.1 Versions Supported

This plug-in supports versions 10gR2, 11gR1, and 11gR2.

B.8.2 Oracle Database Introspection Parameters

Table B–31 lists the introspection parameters for Oracle Database:

Table B–31 Oracle Database Plug-in Introspection Parameters

Parameter	Description
asmHome	This parameter is required if ASM is used as the storage type and it is installed in a separate Oracle Home.
dbHome	The ORACLE_HOME of the Oracle RDBMS to be introspected.
oracleSid	The Oracle System ID (SID) of the Oracle RDBMS to be introspected.
shutdownDBOK	This flag needs to be passed to approve the database reboot.
sysDBAUserName	Database account with SYSDBA privileges. This parameter is required only if OS authentication is disabled for the current database.

B.8.3 Oracle Database Introspection Password Parameters

Table B–32 lists the introspection password parameters for Oracle Database. When performing introspection using the `abctl` tool, a prompt is shown to enter values for these parameters. Oracle Virtual Assembly Builder Studio provides password fields for these parameters.

Table B–32 Oracle Database Plug-in Introspection Parameters (Prompted During Introspection)

Parameter	Description
rootPassword	Optional. The rootPassword parameter is required to change the permissions of the ORACLE_HOME files to make capturing of file sets possible.
sysDBAPassword	Optional. Password for sysDBAUserName user. This parameter is required only if OS authentication is disabled for the current database.

B.8.4 Reference System Prerequisites

This introspection plug-in does not support configurations with multiple NICs.

B.8.5 Requirements

The following requirements apply to Oracle Database:

The base system image OS version must match the version of the reference system.

B.8.6 Resulting Artifact Type

A single appliance.

B.8.7 Wiring

An input is created on the SIDB appliance with a default Listener and Port. The protocol of an SIDB input is set to 'jdbc'.

B.8.8 Wiring Properties

The input endpoint has two editable properties - `port` and `description`, and two non-editable properties - `protocol` and `ORACLE_SID`. The `protocol` indicates what sort of output can be connected to the input.

B.8.9 Oracle Database Appliance Properties

Assemblies with an Oracle Database appliance have user properties (Table B–33) and system properties (Table B–34).

Table B–33 Oracle SIDB Plug-in: User Properties

Name	Type	Req'd	Default	Description
asm-password	String	false	none	Password for SYS asm account.
db-account-password	Password	true	none	The password for database accounts SYS, SYSTEM, SYSMAN, and DBSNMP.

Table B–34 Oracle SIDB Plug-in System Properties

Name	Type	Req'd	Default	Description
ASM_BASE	String	false	none	ASM ORACLE_BASE path.
ASM_HOME	String	false	none	ASM ORACLE_HOME path.
ASM_OWNER	String	false	none	The OS user who owns the ASM ORACLE_HOME.
DATA_ASM_ DISCOVERY_ STRING	String	false	none	Path to discover all data asm disks.
DATA_ASM_DISK_ GROUP_NAME	String	false	none	Name of the data asm diskgroup.
DATA_ASM_DISK_ GROUP_ REDUNDANCY	String	false	none	Data asm diskgroup level of redundancy.
DATA_ASM_DISKS	String	false	none	Paths of the disks that belong to the data asm diskgroup.
DATA_STORAGE_ TYPE	String	false	none	Storage type for database data.
DB_BASE	String	false	none	The Oracle database base path.
DB_GROUP	String	false	none	The group of the OS user who owns Oracle home.
DB_HOME	String	false	none	The Oracle database home path.
DB_HOME_NAME	String	false	none	The name of the Oracle Home.
DB_LISTENER_ NAME	String	false	none	Oracle database listener name.

Table B–34 (Cont.) Oracle SIDB Plug-in System Properties

Name	Type	Req'd	Default	Description
DB_ORACLE_ GROUPS	String	false	none	The OSDBA, OSOPER and OSASM groups.
DB_OWNER	String	false	none	The OS user who owns Oracle home.
DB_USING_ASM	String	false	none	Set to *true* if either of database or recovery files are stored on ASM as per reference system.
DB_VERSION	String	false	none	Version of Oracle database software on reference system.
ORACLE_SID	String	false	none	The Oracle database SID.
ORIGINAL_ GLOBAL_DB_NAME	String	false	none	The database unique name on reference system.
RECOVERY_ASM_ DISCOVERY_ STRING	String	false	none	Path to discover all recovery asm disks.
RECOVERY_ASM_ DISK_GROUP_ NAME	String	false	none	Name of the recovery asm diskgroup.
RECOVERY_ASM_ DISK_GROUP_ REDUNDANCY	String	false	none	Recovery asm diskgroup level of redundancy.
RECOVERY_ASM_ DISKS	String	false	none	Paths of the disks that belong to the recovery asm diskgroup.
RECOVERY_ STORAGE_TYPE	String	false	none	Storage type for database recovery.

B.8.10 Extensions of the Plug-in

None.

B.8.11 Supported Template Types

The supported template type is Oracle Enterprise Linux (OEL).

B.9 Oracle RAC Database (RACDB) Plug-in

The RACDB introspection plug-in examines Oracle Clusterware and RAC Database components and captures their metadata.

B.9.1 Versions Supported

This plug-in supports version 11gR2.

B.9.2 Oracle RAC Database Introspection Parameters

Table B–35 lists the introspection parameters for the RACDB introspection plug-in:

Table B–35 Oracle RACDB Plug-in Introspection Parameters

Parameter	Description
asmHome	This parameter is required if ASM is used as the storage type and it is installed in a separate Oracle Home.
crsHome	The ORACLE_HOME of the Oracle CRS to be introspected.
dbHome	The ORACLE_HOME of the Oracle RDBMS to be introspected.
globalDbName	The global database name of the Oracle RDBMS to be introspected.
shutdownDBOK	This flag needs to be passed to approve the database reboot.
sysDBAUserName	Optional. Database account with SYSDBA privileges. This parameter is required only if OS authentication is disabled for the current database.

B.9.3 Oracle RAC Database Introspection Password Parameters

Table B–36 lists the introspection password parameters for the Oracle RAC Database plug-in. When performing introspection using the `abctl` tool, a prompt is shown to enter values for these parameters. Oracle Virtual Assembly Builder Studio provides password fields for these parameters.

Table B–36 Oracle RACDB Plug-in Introspection Parameters (Prompted During Introspection)

Parameter	Description
rootPassword	Optional. The rootPassword parameter is required to change the permissions of the ORACLE_HOME files to make capturing of file sets possible.
sysDBAPassword	Optional. Password for sysDBAUserName user. This parameter is required only if OS authentication is disabled for the current database.

B.9.4 Reference System Prerequisites

The Clusterware stack must be running during introspection.

B.9.5 Requirements

The following requirements apply to Oracle RAC Database:

The base system image OS version must match the version of the reference system.

B.9.6 Resulting Artifact Type

A single appliance.

B.9.7 Wiring

For an 11.2 series RACDB, a single input is created on the RACDB appliance.

For a pre-11.2 series RACDB, inputs are created on the RACDB appliance for each Listener or Port directive found in the configuration.

B.9.8 Wiring Properties

For 11.2 series RACDB, the input endpoint has two editable properties, `scan-name` and `global-db-name`, and two non-editable properties, `protocol` and `port`, which indicate what sort of output can be connected to the input.

For pre-11.2 series RACDB, the input endpoint has a editable property `global-db-name` and two non-editable properties, `protocol` and `port`.

B.9.9 Oracle Database Appliance Properties

Assemblies with an Oracle Database appliance have user properties (Table B–37) and system properties (Table B–38).

Table B–37 Oracle RAC Database: User Properties

Name	Type	Req'd	Default	Description
asm-password	String	false	none	Password for SYS asm account.
cluster-name	String	false	new_cluster	Name for cluster (only for pre-11.2 series Oracle Database).
db-account-password	Password	true	none	The password for database accounts SYS, SYSTEM, SYSMAN, and DBSNMP.

Table B–38 Oracle RAC Database System Properties

Name	Type	Req'd	Default	Description
CRS_BASE	String	false	none	The Clusterware base path.
CRS_HOME	String	false	none	The Clusterware home path.
CRS_OWNER	String	false	grid	The name of the OS user who will be the owner of the Clusterware home.
CRS_GROUP	String	false	oinstall	The name of the OS user group of the owner of the Clusterware home.
CRS_ORACLE_ GROUPS	String	false	oinstall	The OSDBA, OSOPER and OSASM groups.
VOTING_DISKS_ LOCATIONS	String	false	none	Locations of voting disks (only for File System storage type for clusterware files).
VOTING_DISKS_ REDUNDANCY	String	false	none	Voting disks redundancy. (only for File System storage type for clusterware files).
OCR_DISKS_ LOCATIONS	String	false	none	Locations of ocr disks(only for File System storage type for clusterware files).
OCR_DISKS_ REDUNDANCY	String	false	none	OCR disks redundancy. (only for File System storage type for clusterware files)
SCAN_PORT	String	false	1521	Port for SCAN listener.
CRS_STORAGE_ TYPE	String	false	none	Storage type for clusterware files as per reference system.
CRS_VERSION	String	false	none	Version of Clusterware software on reference system.
CRS_ASM_DISK_ GROUP_NAME	String	false	OVMOCRVD	Name of the clusterware asm diskgroup.

Table B–38 (Cont.) Oracle RAC Database System Properties

Name	Type	Req'd	Default	Description
CRS_ASM_ DISCOVERY_ STRING	String	false	/dev/raw/ovmocrvd*	Path to discover all data asm disks.
CRS_ASM_DISK_ GROUP_ REDUNDANCY	String	false	NORMAL	Clusterware asm diskgroup level of redundancy.
CRS_ASM_DISKS	String	false	/dev/raw/ovmocrvd0,/dev/raw/ovmocrvd1,/dev/raw/ovmocrvd2	Paths of the disks that belong to clusterware asm diskgroup.
ASM_BASE	String	false	none	ASM ORACLE_BASE path.
ASM_HOME	String	false	none	ASM ORACLE_HOME path.
ASM_OWNER	String	false	grid	The OS user who owns ASM ORACLE_HOME.
DATA_ASM_ DISCOVERY_ STRING	String	false	/dev/raw/asm*	Path to discover all data asm disks.
DATA_ASM_DISK_ GROUP_NAME	String	false	none	Name of the data asm diskgroup.
DATA_ASM_DISK_ GROUP_ REDUNDANCY	String	false	none	Data asm diskgroup level of redundancy.
DATA_ASM_DISKS	String	false	none	Paths of the disks that belong to data asm diskgroup.
RECOVERY_ASM_ DISCOVERY_ STRING	String	false	/dev/raw/asm*	Path to discover all recovery asm disks.
RECOVERY_ASM_ DISK_GROUP_ NAME	String	false	none	Name of the recovery asm diskgroup.
RECOVERY_ASM_ DISK_GROUP_ REDUNDANCY	String	false	none	Recovery asm diskgroup level of redundancy.
RECOVERY_ASM_ DISKS	String	false	none	Paths of the disks that belong to the recovery asm diskgroup.
RECOVERY_ STORAGE_TYPE	String	false	none	Storage type for database recovery.
DATA_STORAGE_ TYPE	String	false	none	Storage type for database as per reference system.
RECOVERY_ STORAGE_TYPE	String	false	none	Storage type for database recovery files as per reference system.
DB_USING_ASM	String	false	none	Set to *true* if either of database or recovery files are stored on ASM as per reference system.
DB_VERSION	String	false	none	Version of Oracle database software on reference system.

B.9.10 Extensions of the Plug-in

None.

B.9.11 Supported Template Types

The supported template type is Oracle Enterprise Linux (OEL).

B.10 Oracle Traffic Director Plug-In

The Oracle Traffic Director plug-in introspects an Oracle Traffic Director *configuration* on a reference system. In the context of Oracle Traffic Director, a configuration is a collection of configurable elements (metadata) that determine the run-time behavior of an Oracle Traffic Director instance.

For an overview of Oracle Traffic Director, see "Getting Started with Oracle Traffic Director" in the *Oracle Traffic Director Administrator's Guide*.

B.10.1 Versions Supported

The Oracle Traffic Director plug-in supports Oracle Traffic Director 11.1.1.6.

B.10.2 Oracle Traffic Director Introspection Parameters

Table B–39 lists the introspection parameters for Oracle Traffic Director. All the parameters are mandatory.

Table B–39 Oracle Traffic Director Plug-In Introspection Parameters

Parameter	Description
oracleHome	The fully qualified path to the directory in the reference system in which the Oracle Traffic Director binaries are installed.
oracleInstance	The fully qualified path to the directory in the reference system in which the Oracle Traffic Director administration server instance exists.
configName	The name of the Oracle Traffic Director configuration that the plug-in should introspect.

B.10.3 Reference System Prerequisites

For the Oracle Traffic Director plug-in to successfully introspect the Oracle home and the administration server instance, the following prerequisites must be fulfilled on the reference system:

- Oracle Traffic Director must be installed.
- The Oracle Traffic Director administration server instance must be configured.
- At least one configuration must be available.

Note: The administration server need not be running.

B.10.4 Resulting Artifact Type

The result of the introspection is an atomic assembly that contains the following:

- An Oracle Traffic Director administration server appliance
- An Oracle Traffic Director instance appliance

By default, when the instance appliance is deployed, two virtual machines will be created, regardless of the number of nodes to which the Oracle Traffic Director configuration was deployed on the reference system. The number of virtual machines to be created during the deployment process can be controlled through the scalability properties (see Table B–42, " Scalability Properties of the Administration Server Appliance").

B.10.5 Wiring

The appliances in the Oracle Traffic Director assembly can be wired to external components, and to other appliances and assemblies, through input and output endpoints.

Note: In this release, you can wire the Oracle Traffic Director assembly to only a WLS assembly.

B.10.5.1 Wiring Endpoints of the Administration Server Appliance

An input endpoint is created for the HTTPS listen port of the Oracle Traffic Director administration server appliance. The input endpoint of the administration server appliance has two editable properties, `port` and `description`.

No output endpoint is created for the administration server appliance.

B.10.5.2 Wiring Endpoints of the Instance Appliance

An input endpoint is created for each HTTP listen port in the Oracle Traffic Director configuration. Each input endpoint of the instance appliance has two editable properties, `port` and `description`.

An output endpoint is created for each origin-server pool in the configuration. Each output endpoint of the instance appliance has one editable property, `description`.

Note: Details of the origin servers in the origin-server pool are not captured during introspection. The origin servers will be defined during the deployment process, based on the appliance to which the output endpoint of the Oracle Traffic Director instance appliance is wired.

B.10.6 Oracle Traffic Director Appliance Properties

This section describes the editable properties of the appliances in an Oracle Traffic Director assembly. The properties are captured during introspection and are used to create the virtual machines when the appliances are deployed.

This section contains the following subsections:

- Section B.10.6.1, "Editable Properties of the Administration Server Appliance"
- Section B.10.6.2, "Editable Properties of the Instance Appliance"

B.10.6.1 Editable Properties of the Administration Server Appliance

Table B–40 General Properties of the Administration Server Appliance

Name	Type	Req'd	Default	Description
adminUser	String	true	`admin`	The user name for logging in to the administration server.
adminPassword	String	true	None	The password for the administration server user name.

Table B–41 Resource Properties of the Administration Server Appliance

Property Name	Type	Req'd	Default Value	Description
CPU_MHZ	Integer	true	1000	CPU clock speed.
MEMORY_MB	Integer	true	1024	Memory requirement, in megabytes.
NUMBER_CPUS	Integer	true	1	Number of processors.

Table B–42 Scalability Properties of the Administration Server Appliance

Property Name	Type	Req'd	Default Value	Description
Min	Integer	true	1	The minimum number of virtual machines of the appliance that can be deployed.
Max	Integer	true	1	The maximum number of virtual machines of the appliance that can be deployed. This value must be higher than the Absolute-Max value.
Absolute-Max	Integer	false	1	The absolute maximum number of virtual machines of the appliance that can be deployed.
Target	Integer	true	1	The actual number of virtual machines of the appliance to be deployed.

You can change the values of the scalability properties, but Oracle recommends that you leave them at the default values.

B.10.6.2 Editable Properties of the Instance Appliance

Table B–43 Resource Properties of the Instance Appliance

Property Name	Type	Req'd	Default Value	Description
CPU_MHZ	Integer	true	1000	CPU clock speed.
MEMORY_MB	Integer	true	1024	Memory requirement, in megabytes.
NUMBER_CPUS	Integer	true	1	Number of processors.

Table B–44 Scalability Properties of the Instance Appliance

Property Name	Type	Req'd	Default Value	Description
Min	Integer	true	1	The minimum number of virtual machines of the appliance that can be deployed.

Table B–44 (Cont.) Scalability Properties of the Instance Appliance

Property Name	Type	Req'd	Default Value	Description
Max	Integer	true	2	The maximum number of virtual machines of the appliance that can be deployed. This value must not be higher than the Absolute-Max value.
Absolute-Max	Integer	false	2	The absolute maximum number of virtual machines of the appliance that can be deployed.
Target	Integer	true	2	The actual number of virtual machines of the appliance to be deployed.

B.10.7 Supported Template Types

Oracle Enterprise Linux.

B.10.8 Post-Deployment Tasks

After deploying the Oracle Traffic Director assembly, you should perform the following tasks:

- In the reference system, if Oracle Traffic Director is configured as the SSL termination point, then it is assumed that the certificates in the certificate database of the configuration belong to the virtual servers and not to the host name of the reference system. If the certificates belong to the host name of the reference system, then those certificates should be re-installed on the target virtual machines.
- If SSL is configured between Oracle Traffic Director and the Oracle WebLogic Server managed servers, any new certificates, if required, should be installed after deploying the Oracle Traffic Director assembly.
- Failover groups configured in the reference system are not captured during introspection. Therefore, high availability should be configured afresh by using the Oracle Traffic Director administration console or the CLI.

Note: High availability is supported only when your base image includes *keepalived*.

B.11 Oracle Tuxedo Plug-In

The Oracle Tuxedo introspection plug-in examines a single or multiple-machine Oracle Tuxedo domain, and the Oracle Home Directory that it resides on. The Oracle Home Directory where Tuxedo is installed can also include the Tuxedo add-ons listed below, and those will also be examined:

- Oracle TSAM
- Oracle SALT
- Oracle Tuxedo Application Runtime for CICS and Batch
- Oracle Tuxedo Mainframe Adapter SNA

A single machine domain and its Home Directory, including add-on products, are captured. For a multiple-machine Oracle Tuxedo domain, each machine must be introspected separately and wired into an assembly. See Section B.11.6, "Wiring".

B.11.1 Versions Supported

This plug-in supports version 11gR1.

B.11.2 Oracle Tuxedo Introspection Parameters

Table B–45 lists the introspection parameters for the Oracle Tuxedo introspection plug-in:

Table B–45 Oracle Tuxedo Plug-in Introspection Parameters

Parameter	Description
TUXDIR	Location where Oracle Tuxedo is installed.
TUXCONFIG	Location of the application configuration file, in compiled form. This contains the Tuxedo core configuration as well as a minimal set or values (APPDIR, etc.).
environmentScript	This script will be run before doing introspection to set the environment of the Tuxedo application. The script will be searched relative to the `$APPDIR` directory. As a result of running the script only known Tuxedo-related environment variables will be captured, so as to prevent things such as DISPLAY, or SHELL from being captured that could interfere on the target environment. If not set, the plug-in will attempt to run a `setenv.sh` script (with that exact name) from the `$APPDIR` directory. This behavior will be exclusive, that is, only one script will be run at most.The plug-in will use the output of the `env` command, so care should be taken that such environment setting scripts' output may not interfere with the result of calling `env`. If an environment script is not used, or is used but non-Tuxedo environment variables also need to be set, a well-known standard Java properties file named `ovab-application.properties` will be searched for in `$APPDIR`.
oracleClientDir	Location of Oracle Database client software installation, typically the directory in which the Oracle Instant client software has been unzipped. It is the user's responsibility to ensure that this is accurate, as the Tuxedo plug-in does not have the means to verify that this installation is valid.
tnsNamesLocation	Location of the `TNSNAMES.ora` client configuration file. This file is parsed and made a template which results in an Appliance Output being created if either the database name specified in the `ubbconfig OPENINFO` string or environment variable `ORACLE_SID` (in that order or priority) is found. Multiple `OPENINFO/TNSNAMES.ora` entries will result in multiple outputs being generated.
scriptWorkingDir	The working directory where the environment script will be run from. This is useful when scripts use the current working directory to determine path values.
artSecurityProfile	Location of security profile used by ART Batch or ART CICS.

B.11.3 Reference System Prerequisites

None.

B.11.4 Requirements

The following requirements apply to Oracle Tuxedo:

B.11.4.1 Base Image Requirements

The base system image OS version must match the version of the reference system.

Additionally, you must set IPC kernel parameters on the base system image according to the guidelines listed in *Oracle Tuxedo: Installing the Oracle Tuxedo System*: http://docs.oracle.com/cd/E18050_01/tuxedo/docs11gr1/install/insappd.html).

B.11.4.2 ART CICS/Batch Applications Requiring Microfocus or COBOL IT

For ART CICS/Batch applications which require Microfocus or COBOL IT to be installed, you must create a new base image with Microfocus or COBOL IT pre-installed (the installation path is the same as it is on the reference system) based on the original Oracle Virtual Assembly Builder base image, and then use the new base image to create template for the ART CICS/Batch application.

Only by following this configuration procedure will ART CICS/Batch applications which require Microfocus/COBOL IT boot successfully on the deployed VM.

B.11.4.3 Requirements Related to Scaling

For TMA SNA, scaling is not applicable.

For ART Batch, scaling is applicable, except for one limitation: if the TMQUEUE server which monitors JES2QSPACE queue space runs on a slave machine, you should not use the scaling feature for that machine for ART Batch.

For ART CICS, not all servers are applicable for scaling. Refer to the ART CICS reference guide to determine whether scaling is applicable or not for specified servers of ART CICS.

B.11.5 Resulting Artifact Type

The resulting artifact type depends on whether you introspect a single-machine or multi-machine domain.

B.11.5.1 Single-Machine Oracle Tuxedo Domain

A single scalable appliance for a single-machine Tuxedo domain.

B.11.5.2 Multi-Machine Oracle Tuxedo Domain

For multi-machine Tuxedo domains, each machine in the reference system must be introspected separately. The resulting appliances are of the following types:

- Master: single non-scalable appliance representing the MASTER node in the Tuxedo domain.
- Backup master (optionally): single non-scalable appliance representing the BACKUP MASTER node in the Tuxedo domain. Must be introspected if present on the reference system.
- Other: single scalable appliance representing a non-master and non-backup node in the reference Tuxedo domain. There can be one or many such types of appliances depending on the topology of the reference system.

To deploy the domain, an empty assembly must be created manually, or the appliances must be included in an existing assembly and the wiring performed.

B.11.6 Wiring

This section describes wiring.

B.11.6.1 Multi-Machine Wiring

Inputs will be created on a Master appliance for each machine (except itself) present in the reference system. These are required for non-Master appliances to obtain information on the Master appliance at rehydration time.

Outputs will be created on a Master appliance for each machine (except itself) present in the reference system. These are required for the Master appliance to obtain information on the non-Master appliances at rehydration time. Corresponding inputs and outputs will also be created on non-Master appliances.

These outputs must all be connected to an appliance before deployment. The name of the output and the protocol supported by the output will give hints about the type of appliance to connect the output to.

B.11.6.2 Other Inputs and Outputs

Inputs will be created on an Oracle Tuxedo appliance for the following types of configuration found:

- WSL (Oracle Tuxedo WorkStation protocol)
- JSL (Oracle Jolt)
- ISL (Oracle Tuxedo IIOP protocol)
- Domain (Oracle Tuxedo Domain Gateway)

Outputs will be created on an Oracle Tuxedo appliance for the following types of configuration found:

- Domain (Oracle Tuxedo Domain Gateway)
- Oracle Single-Instance Database
- TMA_SNA (Oracle Tuxedo Mainframe Adapter SNA)
- TSAM (Oracle TSAM)

These outputs must all be connected to either an external resource or to an appliance before deployment. The description on the output and the protocol supported by the output will give hints about the type of appliance to connect the output to.

B.11.7 Wiring Properties

All input endpoints have two editable properties - `port` and `description`, and one non-editable property - a list of `protocols`. The protocols indicate what sort of outputs can be connected to the input.

All output endpoints have one editable property - `description`, and two non-editable properties - `protocol` and `singleton`. The protocol indicates what sort of input can be connected to the output. Singleton indicates what sort of appliance the output can be connected to. If singleton is true, the output can only be connected to an input on an appliance that has a scalability absolute max value of 1.

The following properties are specific to Oracle Tuxedo endpoints (Table B–46 through Table B–48):

Table B–46 Oracle Tuxedo: Appliance Output Properties: Domain

Name	Type	Req'd	Default	Description
existing-address	String	false	Address of the remote domain from the reference system.	Specifies the address of the remote domain this domain will connect to. Only used if the output is connected to an external resource.

The output for TMA_SNA and TSAM can only be connected to an external resource.

Table B–47 Oracle Tuxedo: Appliance Output Properties: TMA_SNA

Name	Type	Req'd	Default	Description
tma-sna-crm-host	String	false	IP address of remote CRM Server from reference system.	Specifies the IP address of the remote CRM Server this machine will connect to. Only used if the output is connected to an external resource.
tma-sna-crm-port	String	false	Port of remote CRM Server from reference system.	Specifies the port of the remote CRM Server this machine will connect to. Only used if the output is connected to an external resource.
tma-sna-crm-address	String	false	Hex format IP Address and port of remote CRM Server from reference system.	Specifies Hex format IP Address and port of remote CRM Server this machine will connect to. Only used if the output is connected to an external resource.

Table B–48 Oracle Tuxedo: Appliance Output Properties: TSAM

Name	Type	Req'd	Default	Description
tsam-manager-addr	String	false	IP Address of remote TSAM manager from reference system.	Specifies the IP Address of remote TSAM manager this machine will connect to. Only used if the output is connected to an external resource.
tsam-manager-port	String	false	Port of remote TSAM manager from reference system	Specifies the port of remote TSAM manager this machine will connect to. Only used if the output is connected to an external resource.

B.11.8 Oracle Tuxedo Appliance Properties

Oracle Tuxedo appliances have user properties (Table B–49) and system properties (Table B–50).

Table B–49 Oracle Tuxedo: User Properties

Name	Type	Req'd	Default	Description
ALOGPFX	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
ALOGRTNSIZE	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
ALTCC	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
ALTCCFLAGS	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
applicationEnvVars	String	false	none	Applications can use this property to specify non-Tuxedo variables using comma-separated keyword/value pairs. For example: CURRENCY=dollar,GROUPNAME=stdev,JDK=/my/jdk/path. This property is populated by the ovab-application.properties file, if it exists in the `$APPDIR` directory.
applicationPassword	String	false	none	If the Tuxedo application uses security (that is, `*RESOURCES` is set to `APP_PW`, `USER_AUTH`, `ACL` or `MANDATORY_ACL`) then this user property must be set to capture the new password to be used at reconfiguration time.
COBCPY	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
COBDIR	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
COBOPT	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.

Table B–49 (Cont.) Oracle Tuxedo: User Properties

Name	Type	Req'd	Default	Description
dbPassword	String	false	none	Replacement value for database username when Tuxedo application has an OPENINFO set for Oracle databases (RM type of Oracle_XA in OPENINFO.) For example: For the following OPENINFO value:"Oracle_XA: Oracle_ XA+Acc=P/Scott/*****+SesTm=30+S qlNet=instance1", the *dbPassword* property may be set in which case it is used to regenerate a new encrypted password.
dbUsername	String	false	none	Replacement value for database username when Tuxedo application has an OPENINFO set for Oracle databases (RM type of Oracle_XA in OPENINFO.) For example: For the following OPENINFO value:"Oracle_XA: Oracle_ XA+Acc=P/Scott/*****+SesTm=30+S qlNet=instance1"the *dbUsername* property may be set to change "Scott" into a different value for the target machine.
FIELDTBLS	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
FIELDTBLS32	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
FLDTBLDIR	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
FLDTBLDIR32	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
FSCONFIG	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
FSMAXCOMMIT	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
FSMAXUPDATE	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.

Table B–49 (Cont.) Oracle Tuxedo: User Properties

Name	Type	Req'd	Default	Description
FSMSGREP	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
FSOFFSET	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
ISSANE	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
QMCONFIG	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
runtimeLoadLibraryPath	String	false	none	Populated with the contents of LD_LIBRARY_PATH after setting the environment. The format of this string is be the same as the actual LD_LIBRARY_PATH to be used on the target system.
shutdownScript	String	false	none	The name of a shutdown script that will be used in place of the `tmshutdown -y` command used on the target machine when it is stopped (after undeployment, or as a result of an Oracle Virtual Assembly Builder `stop` command).
startupScript	String	false	none	The name of a startup script that will be used in place of the `tmboot -y` command used on the target machine when it is started (after deployment or as a result of an Oracle Virtual Assembly Builder `start` command).
TAGENTLOG	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TM_CBL_IGNORE_CONTEXT	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TM_CPAU	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TM_ENGINE_TMSHMSEGSZ	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TM_GWT_OLDSECCHECK	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.

Table B–49 (Cont.) Oracle Tuxedo: User Properties

Name	Type	Req'd	Default	Description
TM_ICU_ COMPATIBILITY	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TM_LOG_ESYS	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TM_ORB_ CLTMAXRTY	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TMCMPLIMIT	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TMCMPPRFM	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TMNETLOAD	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TMNOTHREADS	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TMSICACHEENTRIE SMAX	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TMUSEIPV6	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TPMBACONV	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TPMBENC	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TUX_BLOCKLICIW	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
TUX_SSL_ ENFORCECONSTRA INTSUINMEDSIGS	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.

Table B–49 (Cont.) Oracle Tuxedo: User Properties

Name	Type	Req'd	Default	Description
URLENTITYCACHEDIR	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
URLENTITYCATCHING	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
VIEWDIR	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
VIEWDIR32	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
VIEWFILES	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
VIEWFILES32	String	false	none	Tuxedo environment variable. See "tuxenv(5)" in *Oracle Tuxedo File Formats, Data Descriptions, MIBs, and System Processes Reference*.
KIX_TS_DIR	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
KIX_TD_DIR	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
KIX_TD_QSPACE_DEVICE	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
KIX_TD_QSPACE_NAME	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
KIX_TD_QSPACE_IPCKEY	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
KIX_TECH_DIR	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.

Table B–49 (Cont.) Oracle Tuxedo: User Properties

Name	Type	Req'd	Default	Description
KIX_CWA_SIZE	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
KIX_CWA_IPCKEY	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
KIX_QSPACE_ IPCKEY	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
KIX_TRACE_LEVEL	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
KIX_MAP_PATH	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
DATA	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
SPOOL	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
TMP	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
PROCLIB	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
MT_ACC_FILEPATH	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
MT_DB_LOGIN	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.

Table B–49 (Cont.) Oracle Tuxedo: User Properties

Name	Type	Req'd	Default	Description
MT_LOG	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
MT_TMP	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.
MT_KSH	String	false	none	Tuxedo ART CICS environment variable. See "CICS Runtime Environment Variables" in *Oracle Tuxedo Application Runtime for CICS Reference Guide*.

Table B–50 Oracle Tuxedo System Properties

Name	Type	Req'd	Default	Description
appdir	String	false	none	Application Directory, location of the Tuxedo application executables and files.
masterTuxconfig	String	false	none	Location of the TUXCONFIG file for the Master machine in a multi-machine domain. This is necessary to perform scale-out operations.
masterTuxdir	String	false	none	Location of TUXDIR file the Master machine in a multi-machine domain. This is necessary to perform scale-out operations.
model	String	false	none	Indicates whether this appliance is a single-machine appliance (SHM) or multi-machine appliance (MP).
pmid	String	false	none	Oracle Tuxedo Machine identifier for this appliance.
role	String	false	none	Along with model, is used to qualify the type of appliance when it is part of a multi-machine domain. The role can be 'MASTER', 'BACKUP' or 'SLAVE'. It is always 'SLAVE' for a single-machine domain.
tuxconfig	String	false	none	Used to save the value of TUXCONFIG as introspected.
tuxdir	String	false	none	Used to save the value of TUXDIR as introspected.
kixdir	String	false	none	Used to save the value of KIXDIR as introspected.

Table B–50 (Cont.) Oracle Tuxedo System Properties

Name	Type	Req'd	Default	Description
kixconfig	String	false	none	Used to save the value of KIXCONFIG as introspected.
jesdir	String	false	none	Used to save the value of JESDIR as introspected.

B.11.9 Extensions of the Plug-in

None.

B.11.10 Supported Template Types

The supported template type is Oracle Enterprise Linux (OEL).

B.12 Generic Appliance Plug-in

The generic appliance introspection plug-in allows you to create an appliance that gets configured and deployed using scripts supplied during introspection. The generic appliance introspector plug-in reads and collects the properties of an opaque, standalone, and self-contained product or application, and captures the set of files that make up the product as specified by the user. The output of the plug-in is an appliance.

A generic appliance does not make use of product-specific plug-in code to capture configuration or product location--instead a simple appliance is created and a set of user-supplied properties, paths, and scripts are added to it in a generic manner. The set of scripts passed in at creation are executed at deployment to perform the necessary operations.

B.12.1 Requirements

The following requirements apply to generic appliances:

B.12.1.1 Specify File Sets

You must specify a list of directories to be captured along with all the file and directories underneath. This capability provides the means by which installation binaries, configuration, and data is captured.

B.12.1.2 Scripts Are Launched As Root

All scripts will be launced as the root user. This provides generic appliance scripts the flexibility of performing operations requiring root privileges or switching to another user as desired.

B.12.2 Resulting Artifact Type

A single appliance.

B.12.3 Generic Appliance Plug-in Introspection Parameters

Table B–51 lists the introspection parameters for Oracle Database:

Table B–51 Generic Appliance Plug-in Introspection Parameters

Parameter	Description
productRoots	A required list of one or more colon-separated paths of type `string`. Each path becomes a root of a `FileSet` within the eDefinition of the Appliance. All files within each specified root are captured during a e operation. Each path gets mapped to its own filesystem on the VM at deployment.
propertyFile	Optional. Absolute path of the properties file. Each property in the specified file becomes a user property in the appliance metadata.
scriptRootDir	Optional. Path to the root directory where the scripts are obtained. Scripts are located in subdirectories within this root directory according to operation type.

B.12.4 Property File

If you specify the `propertyFile` parameter, you must reference a file that exists on the reference system and is readable (otherwise, a failure results and the appliance is not created).

A property file is a text file containing a list of (name, value) pairs. Each property in the property file is added as a user property into the appliance. At deployment the user properties in the appliance are written back out to a file with the absolute path indicated by the `$AB_USERPROPS_FILE` environment variable.

A property file must consist of zero or more lines where each line is a property declaration, a comment, or a blank line. More formally, a property file must comply with the following syntax:

Example B–5 Property File Syntax

```
property-file   = *line
line            = prop-decl | comment | blank-line
prop-decl       = name "=" value NL
comment         = *WS "#" *CHAR NL
blank-line      = *WS NL
name            = name-start-char *name-body-char
name-start-char = <any character in "a".."z", "A".."Z", "_">
name-body-char  = <any character in "a".."z", "A".."Z", "0".."9", "_">
value           = *SHCHAR | SQ *SHCHAR SQ | DQ *SHCHAR DQ
NL              = <platform dependent line termination sequence>
WS              = <white space character>
CTL             = <any control character (octets 0 -31) and DEL (127)>
CHAR            = <any character, excluding CTL (and NL), but including WS>
SHCHAR          = <any CHAR, escaped as necessary for shell interpretation>
SQ              = <single quote>
DQ              = <double quote>
```

Any property file that does not comply with the above syntax rules results in an error, and an appliance is not created. Property declarations must be contained on a single line. Ending a line with a slash ("\") does not result in line continuation.

All properties will be marked as "required" in the appliance metadata. Property declarations without any assigned value (nothing after "=") will be set to null in the appliance metadata requiring that the user assign a value to that property prior to deployment.

Whitespace is not permitted anywhere to the left of the equal sign ("=") in a property declaration. Whitespace to the right of the equal sign is assumed to be part of the intended value and is preserved (resulting in a failure if the value is sourced).

Quotes around property values are preserved and are visible to users as part of the value. When editing a property value, it is the responsibility of the user to add, remove, or preserve quotes as necessary according to the rules of shell interpretation.

Comments and blank lines are discarded at dehydration and are not reproduced when the file is regenerated at rehydration.

Typically, a generic appliance script reads the property file into the script environment. A common usage pattern is:

```
#!/bin/bash
#
# This script reconfigures the example server of the
# example product.
#
. $AB_USERPROPS_FILE
$ORACLE_HOME/bin/oim_reconfig.sh $OIM_INSTANCE
```

Here is sample content of a valid properties file:

```
# The following property must have a user supplied value
SHELL=
# This is a variable that should not be changed
PRODUCT_HOME=/my/install/will/not/move
PRODUCT_INSTANCE=/my/instance/will/also/not/move
PRODUCT_PROPERTY="Hello World"
# This is a mispelled variable name
TRUSTROTE=/path/to/file.jks
```

Given the above property file example (including user edits of some values), the following property file content is generated during reconfiguration:

```
SHELL=/bin/bash
PRODUCT_HOME=/my/install/will/not/move
PRODUCT_INSTANCE=/my/instance/will/also/not/move
PRODUCT_PROPERTY="Yo, peoples of planet Earth!"
TRUSTROTE=/path/to/file.jks
```

B.12.4.1 Script Root Directory

The script root directory is the top level directory containing the script subdirectories. If the specified directory does not exist or is not readable then an error will be returned and an appliance will not be created

User supplied reconfiguration scripts must be placed within the root script directory under the following well-known subdirectories: config.d/, start.d/, ping.d/, stop.d/. Scripts under each subdirectory will be captured during dehydration and stored in the appliance. During rehydration the appropriate set of scripts according to the requested operation will be executed in lexicographical order (same order as /bin/ls).

The following is an example of the set of script directories the user might create:.

```
/path/to/script/dirs/
     config.d/
         00config.sh
         01.config.sh
     start.d/
         00start.sh
         01start.sh
```

```
stop.d/
     stop.sh
ping.d/
     ping.sh
```

Any file or directory located in the script root directory other than the set of well-known subdirectories will be ignored and will not be captured during dehydration.

The script root directory need not contain all well-known subdirectories. The omission of a well-known subdirectory is ignored during dehydration with the assumption that no script is needed for that particular phase.

A well-known subdirectory may be empty. An empty well-known subdirectory will not be captured.

Well-known subdirectories must only contain scripts that should be launched by the generic appliance plug-in. The presence of a directory within a well-known subdirectory will generate an error during dehydration and an appliance will not be created. Everything else will be captured and the generic appliance will attempt to execute it during rehydration. Files such as data files, configuration files, and videos will likely fail to execute properly resulting in an overall failure of the corresponding operation. Such files are more appropriately captured using the `productRoots` parameters.

B.12.5 Wiring

No wiring can be performed for generic appliances.

B.12.6 Extensions of the Plug-in

None.

B.12.7 Supported Template Types

The supported template type is Oracle Enterprise Linux (OEL).

C

Common Properties for Oracle Virtual Assembly Builder Components

This appendix describes common properties for components that Oracle Virtual Assembly Builder can introspect, and other properties that can be specified for deployment. It contains the following sections:

C.1 Common Properties

The following OCM-related properties are common to all appliances.

Table C–1 OCM-related Common Properties

Name	Req'd	Default	Description
ocm.anonymousEmailRegistration.emailId	false	none	Email address to use to register with OCM using an email address that is not associated with a metalink account.
ocm.metalinkCsiRegistration.CSI	false	none	Register deployments using a Customer Support Identifier.
ocm.metalinkCsiRegistration.countryCode	false	none	Two-letter country code associated with the CSI.
ocm.metalinkCsiRegistration.metalinkId	false	none	Metalink ID associated with the CSI.
ocm.metalinkEmailRegistration.metalinkEmailId	false	none	Register deployments using an email ID associated with a metalink account.
ocm.metalinkEmailRegistration.metalinkPassword	false	none	Password associated with the metalink account.
ocm.proxyHost	false	none	Required when OCM registration must occur through a proxy.
ocm.proxyPassword	false	none	Required when OCM registration must occur through a proxy.
ocm.proxyPort	false	none	Required when OCM registration must occur through a proxy.

Table C–1 (Cont.) OCM-related Common Properties

Name	Req'd	Default	Description
ocm.proxyUsername	false	none	Required when OCM registration must occur through a proxy.
ocm.repeaterURI	false	none	For use when registering through a configured OCM hub.
ocm.runConfiguration	false	false	Set to true in order to perform OCM registration.

C.2 System Properties

The following OCM-related system properties are common to all appliances. They cannot be modified by users.

Table C–2 OCM-related System Properties

Name	Req'd	Default	Description
ocm.ccrDirPath.0	true	sample value: /swat/middleware_ps1/utils/ccr	Not to be edited by users.

C.3 External Resource Properties

External resources represent services to which an Oracle WebLogic Server domain connects.

C.3.1 Common Properties

All external resource appliances have the properties described in Table C–3 (`hostname` is a user property and `external-appliance` is a system property).

Table C–3 External Appliance Template Properties: Common Properties

Name	Type	Req'd	Default	Description
hostname	String	false	none	The hostname where the service the external appliance is representing resides. By default this value is unset in the external appliance templates. You must provide a value before deployment.

C.3.2 foreignJMS Properties

Table C–4 describes properties for external appliances to connect a foreign JMS output on an Oracle WebLogic Server Admin server.

Table C–4 foreignJms Properties

Name	Type	Req'd	Default	Description
url	String	false	none	The URL used to connect to the foreign JMS server. If not specified in the template then the value from the reference system is retained when the Oracle WebLogic Server domain is deployed.

Table C–4 (Cont.) foreignJms Properties

Name	Type	Req'd	Default	Description
Password	String	false	none	The password used to connect to the foreign JMS server. If not specified in the template then the value from the reference system will be retained when the Oracle WebLogic Server domain is deployed.

C.3.3 jmsBridgeDestination Properties

Table C–5 describes properties for external appliances to connect a JMS message bridge output on an Oracle WebLogic Server Admin server.

Table C–5 jmsBridgeDestination Properties

Name	Type	Req'd	Default	Description
url	String	false	none	The URL used to connect to the JMS bridge destination server. If not specified in the template then the value from the reference system is retained when the Oracle WebLogic Server domain is deployed.
Username	String	false	none	The username used to connect to the JMS bridge destination server. If not specified in the template then the value from the reference system is retained when the Oracle WebLogic Server domain is deployed.
Password	String	false	none	The password used to connect to the JMS bridge destination server. If not specified in the template then the value from the reference system is retained when the Oracle WebLogic Server domain is deployed.

C.3.4 LDAP Properties

Table C–6 describes properties for external resources to connect an LDAP output on an Oracle WebLogic Server Admin server.

Table C–6 LDAP Properties

Name	Type	Req'd	Default	Description
Username	String	false	none	The username used to connect to the LDAP server. If not specified in the template then the value from the reference system is retained when the Oracle WebLogic Server domain is deployed.
Password	String	false	none	The password used to connect to the LDAP server. If not specified in the template then the value from the reference system is retained when the Oracle WebLogic Server domain is deployed.

C.3.5 Non-Oracle JDBC Properties

Table C–7 describes properties for external resources to connect a non-Oracle JDBC output on an Oracle WebLogic Server Admin server.

Table C–7 Non-Oracle JDBC Properties

Name	Type	Req'd	Default	Description
url	String	false	none	The URL used to connect to the non-Oracle database. If not specified in the template then the value from the reference system is retained when the Oracle WebLogic Server domain is deployed.

C.3.6 JDBC Properties

Table C–8 describes properties for external resources to connect an Oracle JDBC output on an Oracle WebLogic Server Admin server.

Table C–8 JDBC Properties

Name	Type	Req'd	Default	Description
global-db-name	String	false	orcl	The global-db-name needed to connect to the Oracle database. If not specified in the external resource user properties, then the deployment will fail.

C.4 Deployer Properties

The Oracle Virtual Assembly Builder allows you to configure the Deployer properties described in Table C–9 through the deployer.properties file. This file must reside in the `<deployer WLS domain dir>/ovab/config` directory.

Table C–9 Deployer Properties Configurable in deployer.properties

Name	Type	Req'd	Default	Description
phoneHomeTimeout	String	false	900	Configures the phone home request timeout, in seconds.

D

Troubleshooting

The following appendix describes techniques for troubleshooting Oracle Virtual Assembly Builder, and identifies troubleshooting items:

D.1 General Issues

This section describes troubleshooting for general issues.

D.1.1 Error Indicating Another Client is Running

If you see this error when attempting to launch `abctl` or abstudio.sh and you are sure there is no other Oracle Virtual Assembly Builder client running, your machine may have run out of file locks.

You may also see such an error when performing concurrent operations through Oracle Virtual Assembly Builder Studio, or `abctl`. To resolve this problem, check whether there is an environmental issue causing locks to release slowly.

D.1.2 Phone Home Timeouts

To troubleshoot phone home timeouts:

1. Make sure the firewall on your Oracle Virtual Assembly Builder host is turned off.

   As root run the following:

   ```
   $/sbin/service iptables status
   $/sbin/service iptables stop
   ```

   The firewall may be configured to start automatically when your machine reboots. To prevent this you can turn it off completely by running:

   ```
   $/sbin/chkconfig iptables off
   ```

2. The default phone home timeout is 15 minutes. You can increase the timeout in `<Deployer WLS domain root>/ovab/config/deployer/deployer.properties`.

   If the file does not exist, create it and add a line similar to `phoneHomeTimeout=<number of seconds>`.

D.1.3 Existing Assembly Archive (OVA file) Prevents Altering of Assembly

If you have built an assembly archive for your assembly then that assembly is locked - meaning you cannot alter its shape, properties, packages or templates until you delete the archive. The archive can be deleted using `abctl delete -archiveOnly -name <assembly name>`.

D.2 Introspection and File Set Capture Failures

This section describes troubleshooting for introspection and file set capture issues.

D.2.1 Introspection of a VM

Introspection of a deployed virtual machine (VM) can fail to complete when the temporary directory used for introspection is a mounted NFS share with NFS file locking. To avoid this problem, remount the NFS file system with locking turned off:

```
mount -o nolock example:/scratch /net/example/scratch
```

D.2.2 Remote Operation Failures

The logs for remote operations are copied at the end of a remote operation into the local directory of `$AB_INSTANCE/logs/remote_<remote machine name>/`. For example, if the remote machine is abc12345, the logs are stored in `$AB_INSTANCE/logs/remote_abc12345.example.com/`

The default remote working dir is `/tmp/abRemote_<remote username>`, but this can be overridden using the `-remoteWorkingDir` flag.

D.2.2.1 Unable to Connect Errors When Running ipv6 on the Remote Machine

If you are unable to perform remote introspection/packaging and get 'unable to connect' errors, one possibility is that you are running ipv6 on the remote machine but the `sshd_config` file is incorrect.

When remote systems are configured to use ipv6 and ipv4, you must have the following line in the `sshd_config` file:

```
AddressFamily any
```

(and not `AddressFamily inet`).

D.2.2.2 Remote Operation Hangs after Entering Password

If your remote operation hangs after you enter the remote password it may be due to an orphaned remote process left over from a previous remote operation that was killed. If the remote working directory is removed out from underneath the remote process this can happen. Go to the remote machine and kill any orphaned remote processes, then clean up the remote working dir and try your remote operation again.

The remote process appears similar to the following:

```
aime1    11662     1  0 11:51 ?        00:00:01 /tmp/user/abRemote_aime1/ab_
```

```
home/jre/jre/bin/java
  -Doracle.core.ojdl.logging.config.file=/tmp/user/abRemote_aime1/ab_
instance/config/logging.xml
  -Djava.util.logging.config.class=oracle.core.ojdl.logging.LoggingConfiguration
  -Djava.security.egd=file:/dev/./your -Dassemblybuilder.spif.app=apps/remotingapp
  -jar /tmp/kaw/abRemote_aime1/ab_home/jlib/oracle.as.assemblybuilder.spif_
0.1.0.jar
```

D.2.2.3 File Permission Problems

Make sure the remote user or sudo user you specify for the remote operation has read permissions for the files in the reference installation.

D.2.2.4 Remote Connection Failure

SSH port forwarding must be enabled on reference systems in order for remote operations (introspection and file set creation) to work properly. Check the ssh config files:

```
~/.ssh/config
/etc/ssh/ssh_config
```

The following error can be encountered if the shell of the remoteUser specified prints things to stdout/stderr during login.

```
Error: Error initializing the remote connection.
Caused by: OAB-90061: Unable to create connection to remote server.
Cause: Timed out trying to connect to IPV4 and IPV6 sockets.
```

Check the profile and rc files for the remote user and take out any logic that does this. Alternatively, you can specify a different remote user and use the sudoUser parameter to specify a user that has permission to examine/capture the reference installation.

D.2.2.5 Remote File Set Capture Failure

The remote working directory must have enough disk space available to store your file sets before they are transferred back to the local machine.

D.3 Template Creation Failures

When a template creation operation fails, check the following:

- Verify you ran `$ORACLE_HOME/oracleRoot.sh` as root during or after installation.
- Verify that modifyjeos is installed.
- Verify that the ova utility is installed.
- Verify that you have a valid base image (System.img) and vm.cfg file. Verify that file permissions are correct.
- Verify that you did not run out of disk space.
- Verify that you have a sufficient number of loop devices for the file sets you are capturing. See Section D.3.1, "Insufficient Number of Loop Devices".

D.3.1 Insufficient Number of Loop Devices

You may run into an issue where the number of Linux loop devices on an Oracle Virtual Assembly Builder host are not sufficient to create templates for a generic product with a large number of file sets.

When creating templates, Oracle Virtual Assembly Builder and modifyjeos require one available Linux loop device for each disk in the template (that is, one each for the System.img and AB.img, and one per product disk). A typical Oracle Linux system has only seven loop devices, meaning templates can be created for a template with a maximum of five file sets.

To create templates for an appliance with more file sets, you must create additional loop devices. One way to do this is as follows:

1. Edit /etc/modprobe.conf. Add a line similar to the following:

   ```
   options loop max_loop=<n>
   ```

   Where <*n*> is the number of loop devices you want created.

2. As root, run the following commands to unload and reload the Linux kernel loop module:

   ```
   # /sbin/modprobe -r loop
   # /sbin/modprobe -v loop
   ```

3. Verify that the new loop devices were created:

   ```
   $ ls -l /dev/loop*
   ```

   You should see <n> loop devices.

D.4 Deployer Communication Failures

This section describes troubleshooting of Deployer communication failures.

D.4.1 Invalid Deployer Response Returned

You may run into the following error:

```
Caused by: Invalid deployer response returned.
  Cause: OAB-113409 - An invalid response was returned by the deployer.
  Action: OAB-113409 - Please check the deployer log for additional details.
```

Check the Deployments for the Oracle WebLogic Server hosting the Deployer and make sure the state of the Deployer application is 'Active' and its health is 'OK'.

D.4.2 401/403 Errors from the Deployer

if you run into 401/403 errors when trying to interact with the Deployer, check the following items in the console of the Oracle WebLogic Server hosting the Deployer.

1. Go to **Deployments** > **Deployer** and verify that the Security Model is "CustomRoles."
2. Go to **Deployments** > **Deployer** > **[Security]** > **[Roles]** > **Application Admin** and make sure that conditions include Group: Cloud Admins or Application Admins.
3. Go to **Deployments** > **Deployer** > **[Security]** > **[Roles]** > **Cloud Admin** and ensure that the conditions include Group: Cloud Admins.

4. Go to **Security Realms** > **myrealm** > [Users and Groups] > [Groups] and ensure that "Application Admins" and "Cloud Admins" exist and are handled by DefaultAuthenticator.
5. Go to **Security Realms** > **myrealm** > [Users and Groups] > [Users] and make sure that "applicationAdmin" and "cloudAdmin" exist and are handled by DefaultAuthenticator.
6. Go to **Deployments** > **Deployer** > [Security] > [URL Patterns] and make sure there is no role definition on `root url-pattern`.
7. When creating a connection in the client, make sure the username is one of the two listed in step 5 and is specified with the password for that user.

D.5 Registration Failures

If your registration has become unresponsive, check the following:

- If you have never had a successful registration, try to register a template using the Oracle VM console directly - bypassing Oracle Virtual Assembly Builder. If this is not successful then the problem is with your Oracle VM environment.
- If you have a very large assembly archive or a slow network you may need to increase the 'Stuck Thread Max Time' setting for both Oracle WebLogic Server where the Deployer is running, and Oracle WebLogic Server running Oracle VM Manager. Access this setting through the admin console, in the *Tuning* tab for the server.

D.6 Deployment Failures

This section describes troubleshooting of deployment failures.

If you cannot determine the cause of the failure from the Studio or Deployer logs you'll have to continue investigating. Log in to the Oracle VM Manager console and see if the VMs for your assembly were created and started.

D.6.1 VM Not Created

Check the Deployer and the Oracle VM and Oracle VM Server logs for an indication of why the VMs were not created.

Note: It is possible for the Deployer's state cache to get out of sync from the Oracle VM environment, especially if cleanup type activity was done in the Oracle VM environment outside of Oracle Virtual Assembly Builder. The Deployer may have recorded that an assembly is still registered or deployed, when it has actually been removed from the Oracle VM environment. If this has happened, you must unregister and undeploy your archive through Oracle Virtual Assembly Builder and re-register and redeploy it.

D.6.2 VM Created, But Not Running

Perform the following steps if the VM is created, but not running:

1. Check the Deployer, and Oracle VM and Oracle VM Server logs, for an indication of why the VMs were not started.
2. Try starting the VM manually and see if any useful output is given.

1. Log in to the Oracle VM Server machine that created the failed VM (you can see which machine in the pool owns the VM via the Oracle VM Manager console)
2. Find the vm.cfg for the VM in question - it will be in some location under /OVS/Repositories, and will have the ID from the Oracle VM console in its path.
3. Use the `xm create -f <vm.cfg file>` command to start the VM.

3. Try mounting the disk images and see if any logs were created. (This can be the case if the VM came up but then went back down for some reason). An Oracle Enterprise Linux image is composed of multiple disks; you must mount the disk you are interested in: such as, AB.img, System.img, or Product_001.img.
   1. The Oracle Virtual Assembly Builder logs are on the AB.img disk - but the name of that image changes once it is registered in the Oracle VM environment. To find the location of the newly named image you need to find the vm.cfg. It will be in some location under `/OVS/Repositories`, and have the VM ID from the Oracle VM console in its path.
   2. Figure out the loop device using the path to the image file you just found.

      ```
      #kpartx -a <path to img file>
      #kpartx -l <path to img file>
      ```

   3. From the listing of the previous command, you can determine which loop device has been mapped to the disk. Mount the disk and specify the loop device.

      ```
      #mount /dev/mapper/loop?p? /mnt
      ```

      The first question mark ('?') above represents the number of the loop device, which is usually zero but may be a different number, and the second question mark is the partition number, which is usually zero but may be a different number.
   4. Go to `/mnt` and look at the files. The AB.img may have logs in `/mnt/logs`, if reconfiguration got far enough to create them.
   5. Enter:

      ```
      #umount /dev/mapper/loop?p?
      #kpartx -d AB.img
      ```

D.6.3 VM Created and Running But Cannot be Pinged

The network configuration for the machine did not complete successfully for some reason. If you are using DHCP make sure your Oracle VM environment supports it. If using static IP addresses make sure you have also specified the corresponding hostname in your deployment plan - this is required.

You must specify all of the following network related properties in your deployment plan:

- at the assembly level
  - network_name (needs to match the name of a network in your target (Oracle VM) environment)
- in the network properties for each appliance
  - hostname (if using static IPs)

- default-gateway
- dns-domains (only one is supported)
- dns-servers (only one is supported)

- on each network interface (NIC) for each appliance
  - ip_address (if using static IPs)
  - netmask
  - usedhcp (should be *false* if using static IPs)

D.6.3.1 How to Access a Running VM that Cannot be Pinged

To access the VM:

1. Log in to the Oracle VM Server machine that created the failed VM (you can see which machine in the pool owns the VM via the Oracle VM Manager console.
2. Run the command `xm list`.
3. Find your VM in the list returned - look on the Oracle VM Manager console for the VM ID.
4. Run the command `xm console <vm ID>` and then hit enter and provide credentials (`user`: root, `password`: the password supplied during template creation).

D.6.3.2 Triaging a Network Configuration Failure

To triage the failure:

1. Check the logs under /assemblybuilder/logs. If there are no logs proceed to the next step.
2. Check to see if the ab service was installed. The ab service is installed to `/etc/init.d/ab`. If it is not there, look at the oraclevm-template service log: `/var/log/oraclevm-template`. The oraclevm-template service installs the ab service.

   If the ab service is missing, make sure the permissions on the ab_service.sh and oraclevm-template.sh in your ORACLE_HOME are correct. These files should be executable. If they are not: fix the permissions, recreate your assembly archive, upload, register and try the deployment again.
3. If you believe the late bindings are incorrect or were not sent to the VM you can run the command `/assemblybuilder/etc/vmapi get +`.

   This command will output the late bindings.

D.6.3.3 VM is Created, Started and Can be Pinged

If the VM is up and pingable, then the network configuration for the VM completed successfully.

1. Log in to the failed VM and check the logs under /assemblybuilder/logs. See the Logs section below for details on what is in the various log files. You should be able to ssh to the machine using the root user and the password you specified when creating the templates.
2. If you believe the late bindings are incorrect or were not sent to the VM you can run the command `/assemblybuilder/etc/vmapi get +`.

   This command will output the late bindings.

D.7 Log Locations and Descriptions

This section provides log locations and descriptions. There are several different logs and log locations that are useful to know about when triaging failures.

D.7.1 Studio Logs

The log level for the Studio log can be altered by editing $AB_INSTANCE/config/logging.xml. Change the following line by indicating the desired level:

```
<logger name="oracle.as.assemblybuilder" level="FINE">
```

Local Logs

$AB_INSTANCE/logs/assemblybuilder.log

$AB_INSTANCE/logs/bottler/* - output from the modifyjeos tool used during template creation

Remote Logs

The logs for remote operations are copied at the end of a remote operation into the local directory of `$AB_INSTANCE/logs/remote_<remote machine name>/`. For example, if the remote machine is abc12345, the logs are stored in `$AB_INSTANCE/logs/remote_abc12345.example.com/`

The default remote working dir is `/tmp/abRemote_<remote username>`, but this can be overridden using the `-remoteWorkingDir` flag.

D.7.2 Deployer Logs

The Deployer application log messages will be in the server and/or domain logs for the WLS where the Deployer application is deployed. Stdout/stderr for the Oracle WebLogic Server may also contain relevant information or stack traces.

<Deployer WLS domain root>/servers/<server targeted by Deployer app>/logs/*

D.7.3 Oracle VM Logs

Oracle VM

/u01/app/oracle/ovm-manager-3/machine1/base_adf domain/servers/AdminServer/logs/AdminServer.log

Oracle VM Server

/var/log/ovs-agent.log

D.7.4 Logs on the VM Instance

/assemblybuilder/logs/ab.out - stdout/stderr and progress messages from Oracle Virtual Assembly Builder infrastructure code and plug-in code

/assemblybuilder/logs/assemblybuilder.log - log messages from Oracle Virtual Assembly Builder infrastructure code and plug-in code

/assemblybuilder/logs/command.out - environment and command details for commands launched via the RehydrateUtils.runCommand() or runCommandAs() methods

/assemblybuilder/logs/proc.<unique>.log - stdout/stderr from processes launched via RehydrateUtils.runCommand() or runCommandAs() where the daemon flag passed in was true

E

Third-Party Licensing

The following appendix contains third-party licensing information. It contains the following sections:

E.1 Java Secure Channel (JSCH) for SSH2

Version: 0.1.44

Vendor: Atsuhiko Yamanaka, JCraft,Inc.

JSch 0.0.* was released under the GNU LGPL license. Later, we have switched over to a BSD-style license.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL JCRAFT, INC. OR ANY CONTRIBUTORS TO THIS SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED

AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

E.2 JViews Diagrammer

Version 8.5.

Vendor: ILOG.

None.

E.3 Velocity Engine

Version: 1.6.4

Vendor: The Apache Software Foundation.

This product is licensed under the Apache 2.0 license agreement. See Section E.3.1, "Apache License Version 2.0".

E.3.1 Apache License Version 2.0

The following applies to all products licensed under the Apache 2.0 License:

You may not use the identified files except in compliance with the Apache License, Version 2.0 (the "License.").

You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. A copy of the license is also reproduced below.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.

Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other

commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright © 1999-2002 The Apache Software Foundation. All rights reserved.; and (ii) the following statement "This product includes software developed by the Apache Software Foundation (http://www.apache.org/)."

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

E.4 Commons Compress

Version 1.1.

Vendor: The Apache Software Foundation.

E.4.1 Apache License Version 2.0

The following applies to all products licensed under the Apache 2.0 License:

You may not use the identified files except in compliance with the Apache License, Version 2.0 (the "License.").

You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. A copy of the license is also reproduced below.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.

Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of,

publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for

reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright © 1999-2002 The Apache Software Foundation. All rights reserved.; and (ii) the following statement "This product includes software developed by the Apache Software Foundation (http://www.apache.org/)."

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

E.5 JSON in Java

Version: None.

Vendor: JSON.org.